Week 0c: Unsupervised Learning
"The Discovery Challenge"
Finding Hidden Patterns Without Labels

Machine Learning for Smarter Innovation

BSc Course Series

December 14, 2025

Overview: The Discovery Challenge

Part 1: The Challenge

- Customer segmentation without labels
- Mathematical similarity definitions
- No ground truth validation
- Cluster number selection
- Quantitative evaluation metrics

Part 2: K-means Algorithm

- Nearest center assignment
- Worked coordinate examples
- Success: Spherical clusters
- Failure: Non-convex shapes
- Diagnostic insights

Part 3: Density & Hierarchy

- Human clustering intuition
- DBSCAN: Finding neighborhoods
- Hierarchical: Building trees
- Handling arbitrary shapes
- Modern implementations

Part 4: Synthesis

- Method taxonomy
- Algorithm selection guide
- Modern applications
- Neural network preview

24 slides — Pattern discovery without supervision — Real-world clustering challenges

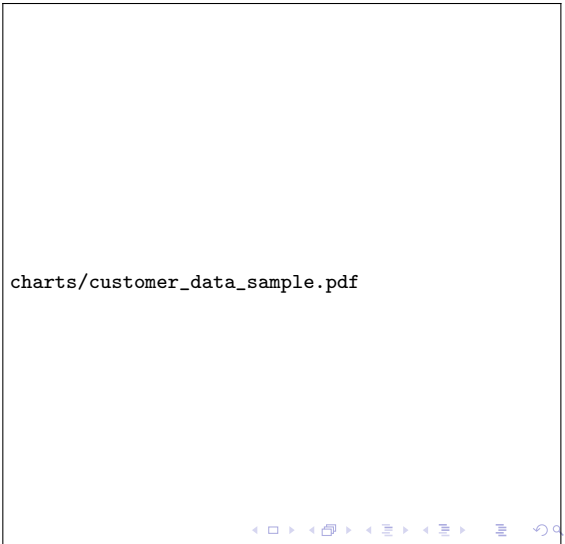
Slide 1: Customer Segmentation Without Labels

The Unsupervised Challenge

- 10,000 customers, no categories
- Purchase history: $amounts, frequency
- Demographics: age, location, income
- Behavioral data: website clicks, time spent

The Question:
"How do we group similar customers when we don't know what similar means?"

Raw Data Sample:



Slide 2: Defining Similarity Mathematically

What Makes Customers Similar?

Euclidean Distance:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \quad (1)$$

Example Calculation:

- Customer A: [$500, 25 visits, age 30]
- Customer B: [$520, 28 visits, age 32]
- Distance $= \sqrt{(500-520)^2 + (25-28)^2 + (30-32)^2}$
- Distance $= \sqrt{400 + 9 + 4} = 20.3$

Distance Visualization:

charts/distance_calculation.pdf

Slide 3: No Ground Truth to Check Against

The Validation Problem

Supervised Learning:

- Training data: (features, labels)
- Test accuracy: Compare predictions vs truth
- Clear success metric

Unsupervised Learning:

- Only features, no labels
- No "correct" clustering exists
- Success is subjective

The Dilemma:

"How do we know if our clusters are good?"

Evaluation Challenge:

Slide 4: Choosing Number of Clusters Problem

The K-Selection Dilemma

Too Few Clusters (k=2):

- Over-generalized segments
- Miss important sub-groups
- Low business actionability

Too Many Clusters (k=50):

- Over-fragmented data
- Noise becomes clusters
- Difficult to interpret

The Sweet Spot:
Meaningful, actionable segments that capture real customer differences.

