Unsupervised Learning Notes

Contents

Introduction to unsupervised learning

Idea of unsupervised learning

1. Slideshow: What is AI, ML, data science, statistics, etc? How does it all fit? Cool diagrams.
2. Supervised learning vs unsupervised learning:
   (a) Supervised learning:
   - Given labeled data, predict a target. Example is predicting a student's GPA from student data.
   - More formally, labeled training data is (X, y) for n samples with relationship
     $$y = f(X) + \epsilon.$$
     Goal is to approximate f via $\hat{f}$. Call supervised because we know answers from our training data.
   - Basic example of linear regression with $p = 1$ for student data. Assume f is linear. Graph with data and best fit line $\hat{f}$. Write general p variable case. Can be used for student intervention (early warning, effect of living on campus, attending class).
   - Main goals are prediction and inference by understanding f (feature importance, model fit and reliability).
   - Linear model is an assumption. Other models include decision trees, neural networks, support vector machine, and more.
   - Well understood area, clear ways to assess quality of results. ISLR text key reference.

   (b) Unsupervised learning:
   - Unsupervised learning is a class of machine learning methods used to discover structure in data
     $$X = \{x_1, x_2, \ldots, x_n\}, \quad x_i \in \mathbb{R}^p \quad \text{(note vector notaiton)}$$
     without labeled outcomes. Instead of predicting a known target, the goal is to explore, summarize, and reveal patterns that are intrinsic to the data.
   - Given unlabeled data X, find structure in the data. No prediction, no supervision. Learning by observation, not by example.
   - What types of students are there given hours studied per week and GPA? What variable combinations belong together (academic, engagement)? Unusual students? Goal is to better understand data.
   - Can be a stand alone analysis or can be used to compliment supervised learning.
3. Core tasks of unsupervised learning:
   (a) Clustering: Group similar observations in the same cluster. K-means, hierarchical, dbscan
   (b) Dimension reduction: Reduce noise and multicolinearity, data viz, large to smaller data, data understanding. PCA, t-SNE, UMAP, SVD. Google tensorflow embedding projector.
   (c) Anomaly detection: Learn distribution of data to quantify outlier probabilities.
4. Concrete examples of unsupervised learning:
   (a) Customer segmentation: Clustering
   - Walmart data on spend average, frequency, mode, product mix, app use.

- No label such as budget shopper or family provider. Want to discover segments rather than predetermine behavior.
- Each data point is a customer in high dimensions.
- Similarity means close distance. Scaling and choice of distance matters.
- Possible clusters: Weekly family stockup, single essentials, deal hunters. These are business driven interpretations.
- Actions: Store layout optimization, personal coupons, inventory planning, regional differences. Not aiming for individual predictions (as with supervised learning).
- Google: Walmart customer segmentation

(b) Spotify music genre: Dimension reduction

- Google: Spotify api dataset
- Many automatic features, how to tell what genre?
- Vectors are high dimension, but human perception is low dimension.
- Can we compress data into low-dimensions?
- Distance reflect song similarity.
- Are there distinct groups of genres or continuous flow?
- Goal is to make data more intelligible.

(c) Credit card fraud: Anomaly detection

- Each data point is a transaction (amount, time, source, location, recent freq, device).
- Millions of these per day, tiny amount are fraud.
- False positive is a problem.
- Does this deviate from expected? Normal behavior is dense regions, but some deviance can naturally occur.
- Distance metric determines deviation from normal.
- Per customer (card?) normalization.
- Action may be to identify fraud patterns to catch more.

5. Much more challenge than supervised learning. No simple goal. Results are subjective. Exploratory, descriptive, and hypothesis-generating rather than predictive.

Readings

1. ISLR 2.1.4, Ch12 thru 12.1
2. HOUL Ch1

Lab

1. EDA and data cleaning, DMCT Ch2 and Ch3

Distance and similarity

Feature scaling and normalization

1. Features on a bigger scale dominate distance calculations. Spend vs weekly visit count for customer segmentation. Illustrate dollars vs thousands for spend on Euclidean distance.

$$d_2(x, y) = \|x - y\|_2 = \sqrt{\sum (x_j - y_j)^2}$$

Many ways to scale to prevent this issue.

2. Standardization ($z-$score scaling):

   (a) Definition: Shift by mean and divide by standard deviation.

   $$x_i' = \frac{x_i - \overline{x}}{s_x}$$

   where s_x is the standard deviation of variable x. Remind of standard deviation calculation. Recall sample vs population notation. Draw distribution picture, two different shapes.

   $$s_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}, \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

   (b) Results in mean 0, variance 1. Now on a standard normal distribution $N(0,1)$. Nice properties of a normal distribution for 68-95-99 percent confidence intervals. Preserves order and relative distribution.

   (c) Equal feature contribution, preserves relative difference.

   (d) Extensions include mean absolute deviation.

3. Min-max scaling:

   (a) Definition: Shift by min and divide by range.

   $$x_i' = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

   (b) Guess what it does. Refer to mall data scaling age.

   (c) Result is in interval $[0,1]$. Preserves order but compresses extremes.

   (d) Good for bounded features.

4. Most of the time scale, can always revers scaling if you keep track of original stats. When scaling is NOT a good idea:

   (a) Feature units are meaningful, important, and want to keep for analysis.

   (b) Binary indicators (0/1), changes to discrete values but different values.

   (c) Counts with semantic meaning, such as a 1-10 satisfaction rating.

   (d) Ratios which are already normalized.

Distance metrics and similarity

1. Distance is a choice we make which encodes different notions of similar. Many options.

2. Key properties are required: Mathematically called a metric in a metric space.

   (a) Positivity: $d(x,y) \geq 0$ for all x, y and $d(x,y) = 0$ only if $y = x$.

   (b) Symmetry: $d(x,y) = d(y,x)$.

   (c) Triangle inequality: $d(x,z) \leq d(x,y) + d(y,z)$ for any x, y, z. Illustrate with pictures, x and z on the horizontal axis.

3. Common distance metrics:

(a) Euclidean distance (ℓ_2 norm): For points $x, y \in \mathbb{R}^d$,

$$d_2(x, y) = \|x - y\|_2 = \sqrt{\sum (x_i - y_i)^2}$$

- Geometry: Straight-line distance, rotation invariant, penalizes large feature deviations heavily.
- Assumes: Features are commensurate, spherical neighborhoods make sense
- Example: Customers close if very same spend pattern and volume. (same shop and spend)

(b) Manhattan distance (ℓ_1 norm):

$$d_1(x, y) = \|x - y\|_1 = \sum |x_i - y_i|$$

- Geometry: City block distance, diamond shaped contours, less sensitive to large feature deviations.
- Assumes: More robust to outliers.
- Example: Customers close if same spend pattern and volume, some diff tolerated. (similar with occasional deviation allowed)

(c) Supremum (ℓ_∞ norm):

$$d_\infty(x, y) = \max\{|x_i - y_i|\}$$

- Geometry: Largest difference only. Shapes are squares.
- Example: Customers close if same spend pattern and volume, one big difference is a problem.

(d) These are all generalized by Minkowski distance:

$$d(x, y) = \|x - y\|_r = \sqrt{\sum (x_i - y_i)^r}$$

Overall, Euclidean is the favorite, but there are good reasons to use others. Show animation in Desmos of different unit circles $x^r + y^r = 1$. $r = \infty$ case is the limit which calc 1 shows supremum norm is found. Show in 2D.

(a) Similarities: Note not a distance, but sometimes makes more sense.

(b) Cosine similarity:

$$sim_{cos}(x, y) = \frac{x \cdot y}{\|x\| \|y\|} = \cos(\theta)$$

- Geometry: measures cosine of angle between rather than distance, lives in $[-1, 1]$, same direction is close to 1, orth and opposite direction, ignores scale. Turns our ordering agrees with Euclidean distance.
- Assumes: Pattern rather than intensity.
- Example: Customers close if same spend pattern but different spend volumes. (shop same regardless of spend)

(c) Correlation: Sample Pearson correlation.

$$sim_{cor}(x, y) = r = cor(x, y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

- Geometry: Measures linear relationship, $r \approx 1$ strong positive linear relationship, $r \approx -1$ strong negative, in between. Book picture vs Wikipedia picture.
- Same as cosine similarity if data is normalized.

Curse of dimensionality

1. In high dimensions, all points become almost equally far apart. Nearest points are almost same distance as farthest.

2. As dimension d increases,

$$\frac{\max d(x,y) - \min d(x,y)}{\min d(x,y)} \to 0$$

Readings

1. IDM Ch 2 Intro, 2.3.7, 2.4.1-2.4.5, 2.4.9-2.4.10 recommend all of Ch 2

2. DMCT Ch 2 Intro, 2.5.1, 2.3.1, 2.3.4, 2.3.7, 2.3.9 recommend all of Ch 2 except 2.6

Lab

Clustering methods

Main reference: DMCT chapter 9

Intro to clustering

1. What is cluster analysis?

   (a) Clustering groups data into clusters so that objects within a cluster are more similar to each other than to objects in other clusters, according to a chosen notion of similarity.

   (b) Clustering is an *ill-posed* problem: there is no single correct solution. Results depend on modeling assumptions, similarity definitions, and analysis goals. There is no universal notion of "true" clusters.

   (c) Similarity (or dissimilarity) is typically defined through a distance or proximity measure.

   (d) Different clustering methods reflect different assumptions about data structure (e.g., partitioning, hierarchical, density-based, grid-based). Some methods that optimize an explicit objective (k-means) while others identify structure without a global objective (hierarchical, DBSCAN).

   (e) Clustering quality can be assessed in multiple ways, including internal criteria, external validation, stability, and interpretability.

   (f) Ongoing research focuses on scalability, high-dimensional settings where distance metrics break, complex cluster shapes, and diverse data types (e.g., text, images).

2. Desiderata for clustering methods:

   (a) Ability to handle different data types (numeric, categorical, mixed)

   (b) Ability to detect non-spherical or non-convex clusters

   (c) Robustness to noise and outliers

   (d) Scalability to large datasets

   (e) Ability to incorporate constraints or side information

   (f) Results that are interpretable and actionable

3. Types of clustering methods

(a) Partitioning methods (customer segmentation, geog segm)

- Assume clusters are compact, well-separated, and cover all data points.
- Partition n objects into k non-overlapping clusters.
- Typically distance-based and solved via iterative optimization.
- Sensitive to initialization, distance choice, and cluster shape assumptions.

(b) Hierarchical methods (fish in Mississippi, species, genus, family, order, class, phylum, kingdom)

- Assume nested cluster structure is meaningful.
- Produce a hierarchy of clusters represented as a tree.
- Agglomerative (bottom-up) or divisive (top-down) approaches.
- Once a merge or split occurs, it cannot be undone.

(c) Density-based methods (social data with natural crowding, maybe no fixed k or hierarchy)

- Assume clusters correspond to regions of high data density separated by low-density regions.
- Clusters are grown based on neighborhood density criteria.
- Naturally identify outliers and allow arbitrary cluster shapes.

(d) Key challenges and limitations:

- Evaluation is inherently difficult due to lack of ground truth; metrics often encode the same assumptions as the algorithm.
- Clustering is exploratory rather than confirmatory.
- Domain knowledge plays a central role (feature selection, scaling, similarity choice, interpretation, clustering alg choice).
- Clustering is not classification, causal inference, or discovery of objective real-world categories; results should not be over-interpreted.

Partitioning methods: K-means and k-medoids clustering

1. Big picture:

   (a) Given data $D = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p$, partition into k disjoint clusters $C_1, \ldots, C_k$, $C_i \cap C_j = \emptyset$.

   (b) Each point belongs to exactly one cluster, and each cluster is summarized by a single representative (center).

   (c) Clustering defined via optimization of an objective function called within-cluster loss.

2. Key assumptions:

   (a) A meaningful distance $d(x, y)$ exists (eg Euclidean distance).

   (b) Clusters are compact and well-separated in the chosen metric.

   (c) Cluster centers summarize cluster geometry.

   (d) All data belong to some cluster (no noise model).

   (e) k is fixed and meaningful.

3. k-means: A centroid-based technique

   (a) Cluster center: centroid (mean) $c_i \in \mathbb{R}^p$.

(b) Objective function to minimize:

$$\min_{C_1,\ldots,C_k} \sum_{i=1}^{k} \sum_{x_j \in C_i} d(x_j, c_i)^2, \quad c_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j.$$

where the distance metric is Euclidean distance (almost always)

$$d(x, y) = \|x - y\| = \sqrt{(x_1 - y_1)^2 + \cdots + (x_p - y_p)^2}.$$

The centroid represents a hypothetical average cluster member (customer).

(c) The within cluster variance is the sum of square errors:

$$WCSS = \sum_{i=1}^{k} \sum_{x_j \in C_i} d(x_j, c_i)^2$$

(d) Algorithm (Lloyd's algorithm):

- This optimization problem is computationally expensive (NP-hard), many ways to select k clusters of n data points. A basic iterative algorithm is used instead.
- Initialize $c_1, \ldots, c_k$ as k random objects from D.
- Cluster assignment step: Belong to nearest centroid.

$$x_j \mapsto \arg\min_i \|x_j - c_i\|^2.$$

- Centroid update step: Average of cluster members.

$$c_i \leftarrow \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j.$$

- Iterate until assignments stabilize.

(e) Each step decreases the objective, converges to a local minimum, may not be global minimum.

(f) Implies Voronoi partition of the feature space, boundaries between clusters where two centroids are equidistant, only depends on centroid not data distribution.

(g) Implicit assumptions: spherical clusters, equal variance, Euclidean geometry.

(h) Variations of k-means involved different distance metrics, smart centroid initialization, and centroid calculation strategies, k-modes for nominal data, groupings of data called microclusters.

4. k-medoids: A representative object-based technique

(a) Motivation: k-means is sensitive to outliers when centroids (means) are calculated.

(b) Cluster center: medoid $o_i \in x_1, \ldots, x_n$. Centroid is now a data point. Also called a representative object.

(c) Objective function to minimize: Note that medoids o must be data points, so computing new centers is not obvious.

$$\sum_{i=1}^{k} \sum_{x_j \in C_i} d(x_j, o_i)$$

Distance $d(\cdot, \cdot)$ need not be Euclidean. Note the lack of squared distance.

(d) More robust to outliers (no averaging).

(e) Example algorithm: Partitioning around medoids PAM

- Random initial medoids. Assign cluster membership. Brute force check all updated medoids per assigned cluster to see which minimizes total distance error. Iterate.

$$WCTD = \sum_{i=1}^{k} \sum_{x_j \in C_i} d(x_j, o_i)$$

- Modification for large data, clustering large applications (CLARA) considers random samples of the dataset.

(f) Slower than k-means due to discrete optimization, especially for large n and large k.

5. k-means vs k-medoids (mathematical contrast)
   - Continuous optimization (means) vs discrete optimization (medoids).
   - Squared Euclidean loss vs general metric loss.
   - Sensitive vs robust to outliers.
   - Fast gradient-like updates vs combinatorial search.
6. Practical issues:
   - Objective function is typically non-convex ⇒ multiple local minima. Repeat iterations can give different cluster solutions.
   - Initialization matters (e.g., random vs k-means++ which chooses initial clusters far apart).
   - Scaling changes the geometry of $|\cdot|$.
   - Choice of k is a modeling decision, not a statistical estimate. If you don't know k, try many and compare results.
7. When partitioning methods work well
   - Clusters roughly convex and isotropic (not stretched in a certain direction).
   - Moderate dimension with meaningful distances, otherwise dimension reduction needed.
   - Clear notion of "center."
   - Need fast baseline clustering.
8. When they fail
   - Non-convex or nested clusters.
   - Unequal cluster variances or densities.
   - Strong outliers (especially k-means).
   - High-dimensional distance concentration.
9. How to determine the right number of clusters? Many ways! Basic discussion here, more later.

   (a) Need k in advance, but the right number of clusters is often ambiguous.

   (b) Good to balance compressibility (simplifying data) with accuracy (cluster meaning and usability).

   (c) An easy approach considers plotting WCSS (within cluster sum of squares) for $k = 2, 3, 4, \ldots, k$ and use the elbow method. Sharp bend says smaller improvements. Expect to decrease to 0. Subjective but something to go by. Better(?) ways!
10. Fun simulator: `https://clustering-visualizer.web.app/kmeans`

Hierarchical Methods

1. Big picture:
   (a) Hierarchical clustering builds a nested sequence of partitions.
   (b) Output is a tree (dendrogram), not a single clustering.
   (c) Clusters exist at multiple resolutions (choices of k).
   (d) No single "best" number of clusters is assumed a priori.
   (e) 2 main types: Agglomerative (bottom-up) and divisive (top-down)

2. Key assumptions:
   (a) Nested structure in the data is meaningful.
   (b) Pairwise dissimilarities capture relevant structure.
   (c) Early decisions (merges or splits) are trustworthy.
   (d) No noise model: all points participate in the hierarchy.

3. Agglomerative hierarchical clustering:
   (a) Algorithm: Bottom-up procedure.
   - Start with n singleton clusters $\{x_1\}, \{x_2\}, \ldots, \{x_n\}$. Compute a pairwise distance matrix.
   - Iterative merge the two closest clusters. The disimilarity between these two clusters indicates the height in the dedrogram where the fusion is places.
   - Computed updated pairwise inter-cluster disimlarities among the $n-1$ remaining clusters. Repeat merging.
   - Merge low in dendrogram means joined clusters were similar. High merge means clusters were dissimilar (separation).

   (b) Cluster-cluster distance (linkage)
   - Requires a linkage function $D(C_a, C_b)$ for clusters C_a, C_b.
   - Common choices:

$$\text{Single / Minimum / Nearest neighbor: } D(C_a, C_b) = \min_{x \in C_a, y \in C_b} d(x, y)$$

   Allows long, thin, winding, non-convex clusters. Connectivity matters, global compactness does not.

$$\text{Complete / Maximum / Farthest neighbor: } D(C_a, C_b) = \max x \in C_a, y \in C_b d(x, y)$$

   Produces compact, spherical clusters and penalizes irregular shapes. Sensitive to outliers.

$$\text{Average: } D(C_a, C_b) = \frac{1}{|C_a||C_b|} \sum_{x \in C_a} \sum_{y \in C_b} d(x, y)$$

$$\text{Mean / Centroid: } D(C_a, C_b) = |m_a - m_b|$$

   Average and mean are compromise between single and complete linkage.

   (c) Linkage choice encodes shape assumptions. Distance metric also important per above discussion (what similar should mean in application).

4. Ward's method (variance-based linkage)

- Merge clusters that miminally increase total within-cluster variance.
- Objective interpretation:

  $$W(C_i, C_j) = WCSS(C_i \cup C_j) - WCSS(C_i) - WCSS(C_j)$$
  $$= \sum_{\vec{x} \in C_i \cup C_j} d(\vec{x}, \vec{c_{ij}})^2 - \sum_{\vec{x} \in C_i} d(\vec{x}, \vec{c_i})^2 - \sum_{\vec{x} \in C_j} d(\vec{x}, \vec{c_j})^2 \quad = \frac{|C_i||C_j|}{|C_i| + |C_j|} \|\vec{c_i} - \vec{c_j}\|^2$$

  where c denotes the clsuter centroid as in k-means. Note only for Euclidean distance for last step.
- Closely related to k-means objective, though indirectly. Best of both worlds in a way.
- Favors compact, spherical clusters.

5. Dendogram: Illustrate example with basic distance measures.

   - Tree structure encoding merge order and merge distances.
   - Vertical height = dissimilarity at which merge occurs. Note near in the horizontal direction does not mean points/clusters are near. Good discussions in ISLR,
   - Cutting the tree at height h induces a partition.
   - Different cuts correspond to different k.

6. How to decide cut?

   - Unlike k-mean, HCA does not optimize a cost function such as WCSS. Other approaches are needed.
   - Dendogram viz is key.
   - Elbow method of linkage distance vs merge step (A sharp increase in linkage distance = merging dissimilar clusters, cut just before the big jump)
   - WCSS works for Ward's linkage because it is inherently close to k-means.
   - Cophenetic Correlation Coefficient (CCC)
     - The cophenetic correlation coefficient (CCC) is specific to hierarchical clustering and is often overlooked.
     - Idea: How well does the dendrogram preserve the original pairwise distances?

       $$\text{CCC} = \text{corr}\Big(d_{ij}, \hat{d}_{ij}\Big)$$

       where d_{ij} is the original distance between points i and j, and $\hat{d}_{ij}$ is the cophenetic distance, defined as the height at which points i and j merge in the dendrogram.
     - Interpretation: Values close to 1 indicate that the hierarchical clustering preserves pairwise distances well. Low values indicate that the dendrogram substantially distorts the geometry of the data.
     - Uses: Comparing linkage methods (single, complete, average, Ward), and choosing an appropriate distance metric.

7. Divisive hierarchical clustering

   - Top-down approach.
   - Start with all points in one cluster.
   - Recursively split clusters.

- Less common due to computational cost.
- Conceptually closer to repeated partitioning.

8. When hierarchical clustering works well
   - Data have meaningful nested or multi-scale structure.
   - Moderate sample size.
   - Interest in relationships between clusters, not just assignments.
9. When it fails
   - Large datasets (computational and memory cost).
   - Strong noise or chaining effects (single linkage).
   - Early incorrect merges propagate upward.
   - Noisy distance measurements.

Density-based clustering: DBSCAN

1. Big picture:
   (a) DBSCAN = Density-based spatial clustering of applications with noise
   (b) Clusters are defined as regions of high point density separated by regions of low density.
   (c) Does not impose global geometry (no centroids, no partition).
   (d) Explicitly allows noise and outliers.
   (e) Number of clusters is determined by the data, not fixed in advanced.
2. Key assumptions:
   (a) A meaningful distance metric exists.
   (b) Clusters correspond to dense regions in the metric space.
   (c) Density is approximately homogeneous within the clusters.
   (d) Low-density regions separate clusters.
3. Parameters:
   (a) $\varepsilon > 0$ (radius parameter).
   (b) minPts $\in \mathbb{N}$ (minimum number of neighbors)
4. Neighborhood definition:
   (a) ε-neighborhood of a point x:
   $$N_\varepsilon(x) = \{y : d(x, y) \leq \varepsilon\}$$
   (b) $|N_\varepsilon(x)|$ gives the density of point x, including the point itself in this count.
5. Point types:
   (a) Core point: At least mnPts within a neighborhood of that point. Point A in diagram if $MinPts \geq 7$.
   $$|N_\varepsilon(x)| \geq \text{minPts}$$

   (b) Border point: Not core, but falls in the neighborhood of a point (within the border), or possibly many points. Point B in diagram.

$$|N_\varepsilon(x)| < \text{minPts}, \quad \text{but } x \in N_\varepsilon(y) \text{ for some core point } y$$

   (c) Noise point: x is neither core nor border. Point C in diagram.

6. Density reachability:
   (a) Directly density-reachable:

$$y \in N_\varepsilon(x), \quad x \text{ is a core point}$$

   (b) Density-reachable: chain of directly density-reachable points.
   (c) Density-connected: two points reachable from a common core point.

7. Cluster definition:
   (a) A cluster is a maximal set of density-connected points.
   (b) Noise points are not assigned to any cluster.

8. Algorithm (conceptual):
   (a) Identify all core points by checking neighborhood density of all possible points.
   (b) Grow clusters by connecting density-reachable points into the same cluster.
   (c) Label remaining points as noise or border.

9. Geometric consequences:
   (a) Can recover non-convex and arbitrarily shaped clusters.
   (b) No forced assignment of all points.
   (c) Cluster boundaries follow low-density regions.
   (d) No global partition space.

