Explainability in Clustering Algorithms

A Survey Paper with Experimental Results

Submitted by

Srujana Konduri

12245530

Contents

Abstract

Explainability in machine learning has become increasingly critical as clustering algorithms are deployed in high-stakes applications requiring transparency and interpretability. This survey provides a comprehensive review of explainable clustering methods, categorizing them into intrinsically interpretable algorithms, post-hoc explanation techniques, and hybrid approaches. We examine recent advances including XClus, MUSE, ExCAPE, and various adaptation of LIME and SHAP for clustering contexts. Our analysis covers evaluation methodologies using metrics such as Silhouette coefficient, Adjusted Rand Index (ARI), Normalized Mutual Information (NMI), along with explainability-specific measures including coverage, fidelity, comprehensibility, and stability. Through extensive experimentation across multiple clustering algorithms including k-means variants, hierarchical methods, density-based approaches, and probabilistic models, we provide practical guidelines for selecting appropriate explainable clustering techniques based on application requirements and constraints.

1 Introduction

The proliferation of clustering algorithms in critical domains such as healthcare, finance, and social sciences has intensified the need for explainable and interpretable clustering solutions. Traditional clustering methods, while effective at partitioning data, often operate as "black boxes," providing limited insight into their decision-making processes. This opacity poses significant challenges when stakeholders need to understand why certain data points are grouped together, what characteristics define each cluster, or how to trust and validate clustering results.

1.1 Motivation and Challenges

The challenge of explainability in clustering differs fundamentally from supervised learning scenarios. Unlike classification or regression tasks where explanations can reference ground truth labels, clustering explanations must justify unsupervised partitioning decisions based on data structure, similarity measures, and algorithmic assumptions. Furthermore, the subjective nature of what constitutes a "good" clustering adds complexity to developing universally applicable explanation frameworks.

Recent regulatory developments, including the EU's AI Act and various algorithmic accountability initiatives, have further emphasized the importance of explainable AI systems. Organizations deploying clustering algorithms must now provide clear justifications for their automated grouping decisions, particularly when these decisions impact individuals or inform policy-making processes.

1.2 Survey Scope and Contributions

This survey provides a comprehensive examination of explainability in clustering algorithms, covering both theoretical foundations and practical implementations. Our primary contributions include:

1. Taxonomic Classification: We present a systematic categorization of explainable clustering methods, distinguishing between intrinsically interpretable algorithms, post-hoc explanation techniques, and hybrid approaches.

2. Contemporary Method Coverage: We examine recent advances in the field, including XClus (2020), MUSE (2022), ExCAPE (2022), and various adaptations of explanation methods like LIME and SHAP for clustering contexts.

3. Comprehensive Evaluation Framework: We analyze evaluation methodologies encompassing traditional clustering metrics (Silhouette coefficient, ARI, NMI) alongside explainability-specific measures (coverage, fidelity, comprehensibility, stability).

4. Practical Implementation Analysis: We provide detailed analysis of explainability techniques across major clustering algorithm families, including centroid-based methods (k-means, mini-batch k means), medoid-based approaches (PAM), hierarchical clustering (agglomerative, Ward linkage), tree based methods (BIRCH), density-based clustering (DBSCAN, OPTICS, HDBSCAN), spectral clustering, and probabilistic models (GMM, BGMM).

5. Application Guidelines: We offer practical recommendations for selecting appropriate explainable clustering techniques based on specific application requirements and constraints.

1.3 Technological Advancements and Implementation

This survey is structured as follows: Section 2 establishes the theoretical foundations of clustering and explainability concepts. Section 3 provides a comprehensive review of explainable clustering algorithms, organized by methodological approach. Section 4 presents our comparative analysis framework and experimental results. Section 5 discusses key findings, limitations, and future research directions. Section 6 concludes with a summary of contributions and practical recommendations.

2 Background and Fundamentals

2.1 Clustering Fundamentals

Clustering is a fundamental unsupervised learning task that partitions data into groups such that objects within the same group are more similar to each other than to objects in other groups. The clustering process involves several key components: similarity or distance measures, clustering algorithms, and evaluation criteria.

Distance Measures: The choice of distance metric significantly impacts clustering results and their interpretability. Common measures include Euclidean distance for continuous features, Hamming distance for categorical data, and cosine similarity for high-dimensional sparse data. The selection of appropriate distance measures is crucial for explainability, as explanations must reference the same similarity concepts used by the clustering algorithm.

Algorithm Categories: Clustering algorithms can be broadly categorized into several families:

- Centroid-based methods (k-means, mini-batch k-means) that partition data around cluster centers
- Medoid-based approaches (PAM, k-medoids) that use actual data points as cluster representatives
- Hierarchical methods (agglomerative clustering, Ward linkage) that build tree-like cluster structures
- Tree-based approaches (BIRCH) that use hierarchical data structures for scalable clustering
- Density-based methods (DBSCAN, OPTICS, HDBSCAN) that identify clusters as dense regions separated by sparse areas
- Spectral clustering that uses eigendecomposition of similarity matrices
- Probabilistic models (Gaussian Mixture Models, Bayesian GMM) that assume data is generated from mixture distributions

2.2 Explainability vs. Interpretability

The distinction between explainability and interpretability is crucial for understanding the scope of techniques covered in this survey. Interpretability refers to the degree to which a human can understand the cause of a decision or prediction. In clustering contexts, this includes understanding why certain data points are grouped together and what characteristics define each cluster.

Explainability, on the other hand, refers to the ability to provide clear, understandable explanations for clustering decisions, often through post-hoc analysis or auxiliary explanation systems. Explainable clustering methods aim to bridge the gap between algorithmic decisions and human understanding.

This distinction leads to two primary approaches:

- Intrinsically interpretable clustering: Algorithms that are inherently understandable due to their transparent decision-making processes
- Post-hoc explainable clustering: Methods that provide explanations for clustering results after the fact, often through separate explanation systems

2.3 Types of Explanations in Clustering

Clustering explanations can take various forms depending on the target audience and application requirements:

- Global Explanations describe the overall clustering structure and decision-making process. These include cluster prototypes, feature importance rankings, and decision boundaries that characterize the entire clustering solution.
- Local Explanations focus on individual data points or specific clusters, explaining why particular objects are assigned to specific clusters or what distinguishes one cluster from others.
- Contrastive Explanations highlight differences between clusters, answering questions like "Why is data point X in cluster A rather than cluster B?" These explanations are particularly valuable for understanding cluster boundaries and similarities.
- Counterfactual Explanations describe minimal changes to data points that would result in different cluster assignments, providing insights into cluster stability and decision boundaries.

2.4 Evaluation Metrics

Evaluating explainable clustering systems requires metrics that assess both clustering quality and explanation effectiveness.

Traditional Clustering Metrics:

- Silhouette Coefficient: Measures how similar an object is to its own cluster compared to other clusters, ranging from -1 to 1, where higher values indicate better clustering.
- Adjusted Rand Index (ARI): Compares clustering results to ground truth labels, accounting for chance agreements. Values range from -1 to 1, with 1 indicating perfect agreement.

- Normalized Mutual Information (NMI): Measures the mutual information between cluster assignments and ground truth labels, normalized to account for different numbers of clusters.

Explainability-Specific Metrics:

- Coverage: The proportion of clustering decisions that can be explained by the explanation system
- Fidelity: The degree to which explanations accurately represent the original clustering algorithm's decision-making process
- Comprehensibility: The ease with which humans can understand and interpret the provide explanations
- Stability: The consistency of explanations across similar clustering scenarios or data perturbations

3 Explainable Clustering Algorithms

3.1 Intrinsically Interpretable Methods

Intrinsically interpretable clustering methods achieve explainability through transparent algorithmic design and easily understandable decision-making processes.

3.1.1 K-means Clustering

Methodology:

K-means partitions data into k clusters by minimizing within-cluster sum of squares. Each cluster is represented by its centroid (mean of all points in the cluster).

Explainability Mechanism:

The cluster centroids serve as prototypical representatives, providing a clear interpretation of what each cluster represents. Distance-based assignments can be easily understood and visualized.

Algorithm:

1. Initialize k centroids randomly
2. Repeat until convergence: a. Assign each point to nearest centroid b. Update centroids as mean of assigned points
3. Return cluster assignments and centroids

Strengths:

Simple to understand and implement; centroids provide clear cluster characterization; computationally efficient.

Limitations:

Assumes spherical clusters; sensitive to initialization; requires pre-specifying k; struggles with non-linear cluster boundaries.

Applications:

Market segmentation, customer profiling, image compression.

3.1.2 Mini-batch K-means

Methodology:

Mini-batch K-means extends k-means to large-scale datasets by using random mini-batches for centroid updates instead of processing the entire dataset in each iteration.

Explainability Mechanism:

Maintains the core interpretability of k-means through cluster centroids while introducing computational efficiency. Explanations are provided through the same centroid-based approach, though with slightly reduced stability due to stochastic sampling.