K-Selection Methods:

charts/elbow_method.pdf

Slide 5: Quantify: Silhouette Scores & Within-Cluster Variance

Internal Validation Metrics

Silhouette Score:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{2}$$

Where:

- $a(i)$: Average distance within cluster
- $b(i)$: Average distance to nearest cluster
- Range: [-1, 1], higher is better

Within-Cluster Sum of Squares (WCSS):

$$WCSS = \sum_{k=1}^{K} \sum_{x \in C_k} ||x - \mu_k||^2 \tag{3}$$

Metric Interpretation:

charts/silhouette_analysis.pdf

Slide 6: Assign to Nearest Center Algorithm

K-means Algorithm Steps

1. Initialize: Place k random centroids **2. Assign:** Each point →nearest centroid **3. Update:** Move centroids to cluster centers **4. Repeat:** Until convergence

Mathematical Foundation:

$$\text{Assign: } c_i = \arg\min_j ||x_i - \mu_j||^2 \tag{4}$$

$$\text{Update: } \mu_j = \frac{1}{|S_j|} \sum_{x_i \in S_j} x_i \tag{5}$$

Where μ_j is centroid j, S_j is cluster j.

Algorithm Visualization:

charts/kmeans_steps.pdf

Slide 7: Worked Example with Actual Coordinates

Step-by-Step Example

Data Points:

- A: (2, 3), B: (3, 4), C: (8, 7), D: (9, 8)

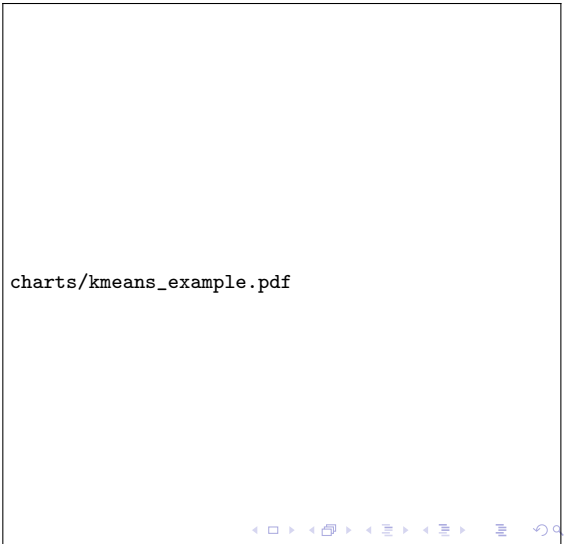
Initial Centroids (k=2):

- μ_1: (1, 1), μ_2: (6, 6)

Iteration 1 - Assignments:

- A to μ_1: $d = \sqrt{(2-1)^2 + (3-1)^2} = 2.24$
- A to μ_2: $d = \sqrt{(2-6)^2 + (3-6)^2} = 5.0$
- A $\rightarrow$Cluster 1

Complete Assignment Table:



Slide 8: [+] SUCCESS: Beautiful on Spherical Clusters

[+] K-means Excels Here

Ideal Conditions:

- Spherical (circular) clusters
- Similar cluster sizes
- Well-separated groups
- Gaussian-distributed data

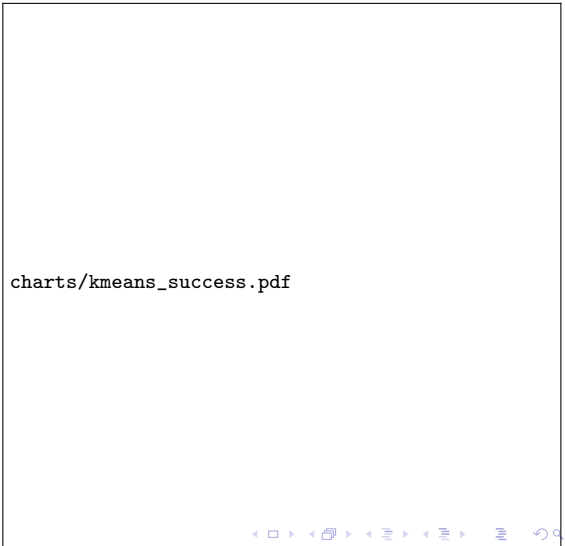
Why It Works:

- Minimizes within-cluster variance
- Natural for spherical boundaries
- Fast convergence
- Stable results

Real Applications:

- Customer segments by spending
- Geographic market regions
- Image color quantization

Perfect K-means Scenario:



Slide 9: [-] FAILURE PATTERN: Breaks on Non-Convex Shapes

[-] K-means Fails Here

Problematic Shapes:

