# Unsupervised Learning: Clustering

## K-means, Hierarchical Clustering, DBSCAN, and Evaluation Metrics

Sarwan Ali

Department of Computer Science
Georgia State University

# Today's Learning Journey


1. Introduction to Unsupervised Learning
2. Clustering Fundamentals
3. K-Means Clustering
4. Hierarchical Clustering
5. DBSCAN
6. Clustering Evaluation Metrics
7. Practical Considerations
8. Real-World Applications
9. Summary and Key Takeaways

# What is Unsupervised Learning?

**Definition:** Learning patterns from data without labeled examples

**Supervised Learning:**

- Has target labels
- Goal: Predict outcomes
- Examples: Classification, Regression

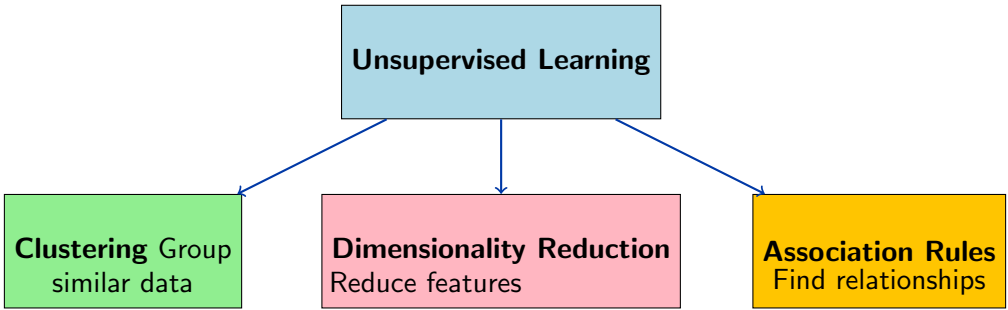
**Unsupervised Learning:**

- No target labels
- Goal: Discover hidden patterns
- Examples: Clustering, Dimensionality Reduction

**Key Insight**

Unsupervised learning helps us understand the **structure** and **relationships** within data

# Types of Unsupervised Learning




**Today's Focus:** Clustering - grouping similar data points together

# What is Clustering?

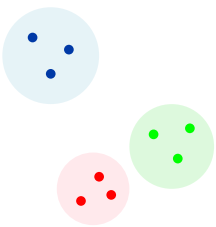
**Definition:** Partitioning data into groups (clusters) where:

- Points within a cluster are **similar**
- Points in different clusters are **dissimilar**

**Applications:**

- Customer segmentation
- Gene sequencing
- Image segmentation
- Social network analysis
- Market research

**Clustered Data**



# Similarity and Distance Measures

**How do we measure similarity?** Through distance metrics!

1. **Euclidean Distance:**

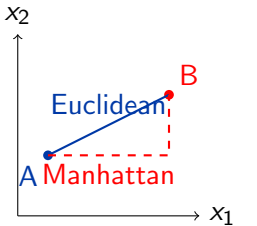
$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

2. **Manhattan Distance:**

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} |x_i - y_i|$$

3. **Cosine Similarity:** $\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{|\mathbf{x}||\mathbf{y}|}$

**Distance Visualization**




**Key Point**

Choice of distance metric significantly affects clustering results!

# K-Means Algorithm Overview

**Goal:** Partition $n$ data points into $k$ clusters

**Key Idea:** Minimize within-cluster sum of squares (WCSS)

$$\text{WCSS} = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} |\mathbf{x} - \boldsymbol{\mu}_i|^2$$

where $C_i$ is cluster $i$ and $\boldsymbol{\mu}_i$ is the centroid of cluster $i$.

**Advantages:**

- Simple and fast
- Works well with spherical clusters
- Scales well to large datasets

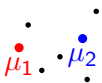
**Disadvantages:**

- Need to specify $k$
- Sensitive to initialization
- Assumes spherical clusters

# K-Means Algorithm Steps

1. **Initialize:** Choose $k$ and randomly place $k$ centroids
2. **Assign:** Assign each point to nearest centroid
3. **Update:** Move centroids to center of assigned points
4. **Repeat:** Steps 2-3 until convergence

**Step 1: Initialize**




**Step 2: Assign**

**Step 3: Update**

# K-Means: Mathematical Formulation

**Objective Function:**

$$J = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} |\mathbf{x} - \boldsymbol{\mu}_i|^2$$