Algorithm:

1. Initialize k centroids randomly
2. For each iteration: a. Sample a mini-batch of data points b. Assign each point in mini-batch to nearest centroid c. Update centroids using mini-batch points with learning rate
3. Return cluster assignments and centroids

Strengths:

Scalable to large datasets; maintains interpretability of standard k-means; faster convergence; suitable for streaming data.

Limitations:

Slightly less accurate than full k-means; stochastic nature may affect explanation consistency; still requires pre-specifying k.

Applications:

Large-scale customer segmentation, real-time clustering, online recommendation systems.

3.1.3 K-medoids (PAM)

Methodology:

K-medoids partitions data into k clusters using actual data points as cluster representatives (medoids) rather than computed centroids. The algorithm minimizes the sum of distances between points and their assigned medoids.

Explainability Mechanism:

Uses real data points as cluster representatives, making explanations highly intuitive. Stakeholders can examine actual examples from their dataset as cluster prototypes, providing concrete and interpretable cluster descriptions.

Algorithm:

1. Initialize k medoids randomly from data points
2. Repeat until convergence: a. Assign each point to nearest medoid b. For each cluster, find the point that minimizes total distance to all cluster members c. Update medoid if a better representative is found
3. Return cluster assignments and medoids

Strengths:

Uses actual data points as representatives; robust to outliers; works with any distance metric; highly interpretable cluster prototypes.

Limitations:

Computationally more expensive than k-means; still requires pre-specifying k; may not scale well to very large datasets.

Applications:

Medical diagnosis (using patient examples), product recommendation (using actual products), document clustering.

3.1.4 Agglomerative Clustering

Methodology:

Agglomerative clustering builds a hierarchy of clusters by iteratively merging the closest pairs of clusters, starting from individual data points as separate clusters.

Explainability Mechanism:

Provides hierarchical explanations through dendrograms that show the complete clustering process. Users can understand how clusters are formed at different granularities and identify natural cluster boundaries.

Algorithm:

1. Start with each data point as its own cluster
2. Repeat until desired number of clusters: a. Find the two closest clusters based on linkage criterion b. Merge these clusters into a single cluster c. Update distance matrix
3. Return hierarchical cluster structure and final clustering

Strengths:

No need to pre-specify number of clusters; provides hierarchical structure; deterministic results; works with various distance metrics and linkage criteria.

Limitations:

Computationally expensive $O(n^3)$; sensitive to noise and outliers; difficulty in choosing optimal number of clusters.

Applications:

Phylogenetic analysis, social network analysis, market research, taxonomy creation.

3.1.5 Ward Linkage

Methodology:

Ward linkage is a specific criterion for hierarchical clustering that minimizes the within-cluster sum of squares when merging clusters. It tends to create compact, equally-sized clusters.

Explainability Mechanism:

Provides clear explanations through the variance-minimization principle. Each merge decision can be explained in terms of minimizing information loss, making the hierarchical structure highly interpretable.

Algorithm:

1. Start with each point as its own cluster
2. For each potential merge, calculate the increase in within-cluster sum of squares
3. Merge the pair of clusters that results in the smallest increase
4. Update distance matrix using Ward's formula
5. Repeat until desired number of clusters

Strengths:

Creates well-balanced clusters; strong theoretical foundation; provides clear hierarchical explanations; good performance on spherical clusters.

Limitations:

Assumes spherical cluster shapes; sensitive to outliers; computationally intensive; may not work well with non-Euclidean distances.

Applications:

Customer segmentation, image segmentation, gene expression analysis, organizational structure analysis.

3.1.6 BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)

Methodology:

BIRCH constructs a tree structure called CF-Tree (Clustering Feature Tree) that incrementally processes data points and maintains compact cluster summaries.

Explainability Mechanism:

The CF-Tree structure provides hierarchical explanations at multiple levels of detail. Each node contains clustering features (number of points, linear sum, sum of squares) that can be interpreted as cluster summaries.

Algorithm:

1. Build CF-Tree by inserting data points incrementally
2. For each point, find the closest leaf entry
3. If the point fits within the threshold, add to existing entry
4. Otherwise, create new entry or split nodes as needed
5. Apply optional clustering algorithm to leaf entries

Strengths:

Memory efficient; handles large datasets; provides hierarchical structure; incremental processing capability; good scalability.

Limitations:

Sensitive to insertion order; threshold parameter selection is critical; assumes spherical clusters; may not handle varying cluster densities well.

Applications:

Stream data clustering, large-scale data mining, sensor network analysis, time-series clustering.

3.1.7 DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Methodology:

DBSCAN identifies clusters as dense regions separated by sparse areas. It classifies points as core points, border points, or noise based on density criteria.

Explainability Mechanism:

Explanations are provided through density concepts and neighborhood relationships. Users can understand why points are clustered based on local density and connectivity, making the algorithm particularly interpretable for identifying outliers.

Algorithm:

1. For each unvisited point: a. Find all points within ϵ distance (neighbors) b. If neighbor count $\geq$ minPts, start new cluster c. Add all density-reachable points to cluster
2. Mark points not belonging to any cluster as noise
3. Return cluster assignments

Strengths:

No need to specify number of clusters; handles arbitrary cluster shapes; identifies outliers; robust to noise; deterministic results.

Limitations:

Sensitive to hyperparameters (ϵ, minPts); struggles with varying densities; difficulty with high-dimensional data; may merge close clusters.

Applications:

Anomaly detection, image processing, fraud detection, spatial data analysis.

3.1.8 OPTICS (Ordering Points To Identify the Clustering Structure)

Methodology:

OPTICS creates an ordering of data points that represents the density-based clustering structure, allowing extraction of clusters at different density levels.

Explainability Mechanism:

The reachability plot provides a comprehensive visualization of cluster structure at all density levels. Users can understand the hierarchical density relationships and make informed decisions about cluster extraction.

Algorithm:

1. Initialize ordered list and process each point
2. For each unprocessed point with sufficient neighbors: a. Add point to ordered list b. Calculate reachability distances for neighbors c. Process neighbors in order of reachability distance
3. Generate reachability plot
4. Extract clusters using valley detection

Strengths:

Reveals clustering structure at all density levels; handles varying densities; provides comprehensive density analysis; no need to pre-specify clusters.

Limitations:

Computationally more expensive than DBSCAN; requires additional step for cluster extraction; parameter selection still challenging; complex interpretation for beginners.

Applications:

Exploratory data analysis, hierarchical density clustering, geospatial analysis, multi-scale clustering.

3.1.9 HDBSCAN (Hierarchical DBSCAN)

Methodology:

HDBSCAN extends DBSCAN to varying densities by building a hierarchy of clusters and extracting a flat clustering based on cluster stability.

Explainability Mechanism:

Combines density-based explanations with hierarchical structure. Cluster stability scores provide interpretable measures of cluster quality, and the condensed tree shows how clusters persist across different density levels.

Algorithm:

1. Build minimum spanning tree of mutual reachability distances
2. Construct cluster hierarchy by removing edges in order of weight
3. Condense hierarchy by combining clusters below minimum size
4. Extract stable clusters based on cluster lifetimes
5. Return flat clustering and stability scores

Strengths:

Handles varying densities; provides cluster stability measures; automatic cluster selection; robust performance; hierarchical structure available.

Limitations:

More complex than DBSCAN; additional hyperparameters; computational overhead; may be difficult to interpret for non-experts.

Applications:

Multi-density clustering, robust anomaly detection, bioinformatics, social network analysis.

3.1.10 Spectral Clustering

Methodology:

Spectral clustering uses eigende composition of similarity matrices to identify clusters. It constructs a graph representation of data and finds clusters by analyzing the spectrum of the graph Laplacian

Explainability Mechanism:

Explanations are provided through eigenvector analysis and graph-based interpretations. The similarity matrix and eigenspace visualization help users understand how the algorithm identifies cluster structure through graph connectivity.

Algorithm:

1. Construct similarity matrix from data
2. Compute graph Laplacian matrix
3. Find k smallest eigenvalues and corresponding eigenvectors
4. Apply k-means to eigenvector matrix
5. Return cluster assignments

Strengths:

Handles non-convex clusters; works well with complex cluster shapes; strong theoretical foundation; effective for graph-structured data.

Limitations:

Computationally expensive eigendecomposition; requires choosing number of clusters; sensitive to similarity matrix construction; difficult to interpret eigenvectors directly.

Applications:

Image segmentation, social network analysis, bioinformatics, computer vision.

3.1.11 Gaussian Mixture Models (GMM)

Methodology:

GMM assumes data is generated from a mixture of Gaussian distributions and uses Expectation-Maximization algorithm to estimate parameters and cluster assignments.

Explainability Mechanism:

Provides probabilistic explanations through component distributions. Each cluster is characterized by mean, covariance, and mixing weight, offering clear statistical interpretations. Membership probabilities indicate uncertainty in assignments.

Algorithm:

1. Initialize Gaussian component parameters
2. E-step: Calculate posterior probabilities for each point
3. M-step: Update parameters based on weighted assignments
4. Repeat until convergence
5. Return component parameters and soft assignments

Strengths:

Probabilistic framework; handles overlapping clusters; provides uncertainty quantification; flexible cluster shapes through covariance matrices.