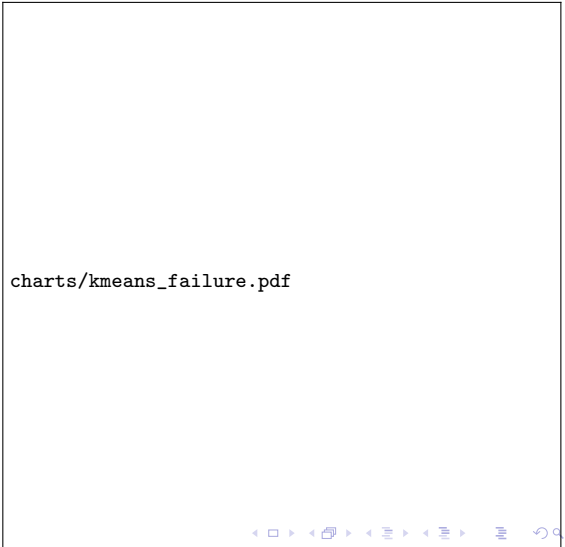
- Crescent/moon shapes
- Elongated clusters
- Nested circles
- Irregular boundaries

Why It Breaks:

- Assumes spherical clusters
- Uses linear decision boundaries
- Centroids pull toward geometric center
- Ignores data density

Crescent Data Example:
Two interlocking crescents →K-means creates artificial vertical split.

Failure Visualization:



Slide 10: Diagnosis: Assumes Convex, Spherical Clusters

K-means Assumptions

Mathematical Constraints:

- Minimizes Euclidean distance to centroids
- Creates Voronoi cell boundaries
- Results in convex cluster shapes
- Equal weight to all dimensions

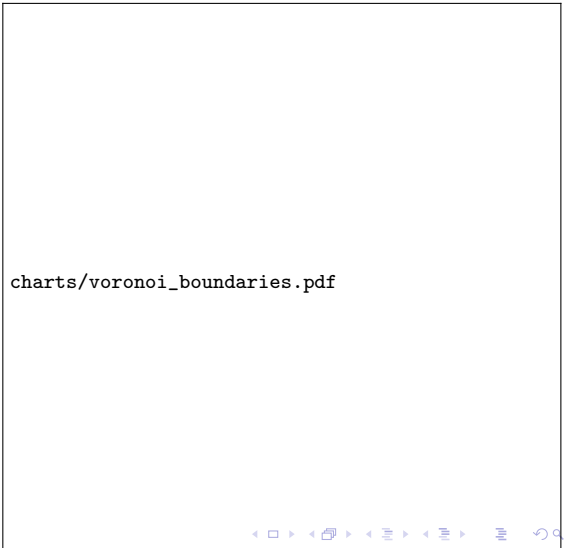
Geometric Intuition:

K-means draws straight lines halfway between cluster centers →Always convex regions.

When to Use K-means:

- Spherical data distributions
- Similar cluster variances
- Fast, scalable solution needed

Voronoi Boundaries:



Slide 11: Human Introspection: How YOU Group by Proximity AND Density

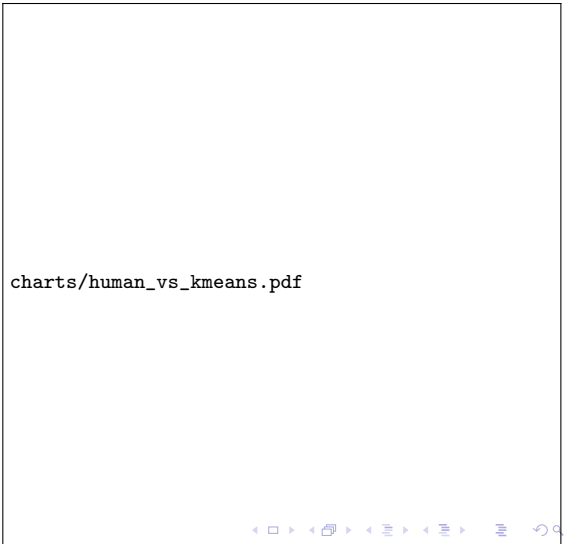
Human Clustering Intuition

How Do You See Groups?