10. Comparison to partitioning methods
   (a) No centroids or objective function.
   (b) No Voronoi geometry.
   (c) k not specified.
   (d) Explicit noise handling.

11. Sensitivity and limitations
   (a) Choice of ε and minPts is critical.
   (b) Struggles with varying cluster densities.
   (c) Distance concentration in high dimensions degrades performance.
   (d) Sensitive to distance scaling.

12. When DBSCAN works well
   (a) Clusters separated by low-density regions.
   (b) Non-spherical, irregular shapes.

   (c) Presence of noise or outliers.
   (d) Low-to-moderate dimensional data.

13. When it fails
    (a) Clusters with significantly different densities.
    (b) High-dimensional data.
    (c) Data without clear density gaps.
    (d) Poorly chosen distance metric.

Comparison of k-means, hierarchical, and DBSCAN methods

1. Key idea: Clustering methods are not interchangeable algorithms. They encode fundamentally different notions of what a cluster is.
2. The question each method answers:
   (a) Partitioning (k-means and k-modes): Given k, how should I divide all points to minimize within-cluster dissimilarity?
   (b) Hierarchical clustering: How are points related across multiple scales of similarity?
   (c) DBSCAN: Which points belong to the same dense region, and which points are noise?
3. Mathematics of the machines:
   (a) Partitioning:
   - Force the data into k compact clusters by minimizing within-cluster loss
   - Imposes Voronoi partition where cluster boundaries are hyperplanes
   - Parameters are k, distance metric, scaling, initialization
   - Fails for non-convex shapes, unequal variance, noise

   (b) Hierarchical:
   - Reveal nested similarity structure through greedy merges or splits
   - No global geometric partition and shape depends on linkage choice
   - Parameters are distance metric, linkage, cut height
   - Fails for noise, chaining, large n

   (c) DBSCAN:
   - Identify dense regions separated by low-density gaps and label the rest as noise
   - No partition of space and geometry adapts to data distribution
   - Parameters are ε, minPts, distance metric
   - Fails for varying densities, high dimension

Assessing cluster quality

1. Fundamental difficulty:
   (a) Clustering is unsupervised: typically no ground truth labels, may not even be clusters in the first place.
   (b) ”Good clustering” is not uniquely defined. Many metrics.
   (c) Evaluation criteria often encodes the same assumptions as the algorithm, can be deceptive.

    (d) Different metrics may rank the same clustering very differently.
    (e) Validation is about usefulness and stability, not correctness.

2. Three perspectives on validation:
    (a) Internal validation: use only the data and clustering structure
    (b) External validation: compare to known labels (when available)
    (c) Relative validation: compare multiple clusterings to eachother

3. Internal validation (geometry-based):
    (a) Measures compactness (within-cluster similarity) and separation (between-cluster dissimilarity).
    (b) Common quantities:

$$W = \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - \mu_k\|^2 \quad \text{(within-cluster dispersion)}$$

$$B = \sum_{k=1}^{K} n_k \|\mu_k - \mu\|^2 \quad \text{(between-cluster dispersion)}$$

    These favor spherical, equal-variance clusters.
    (c) Silhouette analysis:
        - For point i:

$$a(i) = \text{ average distance to points in same cluster}$$

$$b(i) = \min_{k \neq c(i)} \text{ average distance to cluster } k$$

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \in [-1, 1]$$

        - $s \approx 1$: well-clustered
        - $s \approx 0$: ambiguous
        - $s < 0$: likely misclustered
        - Assumes distance-based, compact clusters.
    (d) Silhouette is an internal validation metric like W and B, but it evaluates clustering locally at the point level rather than globally at the centroid level.

4.

Readings

1. DMCT Ch10, cluster analysis basic concepts and methods
2. DMCT 10.1, intro to cluster analysis
3. DMCT 10.2, partition methods (k-means and k-medoids)
4. ISL 12.4.1, k-means clustering
5. DMCT 10.3, hierarchical methods
6. ISL 12.4.2, hierarchical clustering

7. DMCT 10.4.1, DBSCAN
8. ISL 12.4.3, practical issues in clustering
9.

Lab