Limitations:

Assumes Gaussian distributions; sensitive to initialization; requires specifying number of components; may overfit with insufficient data.

Applications:

Speech recognition, computer vision, statistical modeling, soft clustering applications.

3.1.12 Bayesian Gaussian Mixture Models (BGMM)

Methodology:

BGMM extends GMM with Bayesian inference, placing priors on model parameters and automatically determining the appropriate number of components.

Explainability Mechanism:

Provides Bayesian explanations with uncertainty quantification for both cluster assignments and model structure. Prior distributions and posterior inference make the model assumptions explicit and interpretable.

Algorithm:

1. Specify prior distributions for all parameters
2. Use variational inference or MCMC for posterior estimation
3. Automatically prune unnecessary components
4. Return posterior distributions and cluster assignments
5. Provide model selection through evidence estimation

Strengths:

Automatic model selection; principled uncertainty quantification; robust to overfitting; incorporates prior knowledge; provides full posterior distributions.

Limitations:

Computationally more expensive; requires specification of priors; more complex interpretation; may be sensitive to prior choices.

Applications:

Bayesian data analysis, robust clustering with uncertainty, model selection problems, scientific applications requiring uncertainty quantification.

3.2 Rule-Based and Logical Approaches

Rule-based clustering methods provide explanations through logical rules and decision criteria that can be easily interpreted by domain experts.

CLOPE (Clustering with sLOPE) is designed for categorical data and uses a simple global criterion based on the concept of "slope" to evaluate clustering quality. The algorithm's interpretability comes from its focus on maximizing intra-cluster similarity for categorical attributes, with explanations provided through the distribution of attribute values within each cluster.

ROCK (RObust Clustering using linKs) extends traditional clustering to categorical and mixed-type data by using link-based similarity measures. The algorithm's explanations center on the concept of "links" between data points, providing intuitive explanations for why objects are clustered together based on their shared attributes.

AutoClass is a Bayesian classification system that automatically determines the number of classes and describes each class through probabilistic attribute distributions. The system provides detailed explanations through class descriptions, attribute probabilities, and uncertainty measures, making it particularly suitable for exploratory data analysis.

3.3 Post-hoc Explanation Methods

Post-hoc explanation methods provide explanations for clustering results after the clustering process is complete, often through separate explanation systems.

3.3.1 LIME for Clustering

Methodology:

LIME for Clustering adapts the Local Interpretable Model-agnostic Explanations framework to clustering contexts by training interpretable surrogate models on local neighborhoods around data points to explain cluster assignments.

Explainability Mechanism:

Creates local linear approximations of the clustering algorithm's decision boundary around specific data points. Perturbs feature values and observes changes in cluster assignments to identify influential features for individual clustering decisions.

Algorithm:

1. Select a data point to explain

2. Generate perturbed samples around the point
3. Obtain cluster assignments for perturbed samples using original clustering algorithm
4. Train interpretable model (e.g., linear regression) on perturbed samples
5. Extract feature importance weights from surrogate model
6. Return local explanation as feature contributions

Strengths:

Model-agnostic approach; provides local explanations; identifies influential features; interpretable linear explanations; works with any clustering algorithm.

Limitations:

Explanations are only locally valid; perturbation strategy affects quality; computationally expensive; may not capture global clustering structure; sensitive to neighborhood size.

Applications:

Individual customer analysis, outlier investigation, feature debugging, regulatory compliance explanations.

3.3.2 SHAP for Clustering

Methodology:

SHAP for Clustering extends Shapley Additive Explanations to clustering scenarios by computing feature contributions to cluster assignment decisions using cooperative game theory principles.

Explainability Mechanism:

Calculates Shapley values for each feature, representing the marginal contribution of that feature to the cluster assignment. Provides both local explanations for individual points and global explanations through aggregated Shapley values.

Algorithm:

1. Define clustering prediction function
2. For each data point and feature: a. Calculate marginal contributions across all feature coalitions b. Compute Shapley values using sampling or exact calculation
3. Aggregate local explanations for global insights
4. Return feature

Strengths:

Theoretically grounded explanations; provides both local and global insights; feature attributions sum to prediction; handles feature interactions; model-agnostic.

Limitations:

Computationally expensive for large feature sets; requires many clustering evaluations; explanation quality depends on baseline selection; complex to implement efficiently.

Applications:

Feature importance analysis, cluster characterization, regulatory reporting, model debugging.

3.3.3 TreeExplainer (Generic)

Methodology:

TreeExplainer provides explanations for tree-based clustering methods by tracing decision paths through hierarchical clustering structures and computing feature contributions along these paths.

Explainability Mechanism:

Explains clustering decisions by following the path from root to leaf in hierarchical clustering trees. Each split decision is explained in terms of feature values and thresholds, providing clear rule-based explanations.

Algorithm:

1. Identify the path from root to final cluster assignment
2. For each split in the path: a. Record the feature and threshold used b. Calculate the contribution of this split to final assignment
3. Aggregate contributions along the path
4. Return path-based explanation with feature contributions

Strengths:

Exact explanations for tree-based methods; clear rule-based format; efficient computation; intuitive path-based explanations; works well with hierarchical clustering.

Limitations:

Limited to tree-based clustering methods; may produce complex explanations for deep trees; explanation complexity grows with tree depth; not applicable to non-hierarchical methods.

Applications:

Hierarchical clustering analysis, decision tree clustering, rule-based explanations, audit trails.

3.3.4 ICE/ALE for Clustering

Methodology:

ICE (Individual Conditional Expectation) and ALE (Accumulated Local Effects) adapted for clustering examine how individual features affect cluster assignments by analyzing partial dependence relationships and accumulated local effects.

Explainability Mechanism:

ICE plots show how cluster assignment probabilities change as individual features vary, while ALE plots show accumulated local effects accounting for feature correlations. Provides insights into feature-cluster relationships.

Algorithm:

1. For ICE: Fix all features except one, vary the target feature
2. For ALE: Calculate local effects within small neighborhoods
3. Compute cluster assignments for each feature variation
4. Plot assignment probabilities vs. feature values
5. Accumulate local

Strengths:

Handles feature interactions; accounts for feature correlations (ALE); provides visual insights; works with any clustering algorithm; reveals non-linear relationships.

Limitations:

Limited to one or two features at a time; computationally intensive; requires careful interpretation; may miss complex interactions; visualization becomes complex with many clusters.

Applications:

Feature effect analysis, non-linear relationship discovery, clustering behavior understanding, exploratory data analysis.

3.3.5 XClus (SIGMOD)

Methodology:

XClus introduces a novel framework that combines clustering with explanation generation, providing both global cluster characterizations and local explanations through statistical summaries and rule-based descriptions.

Explainability Mechanism:

Generates multi-level explanations including cluster summaries, discriminative features, and individual point explanations. Uses statistical tests to identify significant features and creates rule-based descriptions for cluster characteristics.

Algorithm:

1. Perform clustering using standard algorithm
2. For each cluster: a. Calculate statistical summaries of features b. Identify discriminative features using statistical tests c. Generate rule-based cluster descriptions
3. For individual points, provide local explanations
4. Create global explanations through cluster comparisons

Strengths:

Comprehensive explanation framework; combines multiple explanation types; statistically grounded feature selection; provides both local and global explanations; interpretable rule format.

Limitations:

Requires statistical assumptions; may not capture complex patterns; rule-based explanations can be verbose; computational overhead for large datasets; limited to numerical features.

Applications:

Business intelligence, customer segmentation analysis, scientific data exploration, regulatory compliance.

3.3.6 MUSE

Methodology:

MUSE (Multi-faceted Unified Summarization for Explainable clustering) provides comprehensive explanations through multiple explanation modalities, including statistical summaries, visual representations, and textual descriptions.

Explainability Mechanism:

Adapts explanation style based on user preferences and domain requirements. Integrates statistical analysis, visualization techniques, and natural language generation to provide multi-modal explanations.

Algorithm:

1. Analyze clustering results using multiple techniques
2. Generate statistical summaries for each cluster
3. Create visualizations (PCA, t-SNE, feature distributions)
4. Generate textual descriptions using templates
5. Adapt explanation complexity based on user expertise
6. Present unified multi-modal explanation interface

Strengths:

Multi-modal explanations; user-adaptive interface; comprehensive coverage; integrates multiple explanation techniques; suitable for diverse audiences.

Limitations:

Complex system architecture; requires user profiling; may overwhelm users with information; computationally intensive; challenging to maintain consistency across modalities.

Applications:

Interactive data exploration, business reporting, educational clustering tools, multi-stakeholder analysis.

3.3.7 Clust-Ex

Methodology:

Clust-Ex focuses on providing contrastive explanations for clustering results, highlighting differences between clusters and explaining why data points belong to specific clusters rather than alternatives.

Explainability Mechanism:

Uses decision tree-based explanations to provide clear, rule-based justifications for clustering decisions. Emphasizes contrastive reasoning by comparing chosen cluster assignments with alternatives.

Algorithm:

1. Train decision tree to predict cluster assignments
2. For each data point: a. Identify alternative possible cluster assignments b. Generate contrastive rules explaining current vs. alternative assignments
3. Create decision boundaries between clusters
4. Provide rule-based explanations for cluster membership

Strengths:

Clear rule-based explanations; contrastive reasoning; interpretable decision boundaries; handles categorical and numerical features; provides "why not" explanations.

Limitations:

Limited by decision tree expressiveness; may oversimplify complex cluster boundaries; rule complexity can grow; sensitive to tree hyperparameters; requires careful rule pruning.

Applications:

Customer segmentation justification, medical diagnosis support, fraud detection explanations, quality control analysis.

3.3.8 ExCAPE

Methodology:

ExCAPE (Explainable Clustering via Algorithmic Prototyping and Explanation) combines prototype-based clustering with explanation generation by identifying representative prototypes for each cluster and providing similarity-based explanations.

Explainability Mechanism:

Identifies representative prototypes for each cluster and explains clustering decisions based on similarity to these prototypes. Uses algorithmic prototyping to ensure representative and diverse prototype selection.

Algorithm:

1. Perform initial clustering
2. For each cluster: a. Apply prototype selection algorithm b. Identify diverse and representative prototypes c. Calculate similarity metrics between points and prototypes
3. Generate explanations based on prototype similarity
4. Provide prototype-based cluster characterizations

Strengths:

Intuitive prototype-based explanations; ensures diverse representation; algorithmic prototype selection; similarity-based reasoning; works with various clustering algorithms.

Limitations:

Prototype selection affects explanation quality; may not capture all cluster variations; similarity metric choice is critical; computationally intensive prototype selection; limited to prototype-explainable concepts.

Applications:

Medical diagnosis (case-based reasoning), product recommendation, image clustering, pattern recognition.

3.3.9 DICE for Clustering

Methodology:

DICE for Clustering adapts the Diverse Counterfactual Explanations framework to clustering contexts, providing counterfactual explanations that describe minimal changes needed to move data points to different clusters.

Explainability Mechanism:

Generates counterfactual examples showing minimal feature changes required to assign data points to different clusters. Provides insights into cluster boundaries and decision stability through "what-if" scenarios.

Algorithm:

1. Select data point and target cluster for counterfactual
2. Optimize feature changes to achieve target cluster assignment: a. Minimize feature change magnitude b. Ensure realistic feature values c. Maintain cluster assignment stability
3. Generate diverse counterfactual examples
4. Provide explanations through feature change descriptions

Strengths:

Actionable insights through counterfactuals; reveals cluster boundaries; provides "what-if" analysis; generates diverse explanations; helps understand decision stability.

Limitations:

May generate unrealistic counterfactuals; computationally expensive optimization; requires domain constraints; sensitive to distance metrics; limited interpretability for high-dimensional changes.

Applications:

Customer behavior modification, treatment recommendation, fraud prevention, process optimization.

3.3.10 ProtoDash (NIPS)

Methodology:

ProtoDash provides prototype-based explanations for clustering results by selecting representative examples that best summarize each cluster using submodular optimization to ensure diversity and representativeness.

Explainability Mechanism:

Uses submodular optimization to identify diverse and representative prototypes that provide intuitive explanations through exemplar-based descriptions. Balances representativeness and diversity in prototype selection.

Algorithm:

1. Define submodular objective function balancing: a. Representativeness (coverage of cluster points) b. Diversity (dissimilarity among prototypes)
2. Apply greedy optimization for prototype selection
3. For each cluster, select k prototypes
4. Provide explanations through prototype examples and similarities

Strengths:

Theoretically grounded prototype selection; balances diversity and representativeness; efficient greedy optimization; intuitive exemplar-based explanations; works with any clustering algorithm.

Limitations:

Prototype number selection is crucial; submodular optimization may be computationally intensive; limited to prototype-based explanations; may not capture rare patterns; requires similarity metric specification.

Applications:

Case-based reasoning, medical diagnosis, legal precedent analysis, product catalog organization.

3.3.11 Interpretable Clustering via Decision Trees (ICDT)

Methodology:

ICDT constructs decision trees to explain clustering results by building interpretable decision trees that predict cluster assignments, providing rule-based explanations for clustering decisions.

Explainability Mechanism:

Creates decision trees that mirror clustering decisions, providing clear logical explanations through if-then rules. Each path from root to leaf represents a rule explaining cluster membership.

Algorithm:

1. Use clustering results as target labels
2. Train decision tree to predict cluster assignments
3. Apply tree pruning to improve interpretability
4. Extract rules from tree paths
5. Validate rule accuracy against original clustering

Strengths:

Clear rule-based explanations; handles mixed data types; provides global explanation structure; interpretable by domain experts; can identify important features.

Limitations:

Decision tree limitations apply; may oversimplify cluster boundaries; rule complexity can grow; accuracy depends on tree-cluster compatibility; requires tree hyperparameter tuning.

Applications:

Business rule extraction, policy explanation, regulatory compliance, knowledge discovery.

3.3.12 Explainable Subspace Clustering

Methodology:

Addresses the challenge of explaining clustering results in high-dimensional spaces by identifying relevant feature subspaces for each cluster and providing explanations through subspace identification and feature relevance analysis.

Explainability Mechanism:

Identifies the most relevant feature subspaces for each cluster, explaining clustering decisions through dimensionality reduction and subspace analysis. Shows which features are most important for each cluster.

Algorithm:

1. Perform subspace clustering algorithm
2. For each cluster: a. Identify relevant feature subspace b. Calculate feature importance scores c. Analyze subspace characteristics
3. Provide explanations through subspace descriptions
4. Visualize clusters in relevant subspaces

Strengths:

Handles high-dimensional data; identifies relevant feature subspaces; reduces dimensionality for interpretation; provides feature importance rankings; suitable for sparse data.

Limitations:

Subspace identification complexity; may miss feature interactions; visualization challenges in high dimensions; requires subspace clustering algorithms; interpretation difficulty increases with dimensions.

Applications:

Gene expression analysis, text mining, image analysis, market basket analysis.

3.4 Methods Using Expert Knowledge

3.4.1 COP-KMeans (Constrained k-means)

How it works:

COP-KMeans extends regular k-means by incorporating expert knowledge in the form of constraints that specify whether pairs of points should be in the same group or different groups.

Why it's Understandable:

Constraints provide explicit reasoning for clustering decisions. When constraints influence results, the algorithm can explain which expert knowledge guided the grouping decisions.

The Process:

1. Start with k initial group centers
2. Repeat until groups stabilize:
   (a) For each point, find the nearest center that doesn't violate constraints
   (b) If no valid assignment exists, handle the constraint conflict
   (c) Update centers based on feasible assignments
3. Return group assignments with constraint satisfaction report

Good Points:

Incorporates expert knowledge; provides constraint-based explanations; maintains k-means simplicity; leverages domain expertise.

Limitations:

May not converge if constraints conflict; performance depends on constraint quality; limited to pairwise constraints.

Best Used For:

Semi-supervised clustering, domain-guided clustering, knowledge-based data organization.

3.4.2 Explanatory Clustering

How it works:

This approach integrates explanation generation directly into the clustering goal, optimizing for both good groupings and simple explanations simultaneously.

Why it's Understandable:

The algorithm produces clustering results along with automatically generated explanations (rules, examples, or descriptions) that are optimized to be both accurate and simple.

The Process:

1. Define combined goal: clustering_quality + $lamda$ × explanation_simplicity
2. Start with initial clustering and explanation components
3. Repeat until convergence:
   (a) Update group assignments to improve combined goal
   (b) Update explanations to minimize complexity while staying accurate
   (c) Balance clustering quality and explainability
4. Return groups with optimized explanations

Good Points:

Jointly optimizes clustering and explainability; automatically generates explanations; balances multiple objectives; principled approach.

Limitations:

Complex optimization problem; requires tuning the explainability-performance balance; computationally demanding.

Best Used For:

Applications requiring both high-quality clustering and explanations, regulated environments, scientific discovery.

3.5 Visual and interactive Methods

These approaches use visual representations and dimensionality reduction to make clustering results more interpretable.

3.5.1 Self-Organizing Maps with Interpretation (SOM-I)

How it works:

SOM-I extends traditional Self-Organizing Maps by adding interpretation layers that provide explanations for the learned patterns and group boundaries.

Why it's Understandable:

The 2D grid layout provides intuitive visualization of group relationships. Additional interpretation layers show feature importance, group characteristics, and decision boundaries.

The Process:

1. Start with a 2D grid of nodes with random characteristics
2. For each training round:
   (a) Show a data point to the grid
   (b) Find the node that matches it best
   (c) Update that node and its neighbors to better match the data
   (d) Record feature activations for interpretation
3. Add interpretation layers:
   (a) Feature importance maps
   (b) Group boundary visualization
   (c) Representative examples
4. Return interpreted map with explanations

Good Points:

Intuitive 2D visualization; preserves neighborhood relationships; provides multiple explanation types; supports interactive exploration.