Points close together = same group; Dense regions = natural clusters; Sparse areas = boundaries/noise; Connected components = single cluster

Visual Example: Looking at stars in night sky:
Constellation = dense group; Dark space = natural separator; Isolated stars = outliers
Key Insight: Humans use density, not just distance to centroids.

Human vs K-means Grouping:



Slide 12: Hypothesis: DBSCAN (Density), Hierarchical (Agglomerative)

Alternative Approaches

DBSCAN Hypothesis:

"Clusters are dense regions separated by sparse areas."

Core Principles:

High-density areas = clusters; Low-density areas = boundaries; Isolated points = noise/outliers; No need to specify k

Hierarchical Hypothesis:

"Build clusters by merging similar groups."

Core Principles:

Start with individual points; Merge closest pairs iteratively; Create tree of relationships; Cut tree at desired level

Method Comparison:

charts/method_comparison.pdf

Slide 13: Zero-Jargon: "Find Crowded Neighborhoods"

DBSCAN in Plain English

The Neighborhood Analogy:

- Draw circle around each point
- Count neighbors inside circle
- "Crowded" = many neighbors
- "Sparse" = few neighbors

Simple Rules:

- Core point: Has enough neighbors
- Border point: Near a core point
- Noise point: Not core, not border

Clustering Process:
Connect all core points within neighborhood distance. Add border points to nearest core cluster.

Neighborhood Visualization:

Slide 14: Geometric Intuition: Epsilon-Neighborhoods

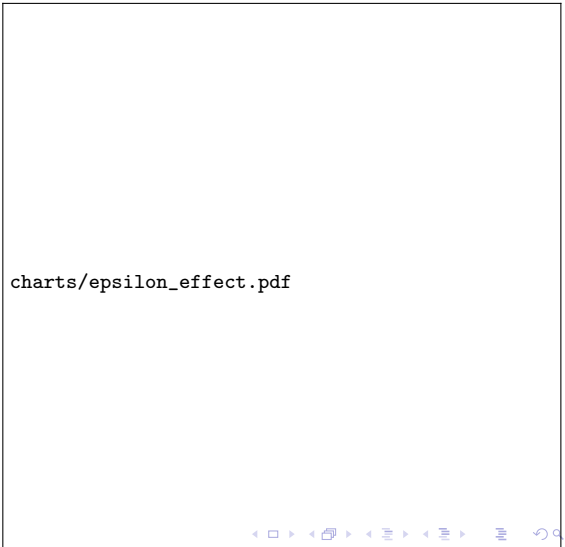
DBSCAN Parameters

Epsilon (ε): Neighborhood radius
Too small = all points are noise; Too large = everything is one cluster; Sweet spot = meaningful neighborhoods

MinPts: Minimum neighbors for core point
Common choice: 2 * dimensions; Higher MinPts = denser clusters; Lower MinPts = more clusters

Geometric Interpretation:
ε-neighborhood = circle of radius ε around each point.

Parameter Effect Visualization:



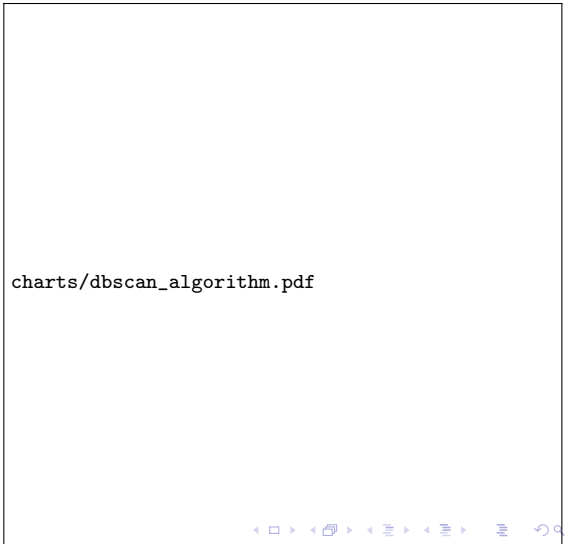
Slide 15: DBSCAN Algorithm Details

DBSCAN Algorithm Steps

1. **Label Points:**
   - Core: $|N_\varepsilon(p)| \geq MinPts$
   - Border: In neighborhood of core point
   - Noise: Neither core nor border
2. **Build Clusters:**
   - Start with unvisited core point
   - Add all density-reachable points
   - Repeat for remaining core points

Density-Reachable:
Point q is density-reachable from p if there's a chain of core points connecting them.