**Centroid Update Rule:**

$$\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\mathbf{x} \in C_i} \mathbf{x}$$

**Assignment Rule:**

$$C_i = \{\mathbf{x} : |\mathbf{x} - \boldsymbol{\mu}_i| \leq |\mathbf{x} - \boldsymbol{\mu}_j| \text{ for all } j\}$$

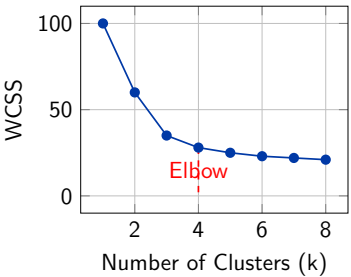
**Convergence**

Algorithm converges when centroids stop moving or maximum iterations reached

# Choosing the Right K

**The Elbow Method:**

1. Run K-means for different values of $k$
2. Plot WCSS vs $k$
3. Look for the "elbow" point
4. Choose $k$ at the elbow




**Other Methods:**

- Silhouette analysis
- Gap statistic
- Domain knowledge

# Hierarchical Clustering Overview

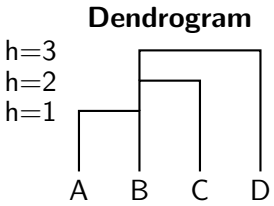
**Builds a hierarchy of clusters without specifying $k$ in advance**

**Agglomerative (Bottom-up):**

- Start: Each point is a cluster
- Iteratively merge closest clusters
- End: One big cluster

**Divisive (Top-down):**

- Start: All points in one cluster
- Iteratively split clusters
- End: Each point is a cluster




**Key Advantage**

Produces a complete clustering hierarchy - can choose any number of clusters

# Linkage Criteria

**How do we measure distance between clusters?**

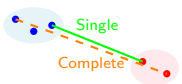
1. **Single Linkage (MIN):**

$$d(C_i, C_j) = \min_{\mathbf{x} \in C_i, \mathbf{y} \in C_j} d(\mathbf{x}, \mathbf{y})$$

2. **Complete Linkage (MAX):**

$$d(C_i, C_j) = \max_{\mathbf{x} \in C_i, \mathbf{y} \in C_j} d(\mathbf{x}, \mathbf{y})$$

**Linkage Types**




3. **Average Linkage:**

$$d(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{\mathbf{x} \in C_i} \sum_{\mathbf{y} \in C_j} d(\mathbf{x}, \mathbf{y})$$

- **Single:** Tends to create elongated clusters (chaining effect)
- **Complete:** Creates compact, spherical clusters
- **Average:** Balanced approach

# Hierarchical Clustering Algorithm

**Agglomerative Clustering Steps:**

1. Start with $n$ clusters (each point is a cluster)
2. Compute distance matrix between all pairs of clusters
3. Merge the two closest clusters
4. Update distance matrix
5. Repeat until one cluster remains

**Time Complexity:** $O(n^3)$ - expensive for large datasets

## Example: Distance Matrix Update

When merging clusters $C_i$ and $C_j$ into $C_{ij}$:

$$d(C_{ij}, C_k) = \alpha_i d(C_i, C_k) + \alpha_j d(C_j, C_k) + \beta d(C_i, C_j) + \gamma |d(C_i, C_k) - d(C_j, C_k)|$$

Different linkage criteria use different values of $\alpha_i, \alpha_j, \beta, \gamma$

# DBSCAN: Density-Based Clustering

**Density-Based Spatial Clustering of Applications with Noise**

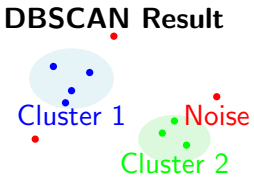
**Key Idea:** Clusters are dense regions separated by sparse regions

**Parameters:**

- $\epsilon$ (eps): Maximum distance for neighborhood
- MinPts: Minimum points to form dense region

**Advantages:**

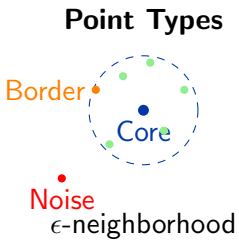
- Finds arbitrary shaped clusters
- Handles noise and outliers
- No need to specify number of clusters



# DBSCAN: Point Classifications

**Three types of points:**

1. **Core Points:**
   - Have at least MinPts points in $\epsilon$-neighborhood
   - Form the "interior" of clusters
2. **Border Points:**
   - Have fewer than MinPts neighbors
   - But are in neighborhood of core point
3. **Noise Points:**
   - Neither core nor border points
   - Considered outliers




**Density Connectivity**

Two points belong to same cluster if there's a path of core points between them

# DBSCAN Algorithm

**Algorithm Steps:**

1. For each unvisited point $p$:
   1. Mark $p$ as visited
   2. Find all points in $\epsilon$-neighborhood of $p$
   3. If neighborhood has $\geq$ MinPts points:
      1. Mark $p$ as core point
      2. Create new cluster with $p$
      3. Add all density-reachable points to cluster
   4. Else if $p$ is in neighborhood of core point: mark as border
   5. Else: mark $p$ as noise

**Time Complexity:** $O(n \log n)$ with spatial indexing, $O(n^2)$ without

## Parameter Selection

- MinPts: Usually set to $2 \times$ dimensions
- $\epsilon$: Use k-distance graph (elbow method)

# Why Evaluate Clustering?

**Challenge:** No ground truth labels in unsupervised learning

**Two Types of Evaluation:**

**Internal Measures:**

- Use only the data itself
- Measure cluster cohesion and separation
- Examples: Silhouette, Davies-Bouldin

**External Measures:**

- Compare with ground truth (if available)
- Measure agreement with true clusters
- Examples: ARI, NMI, Purity

**Goal**

Find clustering that maximizes **intra-cluster similarity** and minimizes **inter-cluster similarity**

# Silhouette Analysis

**Most popular internal clustering evaluation metric**

**For each point $i$:**

- $a(i)$ = average distance to points in same cluster
- $b(i)$ = average distance to points in nearest cluster

**Silhouette coefficient:**

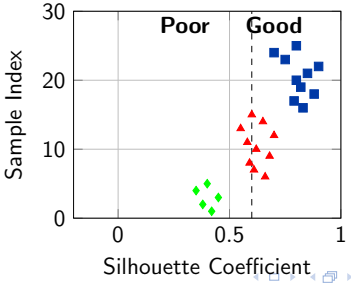
$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

**Interpretation:**

- $s(i) \approx 1$: Well clustered
- $s(i) \approx 0$: On cluster boundary
- $s(i) \approx -1$: Poorly clustered

**Overall Score:**

$$\text{Silhouette} = \frac{1}{n} \sum_{i=1}^{n} s(i)$$



# Davies-Bouldin Index

**Measures average similarity between clusters**

**For clusters $i$ and $j$:**

$$R_{ij} = \frac{S_i + S_j}{M_{ij}}$$

where:

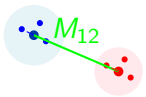
- $S_i$ = average distance from points in cluster $i$ to centroid
- $M_{ij}$ = distance between centroids of clusters $i$ and $j$

**Davies-Bouldin Index:**

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} R_{ij}$$

**Properties:**

- Lower values indicate better clustering
- Range: $[0, \infty)$
- Considers both cohesion and separation




**DB Components**

# External Evaluation Metrics

**When ground truth labels are available**

1. **Adjusted Rand Index (ARI):**

$$ARI = \frac{\text{RI} - E[\text{RI}]}{\max(\text{RI}) - E[\text{RI}]}$$

- Range: $[-1, 1]$, higher is better
- Adjusts for chance agreement

2. **Normalized Mutual Information (NMI):**

$$NMI = \frac{2 \times MI(C, T)}{H(C) + H(T)}$$

- Range: $[0, 1]$, higher is better
- Based on information theory

3. **Purity:** $\text{Purity} = \frac{1}{n} \sum_{i=1}^{k} \max_j |C_i \cap T_j|$

- Range: $[0, 1]$, higher is better
- Simple but biased toward many clusters

# Algorithm Comparison

| Algorithm | Advantages | Disadvantages | Time | Clusters | Best For |
|---|---|---|---|---|---|
| **K-Means** | Simple, Fast, Scalable | Need to specify $k$, Spherical clusters | $O(nkt)$ | Spherical | Large datasets |
| **Hierarchical** | No $k$ needed, Hierarchy | Expensive, Sensitive to noise | $O(n^3)$ | Any shape | Small datasets, Hierarchy |
| **DBSCAN** | Arbitrary shapes, Handles noise | Parameter sensitive | $O(n \log n)$ | Any shape | Irregular clusters |

**Selection Guidelines:**

- **Dataset size:** K-means for large, Hierarchical for small
- **Cluster shape:** K-means for spherical, DBSCAN for irregular
- **Noise tolerance:** DBSCAN best, K-means worst
- **Parameter sensitivity:** Hierarchical least, DBSCAN most

# Data Preprocessing for Clustering

**Critical preprocessing steps:**

1. **Feature Scaling:**
   - Min-Max: $x' = \frac{x - \min}{\max - \min}$
   - Z-score: $x' = \frac{x - \mu}{\sigma}$
   - Robust: $x' = \frac{x - \text{median}}{IQR}$
2. **Handle Missing Values:**
   - Remove incomplete records
   - Impute with mean/median/mode
   - Use algorithms that handle missing data
3. **Dimensionality Reduction:**
   - PCA for linear relationships
   - t-SNE for visualization
   - Feature selection methods
4. **Outlier Detection:**
   - Statistical methods (Z-score, IQR)
   - Distance-based methods
   - Consider domain knowledge

**Warning**

Different preprocessing can lead to completely different clustering results!

# Common Pitfalls and Best Practices

**Common Pitfalls:**

- Not scaling features
- Using wrong distance metric
- Poor parameter selection
- Ignoring domain knowledge
- Over-interpreting results
- Not validating clusters

**Best Practices:**

- Always scale your data
- Try multiple algorithms
- Use multiple evaluation metrics
- Visualize results when possible
- Validate with domain experts
- Document parameter choices

## Validation Strategy

1. Internal metrics (Silhouette, DB Index)
2. Visual inspection (when possible)
3. Domain expert validation
4. Stability analysis (multiple runs)
5. External validation (if labels available)

# Clustering Applications

**Business & Marketing:**

- Customer segmentation
- Market basket analysis
- Recommendation systems
- Fraud detection

**Biology & Medicine:**

- Gene expression analysis
- Drug discovery
- Medical image segmentation
- Disease classification

**Technology:**

- Image segmentation
- Social network analysis
- Web search results
- Anomaly detection

**Science & Research:**

- Astronomy (star classification)
- Climate modeling
- Ecology (species grouping)
- Psychology (behavioral patterns)

**Key Success Factor**

Understanding the domain and having clear objectives for clustering

# Case Study: Customer Segmentation

**Problem:** E-commerce company wants to segment customers for targeted marketing

**Data:** Customer purchase history, demographics, website behavior

**Approach:**

1. **Feature Engineering:** RFM analysis (Recency, Frequency, Monetary)
2. **Preprocessing:** Scale features, handle missing values
3. **Algorithm Selection:** Try K-means, Hierarchical, DBSCAN
4. **Evaluation:** Silhouette analysis, business metrics
5. **Interpretation:** Profile each segment

**Results:**

- High-value customers (10%)
- Regular customers (45%)
- Occasional buyers (35%)
- At-risk customers (10%)

**Business Impact:**

- Personalized marketing
- Retention campaigns
- Product recommendations
- Resource allocation

# Summary

**What we learned today:**

1. **Unsupervised Learning:** Finding patterns without labels
2. **K-Means:** Fast, simple, works well for spherical clusters
3. **Hierarchical:** Creates cluster hierarchy, expensive but flexible
4. **DBSCAN:** Handles noise and arbitrary shapes
5. **Evaluation:** Internal (Silhouette, DB) and External (ARI, NMI) metrics

**Key Principles:**

- No single "best" clustering algorithm
- Preprocessing is crucial
- Always validate results
- Domain knowledge is essential
- Multiple metrics provide better insight

**Remember**

Clustering is exploratory - the goal is to discover meaningful patterns that provide actionable insights

# Next Steps

**To master clustering:**

**Practice:**

- Implement algorithms from scratch
- Work with real datasets
- Experiment with different parameters
- Compare algorithm performance

**Advanced Topics:**

- Spectral clustering
- Gaussian mixture models
- Fuzzy clustering
- Online clustering

**Tools & Libraries:**

- scikit-learn (Python)
- cluster (R)
- WEKA (Java)
- Apache Spark MLlib

**Resources:**

- "Pattern Recognition and Machine Learning" - Bishop
- "The Elements of Statistical Learning" - Hastie et al.
- Online courses and tutorials
- Kaggle competitions

# Thank You!

Questions & Discussion

💡 **Remember:** Clustering is an art as much as it is a science

✉ sali85@student.gsu.edu