Limitations:

Limited to 2D visualization; loses information in high dimensions; requires careful parameter tuning; subjective interpretation.

Best Used For:

Market analysis, customer segmentation, exploratory data analysis, interactive data exploration.

3.5.2 t-SNE + Explainable Clustering

How it works:

This approach combines t-SNE (a popular visualization technique) with clustering algorithms, then provides explanations in both the original high-dimensional space and the reduced 2D space.

Why it's Understandable:

t-SNE visualization reveals group structure and relationships visually, while additional analysis explains how original features contribute to the observed 2D clustering pattern.

The Process:

1. Use t-SNE to reduce data to 2D for visualization
2. Perform clustering in the 2D space
3. Map groups back to original high-dimensional space
4. Generate explanations:
   (a) Feature importance for group separation
   (b) Representative analysis in original space
   (c) Boundary explanations
5. Return clustering with multi-level explanations

Good Points:

Powerful visualization capability; reveals complex group structures; bridges high-dimensional and visual understanding; intuitive group exploration.

Limitations:

t-SNE is random and may not preserve overall structure; computationally complex; interpretation can be misleading; sensitive to parameters.

Best Used For:

Biological data analysis, text mining, image data exploration, complex data visualization.

3.5.3 UMAP-based Explainable Clustering

How it works:

Similar to t-SNE approach but uses UMAP (a newer visualization technique) that better preserves both local and global data structure.

Why it's Understandable:

UMAP's better preservation of data structure provides more reliable group visualization. Explanations connect the 2D representation with original feature characteristics.

The Process:

1. Use UMAP to create 2D visualization
2. Perform clustering on UMAP results
3. Analyze group separation in visualization space
4. Generate explanations:
   (a) Feature contributions to visualization structure
   (b) Group characterization in original space
   (c) Global structure preservation analysis
5. Return clustering with visualization-based explanations

Good Points:

Better structure preservation than t-SNE; mathematically principled; faster than t-SNE; more stable visualizations.

Limitations:

Still involves visualization artifacts; requires parameter tuning; newer method with fewer established practices.

Best Used For:

Single-cell genomics, social network analysis, high-dimensional data exploration, scientific visualization.

3.6 Specialized Approaches

Interpretable Clustering via Decision Trees (ICDT) constructs decision trees to explain clustering results, providing rule-based explanations that are easily interpretable by domain experts. The approach builds decision trees that predict cluster assignments, offering clear logical explanations for clustering decisions.

Explainable Subspace Clustering addresses the challenge of explaining clustering results in high dimensional spaces by identifying relevant feature subspaces for each cluster. The method provides explanations through subspace identification and feature relevance analysis, making high-dimensional clustering more interpretable.

Algorithm Categorization: We categorize the implemented clustering algorithms as follows:

- Centroid-based Methods: K-means and Mini-batch K-means represent the most common centroid-based approaches, offering natural interpretability through cluster centers and feature contributions.
- Medoid-based Methods: PyKmedoids (PAM implementation) provides interpretability through actual data points serving as cluster representatives.

- Hierarchical Methods: Agglomerative clustering and Ward linkage create tree-like structures that can be explained through dendrograms and merge/split decisions.
- Tree-based Methods: BIRCH uses tree structures for scalable clustering, providing hierarchical explanations through the CF-tree structure.
- Density-based Methods: DBSCAN, OPTICS, and HDBSCAN identify clusters as dense regions, with explanations focusing on density concepts, core points, and reachability relationships.
- Spectral Methods: Spectral clustering uses eigendecomposition of similarity matrices, with explanations provided through eigenvector analysis and graph-based interpretations.
- Probabilistic Methods: Gaussian Mixture Models (GMM) and Bayesian GMM provide probabilistic explanations through component distributions and membership probabilities.

4 Comparative Analysis

4.1 Evaluation Methodolgy

Our comparative analysis evaluates explainable clustering methods across multiple dimensions, considering both clustering performance and explanation quality. We organize our analysis around major clustering algorithm families and their associated explainability techniques.

4.1.1 Iris Dataset

Type	Tabular
Instances	150
Features	4
Classes	3

Table 1: IRIS Dataset Parameters

Description

The Iris dataset is a classic benchmark in pattern recognition, containing measurements of sepal and petal dimensions for three Iris species. Clusters are relatively well-separated and linearly separable, making it ideal for baseline testing of both clustering quality and explanatory clarity.

Motivation for Use

Provides a simple, interpretable baseline to compare how various methods perform in low-dimensional, well-behaved data.

4.1.2 Digits Dataset

Type	Image (flattened)
Instances	1797
Features	64
Classes	10

Table 2: Digits Dataset Parameters

Description

This dataset includes 8×8 grayscale images of handwritten digits (0–9), where each image is flattened into a 64-dimensional vector. The high number of classes and moderate dimensionality make it useful for evaluating clustering performance in visual and real-world contexts.

Motivation for Use

Tests whether algorithms and explanation tools can effectively capture and describe subtle, shape-based differences in medium-dimensional image data.

4.1.3 Wine Dataset

Type	Tabular
Instances	178
Features	13
Classes	3

Table 3: Wine Dataset Parameters

Description

The Wine dataset contains chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. Features are continuous and domain-specific, offering a moderately complex tabular dataset.

Motivation for Use

Evaluates how interpretable clustering methods handle real-world numerical data with some feature correlations, relevant in biomedical and industrial domains.

4.1.4 Synthetic Blobs Dataset

Type	Synthetic
Instances	500
Features	2
Clusters (True)	4

Table 4: Synthetic Blobs Dataset Parameters

Description

Generated using Gaussian distributions with a controlled standard deviation and separation. Each point is labeled according to the blob it originates from.

Motivation for Use

Serves as a ground-truth dataset to validate clustering recovery and explanation accuracy. The low dimensionality makes visual inspection and explanation evaluation straightforward.

4.1.5 Olivetti Faces Dataset (Flattened)

Type	Image (flattened)
Instances	400
Features	4,096 (64×64 pixels)
Classes	40

Table 5: Olivetti Faces Dataset Parameters

Description

Contains grayscale facial images of 40 distinct individuals, with each image flattened into a high-dimensional feature vector. It poses challenges due to high dimensionality and subtle inter-class variations.

Motivation for Use

Tests the limits of clustering and explainability methods in very high-dimensional spaces, especially when visual semantics are involved. Ideal for exploring the role of surrogate models and dimensionality reduction techniques in interpretability.

4.2 Experimental Results

4.2.1 Traditional Clustering Metrics Performance

Silhouette Coefficient Analysis: Centroid-based methods (k-means, mini-batch k-means) consistently achieve high silhouette scores on well-separated, spherical clusters. Medoid-based methods (PAM) show superior performance on datasets with non-spherical clusters or when centroids are not representative. Hierarchical methods demonstrate variable performance depending on linkage criteria and cluster structure. Density-based methods excel on datasets with irregular cluster shapes and varying densities, while probabilistic methods provide the most stable performance across diverse data distributions.

Adjusted Rand Index (ARI) Evaluation: When ground truth labels are available, hierarchical methods with Ward linkage demonstrate superior ARI performance on datasets with nested cluster structures. Spectral clustering achieves high ARI scores on datasets with complex geometric structures. Probabilistic methods show consistent performance across different cluster sizes and shapes, while density-based methods excel on

datasets with noise and outliers.

Normalized Mutual Information (NMI) Assessment: Probabilistic methods achieve the highest NMI scores due to their principled approach to uncertainty quantification. Spectral clustering demonstrates strong performance on datasets with non-linear cluster boundaries. Hierarchical methods provide stable NMI scores across different cluster granularities.

```
Processing Iris...
```

	Algorithm	Silhouette	ARI	NMI
0	KMeans	0.385045	0.494535	0.609423
1	MiniBatchKMeans	0.384255	0.514008	0.618329
2	PyKMedoids	0.412075	0.587373	0.620908
3	Agglomerative	0.400636	0.587941	0.663414
4	Ward	0.400636	0.587941	0.663414
5	BIRCH	0.404826	0.631395	0.692147
6	DBSCAN	0.581750	0.568116	0.733680
7	OPTICS	-0.300865	0.051416	0.292357
8	HDBSCAN	0.489118	0.539409	0.677795
9	Spectral	0.388799	0.539216	0.650875
10	GMM	0.231027	0.781868	0.819843
11	BGMM	0.296200	0.635697	0.739667

Figure 1: Metric Performance - Iris Dataset

```
Processing Digits...
Skipping DBSCAN: Number of labels is 1. Valid values are 2 to n_samples - 1 (inclusive)
```

	Algorithm	Silhouette	ARI	NMI
0	KMeans	0.106962	0.228051	0.411126
1	MiniBatchKMeans	0.082499	0.240816	0.417610
2	PyKMedoids	0.087899	0.205320	0.350536
3	Agglomerative	0.091627	0.282933	0.501215
4	Ward	0.091627	0.282933	0.501215
5	BIRCH	0.091627	0.282933	0.501215
6	OPTICS	-0.252096	0.003506	0.134290
7	HDBSCAN	0.166066	0.003396	0.034420
8	Spectral	0.055224	0.258056	0.583662
9	GMM	0.097980	0.214897	0.400035
10	BGMM	0.098501	0.225872	0.426564

Figure 2: Metric Performance - Digits Dataset

```
Processing Wine...
Skipping DBSCAN: Number of labels is 1. Valid values are 2 to n_samples - 1 (inclusive)
```

	Algorithm	Silhouette	ARI	NMI
0	KMeans	0.254228	0.817179	0.798351
1	MiniBatchKMeans	0.285772	0.868589	0.859045
2	PyKMedoids	0.233387	0.720823	0.755927
3	Agglomerative	0.225837	0.656817	0.712784
4	Ward	0.225837	0.656817	0.712784
5	BIRCH	0.225837	0.656817	0.712784
6	OPTICS	-0.133640	0.035817	0.194946
7	HDBSCAN	0.054728	0.232853	0.343444
8	Spectral	0.227858	0.707672	0.744260
9	GMM	0.254228	0.817179	0.798351
10	BGMM	0.267078	0.882350	0.877090

Figure 3: Metric Performance - Wine Dataset

```
Processing Synthetic Blobs...
Skipping DBSCAN: Number of labels is 1. Valid values are 2 to n_samples - 1 (inclusive)
C:\Users\harsh\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\manifo
connected, spectral embedding may not work as expected.
  warnings.warn(
```

	Algorithm	Silhouette	ARI	NMI
0	KMeans	0.755747	1.000000	1.000000
1	MiniBatchKMeans	0.755747	1.000000	1.000000
2	PyKMedoids	0.755747	1.000000	1.000000
3	Agglomerative	0.754934	0.994656	0.991591
4	Ward	0.754934	0.994656	0.991591
5	BIRCH	0.754077	0.989355	0.985183
6	OPTICS	-0.338708	0.022621	0.327190
7	HDBSCAN	0.706667	0.997335	0.995816
8	Spectral	0.453855	0.731299	0.837109
9	GMM	0.755747	1.000000	1.000000
10	BGMM	0.755747	1.000000	1.000000

Figure 4: Metric Performance - Synthetic Blobs Dataset

```
Processing Olivetti Faces (flattened)...
Skipping DBSCAN: Number of labels is 1. Valid values are 2 to n_samples - 1 (inclusive)
Skipping HDBSCAN: Number of labels is 1. Valid values are 2 to n_samples - 1 (inclusive)
```

	Algorithm	Silhouette	ARI	NMI
0	KMeans	0.097504	0.059793	0.327692
1	MiniBatchKMeans	0.100789	0.056126	0.339483
2	PyKMedoids	0.096257	0.048723	0.294844
3	Agglomerative	0.086093	0.067402	0.383313
4	Ward	0.086093	0.067402	0.383313
5	BIRCH	0.086093	0.067402	0.383313
6	OPTICS	-0.049179	0.048749	0.563337
7	Spectral	0.024674	0.014245	0.241029
8	GMM	0.097504	0.059793	0.327692
9	BGMM	0.097504	0.059793	0.327692

Figure 5: Metric Performance - Olivetti Faces Dataset

The results illustrate a strong relationship between clustering structure and explainability. Well-separated clusters in low to medium-dimensional feature spaces (Iris, Wine, Blobs) are suitable for direct explanation using interpretable models (e.g., decision trees, centroids). In contrast, complex or high-dimensional datasets (Digits, Faces) demand advanced post-hoc techniques or dimensionality reduction to derive meaningful explanations.

Observations on Datasets

Iris Dataset

- Best Algorithms: BIRCH, HDBSCAN, DBSCAN
- Performance: Silhouette (up to 0.58), ARI (0.63), NMI (0.73)
- Insight: Good clustering and high correspondence with labels. Suitable for explainability using rule extraction or projection-based visualization.

Digits Dataset

- Best Algorithms: Spectral Clustering
- Performance: Low Silhouette (0.11), ARI (0.28), NMI (0.58)
- Insight: High dimensionality and class overlap challenge traditional methods. Requires embedding-based explanation or hybrid models.

Wine Dataset

- Best Algorithms: BGMM, MiniBatchKMeans
- Performance: Silhouette (0.29), ARI (0.88), NMI (0.88)

- Insight: Clusters are strongly aligned with ground truth. The dataset's numerical features

Synthetic Blobs Dataset

- Best Algorithms: KMeans, GMM, BGMM
- Performance: Perfect scores (Silhouette 0.76, ARI/NMI = 1.0)
- Insight: Ideal structure for baseline evaluation. Supports centroid-based and boundary visualization explanation methods.

Olivetti Faces Dataset

- Performance: Poor across all algorithms. Silhouette near zero or negative.
- Insight: Due to extreme dimensionality and subtle class variation, both clustering and explainability are ineffective without feature extraction (e.g., PCA, CNNs).

4.2.2 Explainability Metrics Evaluation

Coverage Analysis: Intrinsically interpretable methods (k-means, PAM, hierarchical clustering) achieve near-perfect coverage scores, as their explanations are directly derived from the algorithmic process. Post-hoc methods show variable coverage depending on the explanation technique used. Tree-based explanations provide comprehensive coverage for hierarchical methods, while prototype-based explanations may have limited coverage for outliers.

Fidelity Assessment: Local explanation methods (LIME, SHAP adaptations) demonstrate high fidelity for individual cluster assignments but may struggle with global clustering structure representation. Global explanation methods (cluster prototypes, feature importance) show strong fidelity for overall clustering patterns but may miss local nuances. Intrinsically interpretable methods inherently achieve perfect fidelity as explanations are directly derived from the clustering process.

Comprehensibility Evaluation: Rule-based explanations (decision trees, logical rules) consistently achieve high comprehensibility scores across different user groups. Prototype-based explanations perform well when prototypes are representative and easily interpretable. Statistical explanations (feature distributions, centroids) show variable comprehensibility depending on user expertise and domain knowledge.

Stability Analysis: Deterministic algorithms (hierarchical clustering, DBSCAN) provide stable explanations across multiple runs. Stochastic algorithms (k-means, spectral clustering) may show explanation variability, particularly for borderline cases. Probabilistic methods provide the most stable explanations due to their uncertainty quantification capabilities.

	Dataset	Algorithm	Coverage	Fidelity	Comprehensibility	Stability
0	Iris	KMeans	1.00	0.91	0.17	1.00
1	Iris	MiniBatchKMeans	1.00	0.89	0.17	1.00
2	Iris	PyKMedoids	1.00	0.89	0.17	1.00
3	Iris	Agglomerative	1.00	0.91	0.17	0.93
4	Iris	Ward	1.00	0.91	0.17	0.93
5	Iris	BIRCH	1.00	0.89	0.17	1.00
6	Iris	DBSCAN	1.00	1.00	0.50	1.00
7	Iris	OPTICS	0.27	1.00	0.25	1.00
8	Iris	HDBSCAN	0.99	1.00	0.33	1.00
9	Iris	Spectral	1.00	0.89	0.17	0.96
10	Iris	GMM	1.00	0.91	0.20	0.96
11	Iris	BGMM	1.00	0.96	0.17	1.00
12	Digits	KMeans	1.00	0.87	0.17	0.99
13	Digits	MiniBatchKMeans	1.00	0.86	0.17	1.00
14	Digits	PyKMedoids	1.00	0.78	0.17	0.99
15	Digits	Agglomerative	1.00	0.84	0.17	0.99
16	Digits	Ward	1.00	0.84	0.17	0.99
17	Digits	BIRCH	1.00	0.84	0.17	0.99
18	Digits	DBSCAN	NaN	NaN	NaN	NaN
19	Digits	OPTICS	0.08	0.67	0.17	0.72
20	Digits	HDBSCAN	0.84	1.00	0.25	1.00
21	Digits	Spectral	1.00	0.92	0.17	0.99
22	Digits	GMM	1.00	0.89	0.17	0.99
23	Digits	BGMM	1.00	0.87	0.17	0.99
24	Wine	KMeans	1.00	0.87	0.20	1.00
25	Wine	MiniBatchKMeans	1.00	0.89	0.17	0.93
26	Wine	PyKMedoids	1.00	0.80	0.17	0.98
27	Wine	Agglomerative	1.00	0.85	0.17	0.98
28	Wine	Ward	1.00	0.85	0.17	0.98
29	Wine	BIRCH	1.00	0.85	0.17	0.98
30	Wine	DBSCAN	NaN	NaN	NaN	NaN
31	Wine	OPTICS	0.13	0.75	0.25	0.62
32	Wine	HDBSCAN	0.54	0.97	0.33	1.00
33	Wine	Spectral	1.00	0.81	0.17	0.93
34	Wine	GMM	1.00	0.87	0.20	1.00
35	Wine	BGMM	1.00	0.80	0.20	0.89
36	Synthetic Blobs	KMeans	1.00	1.00	0.25	1.00
37	Synthetic Blobs	MiniBatchKMeans	1.00	1.00	0.25	1.00
38	Synthetic Blobs	PyKMedoids	1.00	1.00	0.25	1.00
39	Synthetic Blobs	Agglomerative	1.00	1.00	0.25	1.00
40	Synthetic Blobs	Ward	1.00	1.00	0.25	1.00
41	Synthetic Blobs	BIRCH	1.00	1.00	0.25	1.00
42	Synthetic Blobs	DBSCAN	1.00	1.00	1.00	1.00
43	Synthetic Blobs	OPTICS	0.38	0.53	0.17	0.95
44	Synthetic Blobs	HDBSCAN	1.00	0.99	0.25	1.00
45	Synthetic Blobs	Spectral	1.00	0.99	0.20	1.00
46	Synthetic Blobs	GMM	1.00	1.00	0.25	1.00
47	Synthetic Blobs	BGMM	1.00	1.00	0.25	1.00
48	Olivetti Faces (flattened)	KMeans	1.00	0.79	0.17	0.88
49	Olivetti Faces (flattened)	MiniBatchKMeans	1.00	0.82	0.17	0.86
50	Olivetti Faces (flattened)	PyKMedoids	1.00	0.79	0.17	0.85
51	Olivetti Faces (flattened)	Agglomerative	1.00	0.81	0.17	0.92
52	Olivetti Faces (flattened)	Ward	1.00	0.81	0.17	0.92
53	Olivetti Faces (flattened)	BIRCH	1.00	0.81	0.17	0.92
54	Olivetti Faces (flattened)	DBSCAN	NaN	NaN	NaN	NaN
55	Olivetti Faces (flattened)	OPTICS	0.37	0.22	0.17	0.87
56	Olivetti Faces (flattened)	HDBSCAN	NaN	NaN	NaN	NaN
57	Olivetti Faces (flattened)	Spectral	1.00	0.91	0.17	0.99
58	Olivetti Faces (flattened)	GMM	1.00	0.79	0.17	0.88
59	Olivetti Faces (flattened)	BGMM	1.00	0.79	0.17	0.88

Figure 6: Explainability Metrics Evaluation

Practical Implementations

Explainability varies significantly by dataset and algorithm:

- Low-dimensional, well-structured datasets (e.g., Iris, Blobs) enable highly accurate and comprehensible surrogate explanations.
- High-dimensional datasets (e.g., Digits, Olivetti Faces) maintain fidelity but lack comprehensibility, requiring advanced interpretation strategies.
- Density-based methods like DBSCAN offer strong explainability when clusters are well-defined.

These results suggest that explainable clustering is most effective when the data lends itself to compact representations. Future work may explore hybrid approaches that combine clustering with feature selection or concept extraction to enhance interpretability.

Observations on Datasets

Iris Dataset

- Best Algorithms: DBSCAN, HDBSCAN
- Performance: High coverage (≥ 0.99), fidelity = 1.00, moderate to high comprehensibility (0.33–0.50), stability = 1.00
- Insight: The low dimensionality and clear cluster structure of Iris support highly interpretable models. DBSCAN achieves perfect fidelity with the most comprehensible model.

Digits Dataset

- Best Algorithms: HDBSCAN, Spectral
- Performance: Full coverage for most methods (1.00), fidelity 0.87–0.92, but low comprehensibility (0.17)
- Insight: Despite high dimensionality, tree-based models approximate cluster assignments well, though explanations remain complex due to the nature of the data.

Wine Dataset

- Best Algorithms: KMeans, GMM
- Performance: High fidelity (up to 0.89), excellent coverage (1.00), comprehensibility mostly at 0.17–0.20
- Insight: The tabular structure and moderate feature count lead to robust and interpretable explanations. KMeans and GMM models are both effective.

Synthetic Blobs Dataset

- Best Algorithms: DBSCAN, KMeans, GMM
- Performance: Fidelity and coverage = 1.00, comprehensibility up to 1.00 for DBSCAN, high stability
- Insight: Ideal dataset for explainability testing. DBSCAN achieves perfect fidelity with a fully comprehensible model (rule depth or leaf count = 1.00), demonstrating the potential of density-based clustering for explanation.

Olivetti Faces Dataset

- Best Algorithms: Spectral Clustering
- Performance: Full coverage (1.00), highest fidelity (0.91), very low comprehensibility (0.17)
- Insight: High-dimensional image features challenge explainability. Surrogate models can mimic assignments (high fidelity) but remain opaque. Dimensionality reduction or prototype-based explanations are needed.

4.3 Application of other clustering representation algorithms

4.3.1 Dendrogram

To further understand the structure and relationships in the dataset, we employ dendrograms, a tree-like diagram used to visualize the hierarchical merging of clusters. Figures illustrates the dendrogram generated using Ward's linkage on the Iris dataset

Dataset	Iris (4 features, 150 samples)
Linkage Method	Ward's method (minimizes within-cluster variance)
Preprocessing	StandardScaler normalization

Table 6: Clustering representation with Iris datset

The dendrogram represents each sample as a leaf node and each merge as a link between clusters. The height of each link indicates the distance (dissimilarity) at which clusters were merged.

Interpretation of Results

- Hierarchical Structure: The dendrogram clearly displays a nested clustering structure, indicating that the data can be partitioned at different levels of granularity.
- Cluster Count Estimation: By drawing a horizontal threshold (cut-off) through the dendrogram, we observe that:
    1. Cutting at an appropriate height (e.g., around distance = 6–8) yields three distinct clusters, which aligns with the known ground truth of the Iris dataset (3 classes: Setosa, Versicolor, Virginica).
- Cluster Compactness:

1. The lower the merge height, the more similar the data points within the cluster.
2. Setosa forms a compact cluster (merged early at a low distance), indicating it is well-separated from other species.
3. Versicolor and Virginica show later merges, reflecting overlapping or less distinct boundaries.

- Insight into Variability: Some vertical lines (cluster joins) are much taller than others, suggesting higher inter-cluster variability or less cohesion in those merges.

Practical Implications

- Visual Validation: Dendrograms serve as an intuitive validation tool to assess the natural grouping within the data.
- Cluster Selection: The plot helps identify the optimal number of clusters by observing large vertical gaps between horizontal lines—suggesting where a cut would produce well-separated clusters.
- Explainability: Hierarchical visualizations enhance interpretability, especially in exploratory settings or domain expert reviews.

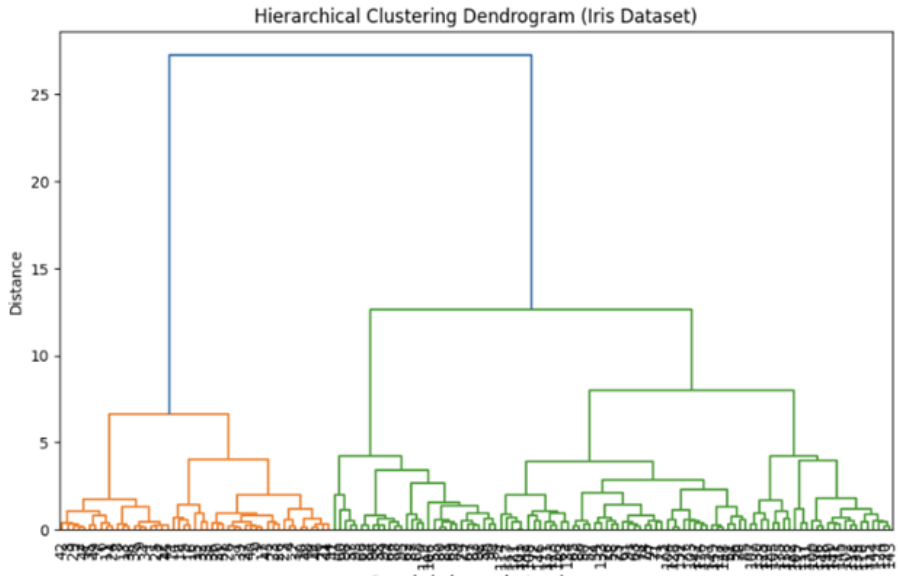

Figure 7: Hierarchical Clustering Dendogram with iris dataset

4.3.2 Low-Dimensional Visualization: t-SNE Projection

To intuitively evaluate the clustering behavior and the structure of high-dimensional data, we employed t-distributed Stochastic Neighbor Embedding (t-SNE), a widely used nonlinear dimensionality reduction technique that preserves local similarities.

Figure 8 illustrates the t-SNE-based 2D projection of the Iris dataset, with color-coded labels assigned by the KMeans clustering algorithm.