Algorithm Flowchart:



Slide 16: Full Walkthrough: Build Dendrogram with Actual Distances

Hierarchical Clustering Example

Data Points:

- A: (1,1), B: (2,1), C: (4,3), D: (5,4)

Distance Matrix:

	A	B	C	D
A	0	1.0	3.6	5.0
B	1.0	0	2.8	4.2
C	3.6	2.8	0	1.4
D	5.0	4.2	1.4	0

Step 1: Merge A-B (distance = 1.0)
Step 2: Merge C-D (distance = 1.4)
Step 3: Merge (AB)-(CD) (distance = 2.8)

Dendrogram Construction:

charts/dendrogram_example.pdf

DBSCAN Results

Crescent Dataset (DBSCAN):

- Parameters: $\varepsilon = 0.3$, MinPts $= 5$
- Result: 2 crescent-shaped clusters
- Noise points: 15 outliers identified
- Silhouette Score: 0.82

Success Factors:

- Handles non-convex shapes perfectly
- Automatic noise detection
- No assumption about cluster count
- Robust to outliers

Comparison: Same data that broke K-means now correctly clustered.

DBSCAN Cluster Visualization:

charts/dbscan_clusters.pdf

Slide 18: Visualization: Dendrograms

Hierarchical Clustering Results

Customer Segmentation Dendrogram:

- 100 customers, 5 features
- Ward linkage minimizes variance
- Height = dissimilarity measure
- Cut at different levels for k clusters

Reading the Tree:

- Leaves = individual customers
- Height = merge distance
- Branches = cluster relationships
- Cut horizontal line $\rightarrow$k clusters

Business Value: Shows natural customer groupings and relationships.

Customer Dendrogram:

Slide 19: Why It Works: Handles Arbitrary Shapes

Why Density-Based Methods Excel

Flexibility Advantages:

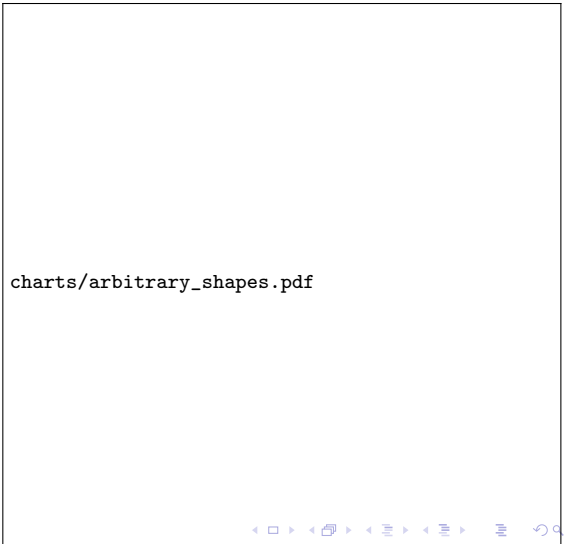
- No geometric assumptions
- Follows data distribution
- Adapts to local density variations
- Separates signal from noise

Real-World Shapes:

- Geographic regions (coastlines)
- Social networks (communities)
- Gene expression patterns
- Anomaly detection boundaries

Mathematical Insight:
Density = local data concentration, not global geometric properties.