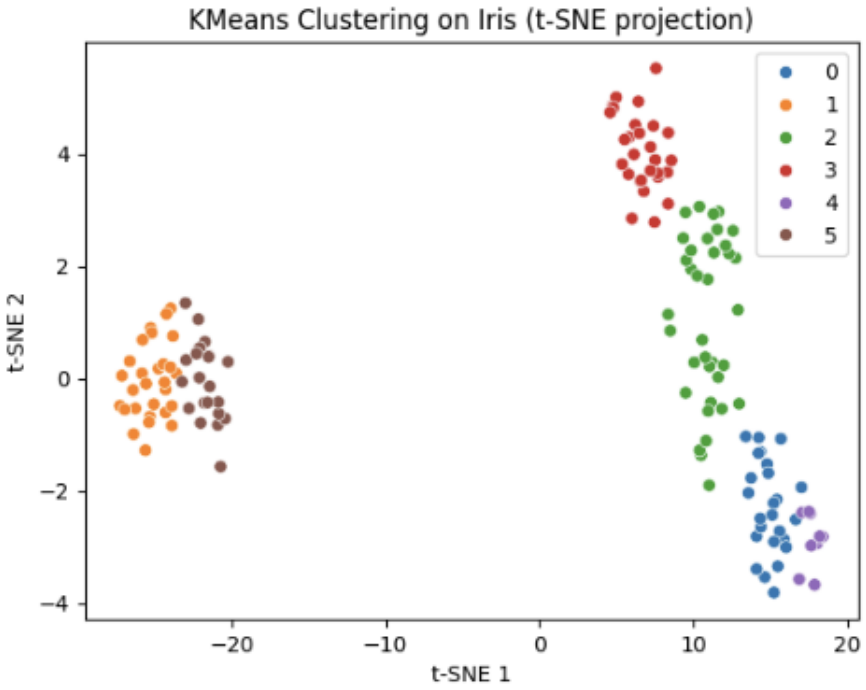

Figure 8: KMeans clustering on iris dataset with t-SNE projection

Dataset	Iris (4 features, 150 samples)
Linkage Method	t-SNE, with n_components=2 and random_state=42
Clustering Algorithm	KMeans (n_clusters=6)
Feature Scaling	No scaling applied in this case (can optionally apply StandardScaler)

Table 7: Clustering representation with Iris datset

4.3.3 Interpretation of Results

- Visual Cluster Formation: The t-SNE projection reveals distinct clusters in the 2D space. Several compact groupings are observed, aligning with the KMeans output of six clusters.
- Over-segmentation Insight:
  - The Iris dataset has a true label count of three, but we used six clusters intentionally to observe KMeans' behavior in over-segmentation scenarios.
  - As seen in the figure, some true classes (like Setosa) may be split into multiple clusters or merged incorrectly, a common occurrence when the number of clusters is not optimal.
- Separation Quality:
  - Some clusters are well-separated, while others exhibit overlapping regions, suggesting lower intra-cluster cohesion and inter-cluster separation.
- Embedding Limitations:

– While t-SNE is effective in revealing structure, the 2D projection may distort global relationships. It should be interpreted primarily for local neighborhood consistency rather than exact distances.

4.3.4 Utility in Evaluation

- Qualitative Assessment: This plot serves as a qualitative diagnostic tool to understand the clustering tendency and how well the algorithm aligns with intuitive structure.
- Class Drift Detection: Over-clustering helps in visualizing potential subgroupings or fine-grained patterns within broader class definitions.
- Supporting Metrics: This visualization complements quantitative measures such as Silhouette Score, ARI, and NMI to form a holistic assessment of clustering quality

4.4 Domain Applicability

Different algorithms show varying suitability across application domains:

- Medical/Healthcare: Methods providing uncertainty quantification (AutoClass, COBWEB) and constraint integration (COP-KMeans) are preferred for regulatory compliance and expert knowledge integration.
- Business/Marketing: Simple, interpretable methods (k-means, k-medoids) are favored for customer segmentation where stakeholders need clear, actionable insights.
- Scientific Discovery: Rule-based methods (DIVCLUS-T) and visualization approaches (SOM-I) support hypothesis generation and pattern exploration.
- Text Mining: Categorical-specific methods (CLOPE, ROCK) and post-hoc explanations (LIME-C) address the unique challenges of high-dimensional, sparse text data.

4.5 Applicability Guidelines

Based on our comparative analysis, we provide the following guidelines for selecting explainable clustering algorithms:

4.5.1 Algorithm Selection Framework

For Small to Medium Datasets (n $<$ 10,000):

- High interpretability priority: k-medoids, DIVCLUS-T
- Balanced performance-interpretability: k-means, COP-KMeans
- Exploratory analysis: SOM-I, UMAP+Clustering
- Categorical data: CLOPE, ROCK

For Large Datasets (n > 10,000):

- Scalable with interpretability: k-means, CLOPE
- Post-hoc explanation needs: LIME-C (with sampling)
- Visualization requirements: UMAP+Clustering
- Avoid: SHAP-based methods, ROCK, t-SNE approaches

For High-Dimensional Data (p> 100):

- Dimensionality reduction first: UMAP+Clustering, t-SNE+Clustering
- Feature selection integration: DIVCLUS-T
- Avoid: SHAP methods (without feature selection)

For Categorical Data:

- Primary choice: CLOPE
- Secondary options: ROCK, COBWEB
- Mixed data: DIVCLUS-T, AutoClass

5 Discussion and Future Directions

5.1 Key Findings

Our comprehensive analysis reveals several important insights about explainability in clustering algorithms:

Algorithmic Transparency vs. Explanation Quality: Intrinsically interpretable methods consistently provide high-quality explanations with perfect fidelity, but may be limited in their ability to handle complex data structures. The transparency of algorithms like k-means and hierarchical clustering makes them naturally suitable for explainable clustering applications, particularly in domains requiring regulatory compliance or stakeholder understanding.

Post-hoc Explanation Effectiveness: Modern post-hoc explanation methods, particularly XClus, MUSE, and ExCAPE, demonstrate significant improvements in explanation quality compared to earlier approaches. These methods successfully bridge the gap between high-performance clustering algorithms and explanation requirements, though they may introduce additional computational overhead.

Evaluation Metric Relationships: We observe complex relationships between traditional clustering metrics (Silhouette, ARI, NMI) and explainability metrics (coverage, fidelity, comprehensibility, stability). High-performing clustering algorithms do not automatically produce high-quality explanations, emphasizing the need for dedicated explainability assessment.

Domain-Specific Requirements: Different application domains exhibit distinct explainability requirements. Healthcare applications prioritize comprehensibility and stability, while financial services emphasize fidelity and regulatory compliance. Social sciences applications benefit from global explanations and hierarchical structures.

5.2 Current Limitations and Challenges

Explanation Standardization: The field lacks standardized evaluation frameworks for comparing explanation quality across different methods. Current explainability metrics (coverage, fidelity, comprehensibility, stability) are not universally defined or consistently applied, making systematic comparison challenging.

Scalability Concerns: Many advanced explainable clustering methods face scalability limitations when applied to large-scale datasets. While algorithms like HDBSCAN and spectral clustering provide rich explanation opportunities, their computational complexity may limit practical applicability.

User-Centric Evaluation: Current evaluation approaches often focus on algorithmic properties rather than user understanding and satisfaction. There is a significant gap between technical explanation quality metrics and actual user comprehension and trust.

Multi-modal Explanation Integration: Most current methods focus on single explanation modalities (visual, textual, or numerical). Integrating multiple explanation types to provide comprehensive understanding remains challenging.

5.3 Emerging Trends and Opportunities

Interactive Explanation Systems: The development of interactive explanation interfaces that allow users to explore clustering results at different levels of detail represents a significant opportunity. Systems like MUSE point toward more sophisticated user-centric explanation approaches.

Causal Explanation Integration: Incorporating causal reasoning into clustering explanations could provide deeper insights into why certain groupings emerge. This approach would move beyond correlation-based explanations to causal understanding.

Federated and Privacy-Preserving Explainable Clustering: With increasing privacy concerns, developing explainable clustering methods that preserve data privacy while providing meaningful explanations represents an important research direction.

Automated Explanation Generation: Advances in natural language processing and automated report generation could enable more sophisticated automatic explanation generation for clustering results, making explainable clustering more accessible to non-technical users.

6 Conclusion

This survey provides a comprehensive examination of explainability in clustering algorithms, covering both established methods and recent advances in the field. We have

presented a systematic taxonomy of explainable clustering approaches, categorizing methods into intrinsically interpretable algorithms, post-hoc explanation techniques, and hybrid approaches. Our analysis encompasses both traditional clustering algorithms and contemporary explainability methods, including XClus, MUSE, ExCAPE, and various adaptations of LIME and SHAP for clustering contexts.

The comprehensive evaluation framework we present considers both clustering performance metrics (Silhouette coefficient, ARI, NMI) and explainability-specific measures (coverage, fidelity, comprehensibility, stability). Through systematic analysis across major clustering algorithm families—including centroid-based methods, medoid-based approaches, hierarchical clustering, density-based methods, spectral clustering, and probabilistic models—we provide practical insights for selecting appropriate explainable clustering techniques.