Shape Flexibility Demo:



Slide 20: Experimental Validation: Different Cluster Types Table

Comparative Performance Study

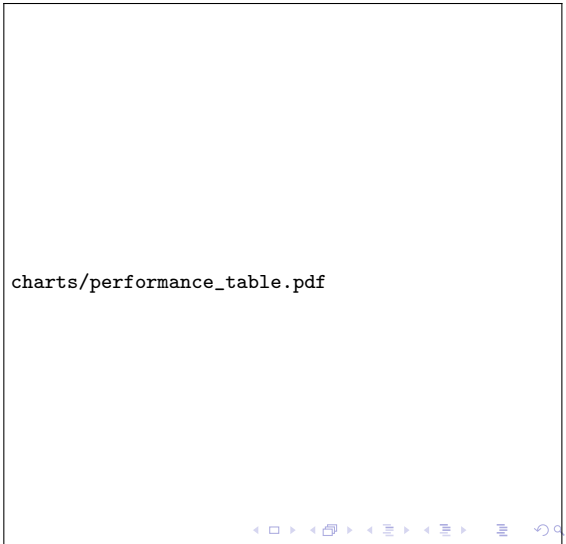
Test Datasets:

- Spherical: Gaussian blobs
- Elongated: Stretched ellipses
- Crescent: Interlocking moons
- Nested: Circles within circles
- Noisy: 20% outliers added

Evaluation Metrics:

- Adjusted Rand Index (ARI)
- Silhouette Score
- Computational Time
- Parameter Sensitivity

Performance Comparison Table:



Slide 21: Implementation: sklearn.cluster

Scikit-learn Implementation

K-means:

`KMeans(n_clusters=3).fit_predict(X)`; `cluster_centers_` for centroids

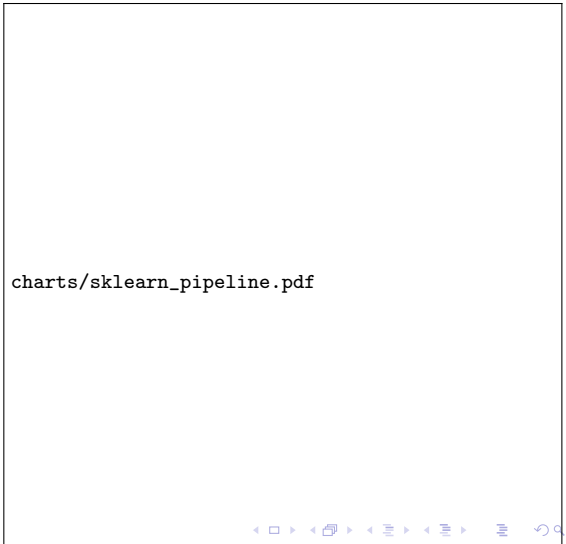
DBSCAN:

```
DBSCAN(eps=0.5, min_samples=5).fit_predict(X)
```

Hierarchical:

```
AgglomerativeClustering(n_clusters=3).fit_predict(X)
```

Production Pipeline:



Slide 22: Clustering Method Taxonomy

Clustering Algorithm Families

Centroid-Based:

- K-means, K-medoids
- Assumes spherical clusters
- Fast, scalable
- Requires k specification

Density-Based:

- DBSCAN, OPTICS, Mean-shift
- Handles arbitrary shapes
- Automatic noise detection
- Parameter sensitive

Hierarchical:

- Agglomerative, Divisive
- Creates cluster tree
- No k pre-specification
- Computationally expensive

Algorithm Taxonomy Tree:

Slide 23: Algorithm Selection Guide

Decision Framework

Ask These Questions:

1. **What shapes do you expect?**
   - Spherical →K-means
   - Arbitrary →DBSCAN
   - Unknown →Hierarchical
2. **Do you know k?**
   - Yes →K-means/Hierarchical
   - No →DBSCAN
3. **How much noise?**
   - Clean data →K-means
   - Noisy data →DBSCAN
4. **What's your dataset size?**
   - Large (¿10K) →K-means
   - Medium →Any method
   - Small (¡1K) →Hierarchical

Selection Decision Tree:

charts/algorithm_selection.pdf

Slide 24: Modern Applications & Neural Network Preview

Real-World Applications

Anomaly Detection:

- Fraud detection in banking
- Network intrusion detection
- Quality control in manufacturing
- Medical diagnosis outliers

Recommendation Systems:

- User behavior clustering
- Product category discovery
- Content similarity grouping
- Market basket analysis

Business Intelligence:

- Customer segmentation
- Market research
- Operational optimization
- Risk assessment

Modern Clustering Evolution: