Title

Unsupervised learning

Abstract

This article provides a comprehensive overview of unsupervised machine learning, highlighting its significance in discovering hidden patterns from unlabeled data. It discusses key motivations, including the prevalence of unlabeled datasets and labeling challenges, and describes essential tasks like clustering, dimensionality reduction, and anomaly detection. The article compares various unsupervised methods, reviews their advantages and limitations, and presents evaluation metrics critical for validating model effectiveness, aiming to equip readers with practical insights for selecting and applying appropriate unsupervised techniques across diverse real-world scenarios.

Keywords

Anomaly Detection, Clustering Algorithms, Dimensionality Reduction, Machine Learning, Model Evaluation, Unlabeled Data, Unsupervised Learning

Key points

- Motivation for Unsupervised Learning
- Tasks of Unsupervised Learning
- Limitations of Unsupervised Learning
- Clustering Algorithms
- Dimensionality Reduction Methods
- Evaluation of Unsupervised Learning

Body text

Machine learning is a program that learns from data to make predictions or decisions about unknown events without explicit instructions (Mitchell, 1997; Murphy, 2012). It is a highly interdisciplinary field building on many scientific domains, such as artificial intelligence, statistics, optimization theory, and information theory; it can also be applied to a variety of problems, including data mining, recognition systems, and recommendation engines (Alpaydin, 2020; Bishop, 2006).

Different from statistics whose goal is conducting inferences from samples, machine learning algorithms aim at recognizing generalizable predictive patterns (Ij, 2018). To be specific, statistical methods concentrate on explicitly verifying assumptions about the problem and refining the specified models or providing quantitative confidences about the model; machine learning methods focus on forecasting unseen outcomes, making minimum assumptions about the data-generating process, and not pursuing the interpretability of the result. Thus, statistical models are chosen based on our domain knowledge, while machine learning models are chosen relying on their empirical capabilities. One thing worth noticing is that, as many methods from statistics and machine learning can be applied to both prediction and inference, the boundary between machine learning and statistics is not very clear.

Generally, machine learning algorithms can be divided into two categories: supervised learning algorithms and unsupervised learning algorithms (Shalev-Shwartz & Ben-David, 2014). Supervised learning requires the algorithm to learn the mapping function between input variables and output variables given a labeled set of input-output pairs. After training, the model can be used to predict outputs for a new set of input data. With unsupervised learning algorithms, however, only the similarity between and among different input variables is evaluated for purposes of uncovering underlying patterns among the data. In this chapter, we will introduce the unsupervised learning algorithms.

Introduction to Unsupervised Learning

Unsupervised learning is a type of machine learning where algorithms analyze and interpret data without explicit labels or predefined outputs (Murphy, 2012). Instead of being guided by labeled examples, the model identifies patterns, structures, or relationships inherent in the data itself. Formally, given a dataset consisting of n unlabeled observations, unsupervised learning aims to uncover latent structures such as clusters, distributions, or embeddings that characterize the underlying data distribution without requiring explicit supervision.

Motivation of Developing Unsupervised Learning

The development of unsupervised learning methods is primarily driven by two key challenges in data analysis: the overwhelming presence of unlabeled data and the difficulty of obtaining labeled data in many real-world scenarios.

Ubiquity of Unlabeled Data and the Need for Structure Discovery. In numerous domains, vast amounts of data are collected without predefined labels or explicit categorization. Text corpora, image repositories, high-dimensional sensor readings, and financial transactions all represent large-scale datasets where the underlying structures are unknown. Traditional supervised learning methods rely on labeled datasets to learn predictive patterns, but in cases where labels are absent, machine learning models must independently uncover meaningful representations. Unsupervised learning addresses this challenge by enabling models to extract latent patterns, cluster similar instances, and reveal hidden relationships within raw data, making it a fundamental approach for exploratory analysis and knowledge discovery.

High Cost and Practical Challenges of Label Acquisition. Even in cases where labeling is conceptually possible, obtaining high-quality labeled data is often impractical due to the required expertise, labor-intensive nature, and sheer data volume. In fields such as medical diagnostics, labeling requires domain experts, making large-scale annotation both expensive and time-consuming. Similarly, in scientific research, newly emerging phenomena may not yet have well-defined labels, preventing the use of standard supervised approaches. In anomaly detection, rare events (e.g., fraudulent transactions, system failures) occur infrequently, leading to severe data imbalance and making labeled datasets sparse or unreliable. Unsupervised learning provides a viable solution by enabling data-driven analysis without requiring human-labeled annotations, allowing for adaptive pattern recognition in scenarios where labels are scarce or unavailable.

By addressing these challenges, unsupervised learning methods play a crucial role in automating data exploration, reducing reliance on labeled datasets, and enhancing knowledge discovery across diverse domains.

Tasks of Unsupervised Learning

Unsupervised learning encompasses a variety of tasks aimed at extracting meaningful insights from unlabeled data. These tasks serve as the foundation for numerous applications across disciplines, enabling knowledge discovery, pattern recognition, and data-driven decision-making. The three most common and impactful tasks in unsupervised learning are clustering, dimensionality reduction, and anomaly detection.

Clustering: Identifying Latent Group Structures. Clustering techniques group similar data points based on shared characteristics, uncovering hidden structures within datasets. By partitioning data without predefined categories, clustering enables the identification of meaningful subgroups, making it widely applicable across diverse fields. In finance, for instance, clustering is widely used to segment investors according to their trading behaviors, risk tolerance, or investment strategies, allowing for more personalized financial planning and targeted recommendations (e.g., Thompson et al., 2021). Similarly, in psychology, clustering aids in identifying subtypes of mental health conditions by analyzing symptom patterns, offering insights into previously unrecognized diagnostic categories and improving treatment personalization (e.g., Bolin et al., 2014). In the field of education, clustering techniques help uncover distinct student engagement profiles based on learning behaviors, facilitating the design of tailored instructional interventions to enhance academic performance (e.g., Liu, 2022). Beyond these domains, clustering is also instrumental in social sciences, particularly in social network analysis, where it enables the discovery of hidden community structures and the identification of key influencers within populations, shedding light on underlying behavioral and relational dynamics (e.g., Burt, 2013). By autonomously detecting patterns and segmenting data without requiring predefined labels, clustering serves as a fundamental tool for knowledge discovery and data-driven decision-making in complex, high-dimensional environments.

Dimensionality Reduction: Simplifying High-Dimensional Data. High-dimensional datasets often contain redundant or irrelevant information that can obscure meaningful patterns. Dimensionality reduction techniques extract essential features while preserving the structure of the data, enhancing interpretability and computational efficiency. In economics, dimensionality reduction is used to condense vast economic indicators into more interpretable representations, facilitating the analysis of market dynamics and economic trends (e.g., Chhikara et al., 2022). Similarly, in psychology, it aids in visualizing intricate personality trait data, improving the interpretability of psychological assessments and enabling more effective behavioral analysis (e.g., Shchurenkova, 2017). In education, where online learning platforms generate extensive student interaction data, dimensionality reduction helps distill this information to uncover key engagement patterns and optimize learning strategies (e.g., Chellatamilan & Suresh, 2012). By refining large-scale data into more manageable and meaningful representations, this technique is particularly valuable in exploratory research, where predefined feature sets may not yet exist.

Anomaly Detection: Identifying Deviations from Normal Patterns. Anomaly detection focuses on recognizing rare or unusual patterns in data, making it an essential tool for fraud detection, security monitoring, and error identification. Given that anomalous events are rare and typically lack predefined labels, unsupervised learning is particularly well-suited for this task, enabling models to autonomously flag irregularities for further investigation. In finance, for instance, anomaly detection is widely used to flag fraudulent transactions and detect market irregularities, playing a crucial role in preventing financial crimes and ensuring regulatory compliance (e.g., Niu, 2019). Similarly, in psychology, it helps identify atypical behavioral patterns that may signal underlying mental health conditions, facilitating early diagnosis and timely intervention (e.g., Huysmans et al., 2018). In the field of education, anomaly detection is employed to uncover abnormal test-taking behaviors and potential instances of academic dishonesty, where obtaining labeled data for fraudulent activities is often impractical (e.g., Cizek, 2003). By allowing for the detection of hidden irregularities in complex, high-dimensional datasets, unsupervised anomaly detection provides a powerful tool for ensuring security, maintaining integrity, and facilitating data-driven decision-making across a wide range of fields.

Limitations of Unsupervised Learning

Despite its broad applicability and ability to uncover hidden structures in data, unsupervised learning faces several challenges and limitations that affect its reliability, interpretability, and practical implementation. Issues such as interpretability, result variability, evaluation difficulties, data sensitivity, and high-dimensional complexity can limit its effectiveness in certain applications. These limitations are particularly relevant in fields such as social sciences and data analysis, where understanding the meaning behind discovered patterns is as crucial as identifying them. To maximize the reliability of unsupervised learning, practitioners must employ rigorous preprocessing, robust validation techniques, and domain-specific knowledge to ensure meaningful and interpretable results.

Interpretability Challenges. One of the primary limitations of unsupervised learning is the difficulty in interpreting its results, especially in domains where theoretical grounding is essential. Clustering algorithms, for instance, can segment data into different groups, but whether these groups align with meaningful sociological or psychological constructs often requires additional validation. Similarly, dimensionality reduction techniques can effectively reduce data complexity, but the newly generated components may lack clear semantic meaning, making it challenging for researchers to draw substantive conclusions from the results.

Uncertainty and Variability in Results. The outcomes of unsupervised learning are highly dependent on the choice of algorithms, parameter settings, and data characteristics. Different clustering methods may produce varying results on the same dataset. Even within the same algorithm, changes in initialization conditions or hyperparameters can lead to different solutions, introducing an element of uncertainty. This variability necessitates careful cross-validation and robustness checks to ensure the stability and reliability of findings.

Lack of Ground Truth for Evaluation. Unlike supervised learning, which has clearly defined performance metrics such as accuracy and precision, evaluating unsupervised learning models is inherently challenging due to the absence of labeled data. Evaluation metrics can provide some guidance in clustering tasks, but they are often dataset-dependent and may not generalize across different applications. In many cases, researchers must rely on qualitative assessments, expert judgment, or external validation to determine the validity of their results, which adds a layer of subjectivity to the evaluation process.

Sensitivity to Data Quality. Unsupervised learning algorithms are highly sensitive to data quality, and issues such as noise, missing values, or outliers can significantly impact model performance. For instance, in clustering, outliers can distort the formation of meaningful groups, leading to unreliable insights. As a result, thorough data preprocessing, including cleaning, normalization, and outlier detection, is critical for obtaining reliable outcomes.

Challenges with High-Dimensional Data. Many real-world datasets, particularly in social sciences and behavioral research, contain a large number of variables, leading to the well-known "curse of dimensionality." In high-dimensional spaces, distance-based clustering methods often become less effective, as data points tend to appear uniformly distributed. While dimensionality reduction techniques can help mitigate this issue, they may also discard important contextual information. Striking a balance between reducing dimensionality and preserving meaningful features is a persistent challenge in unsupervised learning applications.

Clustering Algorithms

Clustering algorithms can be broadly categorized based on their underlying principles, including partition-based, hierarchical, density-based and model-based clustering methods, each suited to different data characteristics and application needs (Yin et al., 2024). Partition-based methods divide data into non-overlapping clusters by minimizing intra-cluster variance, making them efficient but sensitive to cluster shapes and noise. Hierarchical clustering builds a nested structure using agglomerative (bottom-up) or divisive (top-down) approaches, allowing flexible cluster selection but being computationally expensive. Density-based methods detect arbitrarily shaped clusters based on density, handling noise well but struggling with varying densities. Model-based clustering assumes data follows probabilistic distributions or structured models, enabling flexible clustering but requiring careful parameter tuning. The following sections will provide a detailed exploration of these methods, highlighting their strengths, limitations, and application contexts.

Partition-Based Clustering

Partition-based clustering is a widely used approach in unsupervised learning that divides a dataset into multiple non-overlapping clusters, where each data point is assigned to a single cluster. These methods aim to minimize intra-cluster variance while maximizing inter-cluster separation, making them particularly effective for structured datasets where clear groupings exist. Due to their computational efficiency and simplicity, partition-based clustering techniques are extensively used in various applications such as market segmentation, image compression, and anomaly detection.

One of the most fundamental partition-based clustering methods is K-Means, which partitions data into K clusters by iteratively refining cluster centroids (Ahmed & Islam, 2020). The algorithm begins by selecting K initial centroids, either randomly or using an improved initialization method like K-Means++ (Vardakas & Likas, 2024). Each data point is then assigned to the nearest centroid based on Euclidean distance. The centroids are updated as the mean of all points in each cluster, and this process repeats until centroids stabilize or a predefined number of iterations is reached. K-Means assumes that clusters are roughly spherical, equally sized, and well-separated, which may not hold in all datasets. Furthermore, the algorithm requires specifying K in advance, a challenge that is often addressed using methods like the Elbow Method or Silhouette Score to determine an optimal cluster count. Despite these limitations, K-Means remains a widely used technique due to its simplicity, speed, and effectiveness in various clustering tasks.

To address the limitations of K-Means, several variants have been developed. K-Medoids (Kaur et al., 2014), for instance, follows a similar approach but selects actual data points as cluster centers rather than using the mean. This makes K-Medoids more robust to outliers, as it minimizes the influence of extreme values that could distort centroid calculations. Another notable variation is Fuzzy C-Means (FCM; Izakian & Pedrycz, 2013), which extends K-Means by allowing data points to belong to multiple clusters with different membership probabilities rather than making hard assignments. In FCM, each point has a degree of belonging to each cluster, calculated using a fuzzifier parameter mm, which controls the level of cluster overlap. The closer a point is to a centroid, the higher its probability of belonging to that cluster. FCM is particularly useful in applications where boundaries between clusters are not well-defined, such as medical diagnostics, where symptoms may correspond to multiple disease categories, or image segmentation, where pixels can be associated with multiple regions.

Despite their advantages, partition-based clustering methods have inherent challenges. They perform poorly when clusters have irregular shapes, varying densities, or overlap significantly. Additionally, K-Means and FCM require the number of clusters to be predefined, which is not always straightforward in exploratory analysis. Computationally, these methods can be sensitive to initialization, leading to varying results if centroids are not properly chosen. While improvements such as K-Means++ mitigate these initialization issues, alternative clustering methods like density-based clustering or hierarchical clustering may be preferable when dealing with more complex data distributions.

Partition-based clustering remains a cornerstone of unsupervised learning, offering a balance between efficiency and interpretability. Whether applied in finance, healthcare, or image processing, these methods provide powerful tools for uncovering hidden structures in data, enabling better decision-making and insights in a wide range of domains.

Hierarchical Clustering

Hierarchical clustering is a classical unsupervised learning method that organizes data into a hierarchical tree structure, often represented as a dendrogram (Cohen et al., 2019). Unlike partition-based clustering methods, which require the number of clusters to be predefined, hierarchical clustering provides a nested clustering structure that can be explored at different levels of granularity. This flexibility makes it particularly useful in scenarios where the true number of clusters is unknown or when an interpretable clustering hierarchy is desired. By visualizing the clustering process as a tree, hierarchical clustering allows users to cut the dendrogram at different levels to extract different clustering results based on domain knowledge.

There are two primary approaches to hierarchical clustering: agglomerative (bottom-up) clustering and divisive (top-down) clustering (Shetty & Singh, 2021). In agglomerative clustering, each data point is initially treated as an individual cluster, and the algorithm iteratively merges the two most similar clusters until only a single cluster remains. This process can be visualized as constructing a tree from the bottom up, where the root represents the entire dataset, individual leaves represent single data points, and intermediate nodes represent merged clusters. The result is a hierarchy of nested clusters that provides multiple levels of data grouping. In contrast, divisive clustering takes the opposite approach by initially treating all data points as a single cluster and recursively splitting them into smaller subclusters until each data point belongs to its own group. Though less commonly used than agglomerative clustering, the divisive method is useful when global cluster structure needs to be identified first before refining individual subgroups.

A key factor in hierarchical clustering is the linkage criterion, which determines how the distance between clusters is measured. Single linkage defines cluster similarity based on the shortest distance between any two points in different clusters, often resulting in elongated, chain-like clusters. Complete linkage, in contrast, considers the maximum distance between points in different clusters, producing more compact and well-separated clusters. Average linkage computes the mean distance between all pairs of points across clusters, providing a balance between single and complete linkage. Ward’s method, a popular choice, minimizes the increase in intra-cluster variance when merging clusters, leading to well-balanced and interpretable results. The choice of linkage criterion can significantly impact the clustering outcome, and different linkage methods may yield different hierarchical structures.

Despite its advantages in revealing hierarchical relationships, hierarchical clustering has several limitations. Its computational complexity is high, making it inefficient for very large datasets. Unlike iterative clustering methods such as K-Means, hierarchical clustering does not allow points to be reassigned once they have been merged or split, which can sometimes lead to suboptimal clustering results. Additionally, the selection of the linkage criterion is critical, as different linkage strategies can produce vastly different cluster structures.

Nevertheless, hierarchical clustering remains a widely used technique in social network analysis, where it helps uncover community structures by detecting groups of closely connected individuals. In market segmentation, hierarchical clustering is used to identify customer groups with similar behaviors, enabling targeted marketing strategies. The ability to capture nested structures and provide an intuitive, visual representation of clustering relationships makes hierarchical clustering a valuable tool in exploratory data analysis and pattern recognition.

Density-Based Clustering

Density-based clustering is a powerful approach in unsupervised learning that groups data points based on regions of high density while distinguishing sparse areas as noise (Kriegel et al., 2011). Unlike partition-based methods such as K-Means, which assume clusters have a spherical shape, density-based clustering is well-suited for identifying clusters of arbitrary shapes and handling datasets with noise and outliers. This flexibility makes it particularly valuable in real-world applications where data distributions are complex and do not conform to simple geometric assumptions.

One of the most widely used density-based clustering algorithms is DBSCAN (Density-Based Spatial Clustering of Applications with Noise; Schubert et al., 2017), which defines clusters as contiguous high-density regions separated by lower-density areas. Instead of requiring a predefined number of clusters, DBSCAN relies on two user-defined parameters: the distance threshold ϵ, which determines the neighborhood around a point, and minPts, which specifies the minimum number of points required to form a dense region. The algorithm classifies points into three categories: core points, which belong to dense regions; border points, which are close to dense regions but do not meet the density requirement themselves; and noise points, which do not belong to any cluster. The clustering process begins by randomly selecting a data point as a potential core point. If it has at least minPts neighbors within the ϵ radius, a new cluster is formed. The algorithm then iteratively expands this cluster by including neighboring core points until all reachable dense regions have been assigned. If a selected point does not meet the density requirement, it is labeled as a border point or noise, and the process moves to the next unvisited point.

DBSCAN is particularly effective in scenarios where clusters have irregular shapes, such as in geospatial data analysis, where points naturally form non-convex groups, or in anomaly detection, where it can separate normal data points from outliers in financial transactions. However, the algorithm struggles with datasets where clusters have varying densities, as a single global ϵ may not be suitable for all cluster structures. To address this limitation, an extension of DBSCAN, ordering points to identify the clustering structure (OPTICS), dynamically adjusts ϵ, allowing it to detect clusters of different densities. By generating a reachability plot, OPTICS provides insights into the hierarchical structure of clusters, making it particularly useful when the number of clusters and their densities are unknown.

Another notable density-based method is Mean Shift, which iteratively shifts data points toward the nearest high-density region based on kernel density estimation (Carreira, 2015). Unlike DBSCAN and OPTICS, Mean Shift does not rely on pre-defined parameters like ϵ or minPts, making it adaptive to varying cluster densities. This method is especially effective for mode-seeking tasks, such as image segmentation, where it groups pixels based on intensity distribution, and object tracking, where it identifies distinct objects in motion.

Despite their advantages, density-based clustering methods face challenges when applied to high-dimensional data, where density estimation becomes less reliable due to the curse of dimensionality. Additionally, selecting optimal parameters, particularly ϵ and minPts, often requires domain expertise or cross-validation techniques. Nonetheless, when applied correctly, density-based clustering provides robust solutions for detecting complex cluster structures, handling noise, and identifying anomalies, making it an essential tool in exploratory data analysis and real-world machine learning applications.

Model-Based Clustering

Model-based clustering assumes that data is generated from an underlying probabilistic distribution or structured model, and clustering is achieved by estimating the parameters governing these distributions. Unlike distance-based methods such as K-Means, which rely on geometric proximity, model-based clustering provides a probabilistic framework that accounts for uncertainty in cluster assignments and can accommodate complex data structures.

A widely used approach in model-based clustering is the Gaussian Mixture Model (GMM; Reynolds, 2009), which represents the data as a mixture of multiple Gaussian distributions. Each cluster is modeled as a multivariate Gaussian distribution characterized by parameters such as mean, covariance matrix, and mixture weight. Since direct optimization of these parameters is challenging, the Expectation-Maximization (EM; Moon, 1996) algorithm is employed to iteratively refine them by maximizing the likelihood of the observed data. Unlike K-Means, which assigns points to clusters in a hard manner, GMM allows for soft cluster assignments, meaning each data point has a probability of belonging to multiple clusters. This flexibility makes GMM particularly useful for clustering problems where data points do not fit neatly into discrete categories but instead exist on a spectrum. However, the performance of GMM is highly dependent on the choice of the number of components, which can be determined using model selection techniques such as the Bayesian Information Criterion (BIC) or Akaike Information Criterion (AIC).

An extension of GMM, the Dirichlet Process Gaussian Mixture Model (DPGMM; Görür & Edward, 2010), addresses the limitation of needing to specify the number of clusters in advance. DPGMM is a non-parametric Bayesian approach that allows the number of clusters to be learned dynamically from the data rather than being pre-defined. This makes it particularly well-suited for applications where the number of natural groupings is unknown or difficult to estimate.

Another notable model-based clustering technique is the Self-Organizing Map (SOM; Vesanto & Alhoniemi, 2000), a neural network-based method that transforms high-dimensional input space into a structured, lower-dimensional representation. SOM consists of a grid-like structure of nodes, where each node represents a prototype vector, and similar data points are mapped to neighboring nodes, preserving topological relationships. Unlike traditional neural networks that employ error backpropagation, SOM uses competitive learning, where nodes compete to become the best matching unit (BMU) for each input data point. The BMU and its neighboring nodes are then updated iteratively to better represent the data distribution. Over time, clusters naturally emerge as distinct regions within the SOM grid. This method is particularly effective for high-dimensional data visualization, exploratory data analysis, and clustering tasks where an interpretable topological structure is beneficial, such as pattern recognition in financial risk modeling, cognitive trait clustering in psychology, and gene expression profiling in bioinformatics.

Despite their advantages, model-based clustering methods come with challenges. GMM can struggle with covariance estimation in high-dimensional spaces, often requiring regularization techniques to prevent overfitting. Similarly, SOM requires careful tuning of parameters such as grid size, learning rate, and neighborhood function, which can significantly influence clustering results. Additionally, feature selection is often an integral part of model-based clustering, as the relevance of features can impact the learned model parameters and the overall quality of clustering.

Comparison and Summary

Partition-based clustering assumes that clusters are well-separated groups where data points in the same cluster are closer to their respective cluster center than to other cluster centers. The core idea of this approach is to iteratively assign data points to the nearest center and update the centers accordingly. This method is particularly effective when clusters have spherical shapes and similar sizes, but struggles when dealing with non-convex clusters or imbalanced data distributions. A key limitation is the requirement to predefine the number of clusters, making it less flexible in exploratory analysis.

Hierarchical clustering shares a similar motivation with partition-based methods in that it groups nearby data points together. However, instead of defining clusters through a set of centers, it constructs a hierarchy of clusters based on pairwise distances. This process can be visualized as a dendrogram, allowing users to analyze clustering structures at different levels of granularity. Compared to partition-based methods, hierarchical clustering does not require predefining the number of clusters, making it more adaptive to complex data distributions. However, its reliance on linkage criteria and the absence of global optimization mechanisms may lead to unstable clustering results.

Density-based clustering, such as DBSCAN, extends the concept of distance similarity by incorporating density criteria. Unlike K-means and hierarchical clustering, which rely purely on distance metrics, DBSCAN assumes that clusters are high-density regions separated by low-density gaps. This allows it to identify arbitrarily shaped clusters and effectively handle noise points. However, its performance is sensitive to parameter selection, and it struggles when clusters have varying densities or when the density transition between clusters is smooth. While partition-based and hierarchical clustering treat clustering as a global partitioning problem, DBSCAN approaches it from a local density perspective, making it well-suited for applications with unevenly distributed data.

Model-based clustering, such as Gaussian Mixture Models (GMM), differs fundamentally from the other three approaches. Instead of relying on distance measures or density estimations, model-based methods assume that data is generated from an underlying probabilistic distribution. The goal is to find the best-fitting parameters that describe these distributions. Unlike K-means or DBSCAN, GMM allows for soft clustering, where each data point has a probability of belonging to multiple clusters rather than being assigned to a single cluster. This makes it more flexible in modeling complex data structures but also increases the computational cost and the need for appropriate distribution assumptions.

In summary, partition-based and hierarchical clustering rely on distance-based similarity, with the former optimizing cluster centers and the latter forming a nested hierarchy. Density-based clustering extends this idea by considering both distance and density, making it robust to irregular cluster shapes. Model-based clustering, in contrast, frames clustering as a statistical estimation problem, allowing for soft clustering and greater flexibility but requiring stronger assumptions about the data distribution. The choice of clustering method depends on the dataset's structure, the desired level of interpretability, and the specific application requirements.

Dimension Reduction

Dimensionality reduction techniques aim to simplify high-dimensional data while preserving essential patterns, improving computational efficiency, mitigating the curse of dimensionality, and enhancing interpretability (Sorzano et al., 2014). In unsupervised learning settings, these techniques can be broadly categorized into linear and nonlinear methods, each suited to different data structures and analytical needs (Huang et al., 2019). Linear dimensionality reduction methods transform data using algebraic operations like projection and decomposition, assuming that meaningful variations lie along linear subspaces. These methods offer efficiency and interpretability but struggle with complex, nonlinear relationships. In contrast, nonlinear dimensionality reduction methods capture intricate structures by preserving local or manifold-based relationships, making them well-suited for highly complex, curved, or non-Euclidean data. These methods uncover underlying patterns that linear methods might overlook but often involve higher computational costs and reduced interpretability. The following sections will provide a detailed exploration of these approaches, discussing their principles, strengths, limitations, and application contexts.

Linear Dimensionality Reduction

Linear Dimensionality Reduction methods simplify high-dimensional data by applying linear transformations to find lower-dimensional representations that capture the most significant variations. These methods assume that the essential structure of the data lies within a linear subspace, making them computationally efficient and interpretable. Principal Component Analysis (PCA; Abdi & Williams, 2010) is one of the most commonly used approaches, projecting data onto the directions of maximum variance to minimize information loss. Singular Value Decomposition (SVD; Abdi, 2007), a matrix factorization technique, is particularly useful for reducing dimensionality in sparse data, such as text-based representations. Factor Analysis (FA; Kline, 2014) seeks to model the data as a combination of latent factors, separating signal from noise. Independent Component Analysis (ICA; Naik & Kumar, 2011) goes a step further by decomposing the data into statistically independent components, often used in applications like blind source separation. These methods are particularly effective when data exhibits a predominantly linear structure, with applications ranging from financial risk analysis to feature extraction in machine learning pipelines.

Principal Component Analysis. PCA transforms high-dimensional data into a lower-dimensional space by identifying the directions of maximum variance. PCA assumes that the most important structure in the data lies along linear directions and that features are scaled appropriately. The algorithm begins by computing the covariance matrix of the data and extracting its eigenvectors and eigenvalues. These eigenvectors, known as principal components, represent the new coordinate axes, with the top components capturing the most significant variations in the data. The data is then projected onto these principal components, reducing dimensionality while retaining as much variance as possible.

Singular Value Decomposition. SVD is a matrix factorization technique that decomposes a data matrix into three component matrices, revealing its intrinsic structure. Given an $m \times n$ data matrix X, SVD expresses it as:

$$X = U\Sigma V^T,$$

where U and V are orthogonal matrices representing the left and right singular vectors, and Σ is a diagonal matrix containing singular values that indicate the importance of each component. Unlike PCA, which explicitly relies on computing the covariance matrix, SVD directly operates on the original data matrix, making it particularly effective for handling sparse and high-dimensional datasets. SVD is widely used in applications such as noise reduction, latent semantic analysis in natural language processing, and recommendation systems, where extracting latent structures from large-scale data is essential. By truncating the lower singular values and their corresponding vectors, SVD provides a compact, lower-dimensional representation while preserving essential information.

Factor Analysis. FA is a statistical technique used to uncover latent variables, or factors, that explain the observed correlations among features in high-dimensional data. Unlike PCA, which seeks directions of maximum variance, FA assumes that observed variables are influenced by a smaller number of underlying factors along with some noise. Mathematically, FA models the data matrix X as a linear combination of these latent factors plus an error term:

$$X = LF + \epsilon,$$

where L is the loading matrix representing the influence of each factor on the observed variables, F is the matrix of latent factors, and ϵ represents unique variances (noise). The goal of FA is to estimate L and F, revealing the hidden structure behind the data. FA is particularly useful in applications where the observed variables are expected to be driven by a small number of underlying dimensions, such as psychology (e.g., identifying latent personality traits), finance (e.g., modeling market forces affecting asset prices), and biomedical sciences (e.g., detecting underlying genetic factors from medical data). Compared to PCA, FA provides a more interpretable decomposition by explicitly modeling noise and separating common variance from unique variance, making it especially valuable for identifying meaningful latent constructs in complex datasets.

Independent Component Analysis. ICA is a computational technique designed to separate a multivariate signal into statistically independent, non-Gaussian components. Unlike PCA, which finds orthogonal directions of maximum variance, and FA, which models latent factors with some shared variance, ICA aims to extract components that are as statistically independent as possible. The key assumption in ICA is that observed data is a mixture of independent source signals, and the goal is to recover these hidden sources. Mathematically, ICA models the observed data X as:

$$X = AS$$

where A is the mixing matrix, and S is the source signal matrix, containing independent components that ICA seeks to recover. By estimating an unmixing matrix W, ICA attempts to compute an approximation of S, such that the recovered components are statistically independent. ICA is widely used in signal processing, particularly for applications like blind source separation. A classic example is the "cocktail party problem", where ICA can separate individual voices from a mixed audio recording. Additionally, ICA has applications in neuroscience, where it is used to analyze EEG or fMRI data by isolating independent neural activity patterns, and in image processing, where it helps in feature extraction and compression. Compared to PCA and FA, ICA is especially effective when the goal is to uncover independent underlying signals rather than merely reducing dimensionality based on variance.

While PCA, SVD, FA, and ICA all serve the purpose of dimensionality reduction, they differ in their underlying assumptions and applications. PCA focuses on capturing directions of maximum variance, making it effective for feature extraction and noise reduction in structured datasets. SVD generalizes matrix decomposition without relying on covariance computations, making it particularly useful for sparse and high-dimensional data such as text mining and recommendation systems. FA assumes that observed variables are influenced by a small number of latent factors, explicitly modeling noise and shared variance, which makes it valuable for uncovering hidden structures in psychological, financial, and biomedical research. ICA, in contrast, seeks to extract statistically independent components, making it ideal for signal separation tasks such as EEG analysis and blind source separation. The choice among these methods depends on the data's structure and the specific goals of the analysis, whether it is maximizing variance, factorizing a matrix, identifying latent relationships, or extracting independent signals.

Nonlinear Dimensionality Reduction

Nonlinear Dimensionality Reduction methods extend beyond linear transformations to capture complex, curved structures in high-dimensional data. These methods are particularly valuable when data lies on a nonlinear manifold, where linear techniques like PCA fail to preserve meaningful relationships. t-Distributed Stochastic Neighbor Embedding (t-SNE; Wattenberg et al., 2016) is a widely used approach that maps high-dimensional data into a lower-dimensional space by preserving local pairwise similarities, making it highly effective for data visualization. Uniform Manifold Approximation and Projection (UMAP; McInnes et al., 2018) builds on similar principles but offers faster computation and better global structure preservation, making it suitable for large-scale datasets. Locally Linear Embedding (LLE; Roweis & Saul, 2000), Multidimensional Scaling (MDS; Davison & Sireci, 2000) and Isometric Mapping (Isomap; Balasubramanian & Schwartz, 2002) maintain local or geodesic distances to uncover the underlying structure of non-Euclidean data. Additionally, deep learning-based methods like Autoencoders (Bank et al., 2023) leverage neural networks to learn compact, nonlinearly transformed representations, particularly for high-dimensional inputs like images and text. While nonlinear techniques excel at uncovering intricate data patterns, they often involve higher computational costs and may require careful parameter tuning.

t-Distributed Stochastic Neighbor Embedding. t-SNE is a nonlinear dimensionality reduction technique designed for visualizing high-dimensional data by preserving local similarities. Unlike PCA, which relies on linear projections, t-SNE models data relationships using probabilistic similarity distributions. The algorithm begins by computing pairwise similarities between high-dimensional data points using a Gaussian distribution. It then maps these relationships into a lower-dimensional space, typically 2D or 3D, by minimizing the Kullback-Leibler divergence between the high-dimensional and low-dimensional similarity distributions. To achieve this, t-SNE employs a Student's t-distribution with a heavy tail in the lower-dimensional space, preventing crowding and preserving meaningful local structures. A key parameter, perplexity, controls the balance between local and global structure preservation. While t-SNE is highly effective for exploratory data analysis, especially in clustering applications, it is computationally expensive and does not provide a straightforward mapping function for new data points. Additionally, results can be sensitive to parameter choices, requiring careful tuning to achieve optimal visual representations.

Uniform Manifold Approximation and Projection. UMAP is a nonlinear dimensionality reduction technique that constructs a low-dimensional representation of high-dimensional data while preserving both local and global structures. Unlike t-SNE, which focuses on preserving local similarities, UMAP is rooted in manifold learning and graph theory, making it computationally efficient and scalable to large datasets. The algorithm begins by constructing a weighted k-nearest neighbor (k-NN) graph in the high-dimensional space, capturing the local topology of the data. It then optimizes a lower-dimensional embedding by preserving these neighborhood relationships using a fuzzy topological structure. Unlike t-SNE, which models pairwise similarities with probability distributions, UMAP uses a cross-entropy loss function to optimize the embedding, making it faster and more memory-efficient. Additionally, UMAP provides a learned transformation, allowing it to generalize to new data points—a feature lacking in t-SNE. This makes UMAP particularly useful for applications such as clustering, feature extraction, and visualization in large-scale machine learning tasks. However, its performance is sensitive to hyperparameters like n_neighbors (controlling local vs. global structure retention) and min_dist (determining how closely points can cluster together), requiring careful tuning for optimal results.

Locally Linear Embedding. LLE is a nonlinear dimensionality reduction technique that preserves the local geometry of high-dimensional data by assuming that each data point and its nearest neighbors lie on a locally linear patch of a lower-dimensional manifold. Unlike t-SNE and UMAP, which rely on probabilistic approaches, LLE directly reconstructs each data point as a linear combination of its nearest neighbors, capturing intrinsic geometric relationships. The algorithm begins by identifying the k-nearest neighbors of each data point and computing optimal reconstruction weights that minimize the difference between the original point and its weighted sum of neighbors. These weights are then used to embed the data into a lower-dimensional space while preserving the local reconstruction structure. LLE is particularly useful for discovering low-dimensional manifolds in high-dimensional data, making it effective for applications such as image processing and speech recognition. However, it assumes that the manifold is well-sampled and locally linear, making it sensitive to noise and requiring careful tuning of the number of neighbors (k) to balance capturing local structure and avoiding overfitting. Additionally, LLE struggles with preserving global structure and is less effective when dealing with data that lies on complex, highly curved manifolds. Variants such as Modified LLE (MLLE; Zhang & Wang, 2006) and Hessian LLE (HLLE; Donoho, 2003) have been developed to improve stability and better handle challenging data distributions.

Multidimensional Scaling. MDS is a nonlinear dimensionality reduction technique that seeks to preserve the pairwise dissimilarities between data points when projecting them into a lower-dimensional space. Unlike PCA, which relies on variance maximization, MDS focuses on maintaining the relative distances between points based on a given dissimilarity metric, making it particularly useful when working with non-Euclidean distance measures such as correlation or geodesic distances. The algorithm starts by constructing a dissimilarity matrix, where each entry represents the distance between a pair of data points in the high-dimensional space. It then finds a lower-dimensional embedding that best preserves these distances by minimizing a stress function, which quantifies the difference between the original and embedded distances. Classical MDS achieves this by performing eigenvalue decomposition on the distance matrix, while non-metric MDS optimizes the embedding iteratively using gradient-based techniques.

MDS is widely used in fields such as psychology, bioinformatics, and marketing to visualize high-dimensional relationships in an interpretable way. However, it is computationally expensive due to the pairwise distance computation and eigenvalue decomposition, making it less scalable for very large datasets. Additionally, the quality of the embedding depends on the choice of distance metric, which must be carefully selected based on the underlying structure of the data. Variants such as metric MDS (which assumes metric distance functions) and non-metric MDS (which allows for ordinal distance preservation) provide flexibility for different types of datasets.

Isometric Mapping. Isomap is a nonlinear dimensionality reduction technique that extends classical MDS by incorporating geodesic distances to better capture the intrinsic geometry of a high-dimensional dataset. Unlike linear methods such as PCA, which assume that important variations lie along straight-line directions, Isomap preserves the underlying manifold structure by approximating the shortest paths between data points along the manifold rather than in the ambient high-dimensional space. The algorithm begins by constructing a weighted k-nearest neighbor (k-NN) graph, where each data point is connected to its closest neighbors based on Euclidean distance. It then estimates the geodesic distances between all pairs of points using Dijkstra's algorithm or Floyd-Warshall algorithm, effectively capturing the true distances along the curved manifold. Finally, it applies classical MDS on the resulting geodesic distance matrix to project the data into a lower-dimensional space while maintaining the global structure.

Isomap is particularly effective for datasets that lie on a smooth, low-dimensional manifold within a high-dimensional space, making it useful in applications such as image recognition, speech processing, and computational biology. However, it assumes that the manifold is well-sampled and that geodesic distances can be accurately estimated, making it sensitive to noise and outliers. Additionally, its computational cost grows with the dataset size due to the shortest-path computation and eigenvalue decomposition steps, making it less scalable than UMAP or t-SNE for very large datasets.

Autoencoder. Autoencoders are a deep learning-based approach to nonlinear dimensionality reduction, utilizing neural networks to learn compact representations of high-dimensional data. Unlike traditional methods such as PCA or t-SNE, which rely on algebraic transformations or probabilistic modeling, autoencoders use an encoder-decoder structure to map input data to a lower-dimensional latent space and then reconstruct it back to the original space. By minimizing reconstruction error, autoencoders extract essential features and patterns from the input data, making them effective for feature learning, anomaly detection, and generative modeling.

A standard autoencoder consists of two main components: an encoder, which transforms high-dimensional data into a lower-dimensional latent representation, and a decoder, which reconstructs the original input from this representation. The encoder applies a series of nonlinear transformations, compressing the data while preserving its most relevant features. The decoder then attempts to reconstruct the input as accurately as possible, ensuring that crucial information is retained in the compressed representation.

Several variations of autoencoders improve their effectiveness in different applications. Denoising autoencoders (DAE; Vincent, 2011) introduce noise to the input data during training, forcing the model to learn robust latent representations that can reconstruct clean inputs. Variational autoencoders (VAE; Doersch, 2016) incorporate probabilistic modeling, learning a structured latent space that facilitates generative modeling by ensuring smooth transitions between latent representations. Sparse autoencoders apply sparsity constraints to the latent representation, encouraging the model to identify the most critical features while ignoring redundant information. In image-related tasks, convolutional autoencoders (CAE; Masci et al., 2011) replace fully connected layers with convolutional layers, making them more suitable for spatial data by preserving local structures.

Autoencoders are widely used in applications such as dimensionality reduction, feature extraction, anomaly detection, and generative modeling. They are particularly effective for high-dimensional and complex datasets, such as images, text, and sensor data, where traditional methods struggle to capture meaningful structures. However, their performance heavily depends on the choice of network architecture, hyperparameter tuning, and sufficient training data. Additionally, unlike PCA, autoencoders do not guarantee an interpretable linear transformation, making it challenging to analyze the learned representations in certain cases. Despite these limitations, autoencoders remain a powerful tool in deep learning for capturing underlying patterns in high-dimensional data.

While t-SNE, UMAP, LLE, Isomap, MDS, and Autoencoders all reduce dimensionality nonlinearly, they differ in purpose and suitability. t-SNE is ideal for visualization, preserving local structure well but struggling with global patterns and large datasets due to high computational costs. UMAP is faster, preserves more global structure, and can be applied to new data, making it useful for both visualization and clustering. LLE works well for smooth, locally linear manifolds, such as image and shape data, but fails with noisy or disconnected structures. MDS is useful when pairwise distance preservation is key, such as in psychology and marketing, but is slow for large datasets. Isomap extends MDS by preserving geodesic distances, making it useful for curved manifolds like facial recognition data, though it is sensitive to outliers and computationally expensive. Autoencoders, leveraging deep learning, are effective for high-dimensional, complex data like images and text but require significant data and computational power. Choosing the right method depends on the data structure and goal: t-SNE and UMAP for visualization, LLE and Isomap for manifold learning, MDS for distance-based embeddings, and Autoencoders for deep feature extraction.

Evaluation Methods for Unsupervised Learning

Evaluating unsupervised learning models is challenging because there are no ground truth labels to compare against, unlike in supervised learning. Instead of measuring accuracy directly, evaluation methods in unsupervised learning focus on clustering quality and dimensionality reduction effectiveness. This section covers key evaluation techniques when no labels are available.

Evaluating Clustering Results

Clustering evaluation methods assess how well an algorithm groups similar data points while keeping different clusters distinct. Since there are no ground truth labels, evaluation relies on internal validation metrics, which measure cluster cohesion and separation based on data structure.

Internal Evaluation Metrics. These metrics measure how well clusters are formed based on intra-cluster compactness and inter-cluster separation.

- Silhouette Score (Shahapure & Nicholas, 2020): Measures how similar a data point is to its own cluster compared to the nearest other cluster. Values range from -1 to 1, with higher values indicating well-defined clusters.
- Davies-Bouldin Index (DBI; Davies & Bouldin, 1979): Assesses the compactness and separation of clusters by comparing intra-cluster distances to inter-cluster distances. Lower values indicate better clustering.
- Calinski-Harabasz Index (CH Score; Caliński & Harabasz, 1974): Evaluates the ratio of between-cluster variance to within-cluster variance, where higher values suggest better-defined clusters.
- Within-Cluster Sum of Squares (WCSS): Measures the spread of points within each cluster. A lower WCSS suggests that clusters are compact and well-separated.

Stability Analysis. To ensure clustering robustness, stability analysis tests whether the clusters remain consistent under different conditions.

- Clustering Consistency: Runs the same clustering algorithm multiple times with different initial conditions or small perturbations in the data to check if cluster structures remain stable.
- Cross-validation Stability: Splits the dataset into subsets and applies clustering separately on each subset. Similar cluster structures across subsets indicate robustness.

Evaluating Dimension Reduction Results

Dimensionality reduction techniques aim to simplify data while preserving meaningful structures. Evaluation methods assess information retention, structure preservation, and visualization quality.

Preservation of Information. These metrics evaluate how much information from the original data is retained after dimensionality reduction.

- Explained Variance Ratio: Used in PCA, it measures the proportion of total variance retained by the selected components. A higher variance ratio suggests less information loss.
- Reconstruction Error: Computes the difference between the original and reconstructed data after dimensionality reduction. Lower values indicate better preservation of information.

Preservation of Data Structure. These metrics assess whether important structures, such as nearest neighbors and global relationships, are maintained in the reduced space.

- Trustworthiness: Measures whether close points in high-dimensional space remain close in low-dimensional space. Higher values indicate better structure preservation.
- Continuity: Evaluates whether points that are distant in high-dimensional space remain distant in the reduced space.

Visualization Quality. When dimensionality reduction is used for visualization, interpretability is important.

- K-Nearest Neighbor (KNN) Preservation: Compares nearest neighbors before and after dimensionality reduction to check if local relationships are maintained.
- Spearman Rank Correlation: Measures whether the relative ordering of points is preserved after dimensionality reduction.

Evaluating unsupervised learning models without labels requires specialized metrics. Clustering quality is assessed using internal validation metrics like Silhouette Score and DBI, while dimensionality reduction effectiveness is measured through information retention and structure preservation metrics. The appropriate evaluation method depends on the model's purpose, data characteristics, and computational constraints.

Conclusion

In summary, unsupervised learning methods play an essential role in modern data science by addressing critical challenges associated with unlabeled datasets and reducing the reliance on costly labeled data. Key tasks—clustering, dimensionality reduction, and anomaly detection—enable meaningful pattern discovery, enhanced data interpretation, and practical applications across diverse fields such as finance, psychology, and education. Despite their benefits, these techniques present inherent limitations, including interpretability issues, result variability, evaluation difficulties, and sensitivity to data quality and dimensionality. Moving forward, ongoing research and advancements in computational approaches, evaluation metrics, and algorithmic robustness promise to further expand the effectiveness and applicability of unsupervised learning methodologies.

Reference

Abdi, H. (2007). Singular value decomposition (SVD) and generalized singular value decomposition. *Encyclopedia of measurement and statistics*, *907*(912), 44.

Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, *2*(4), 433-459.

Ahmed, M., Seraj, R., & Islam, S. M. S. (2020). The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, *9*(8), 1295.

Alpaydin, E. (2020). *Introduction to machine learning* (4th ed.). MIT Press.

Ankerst, M., Breunig, M. M., Kriegel, H. P., & Sander, J. (1999). OPTICS: Ordering points to identify the clustering structure. *ACM Sigmod record*, *28*(2), 49-60.

Balasubramanian, M., & Schwartz, E. L. (2002). The isomap algorithm and topological stability. *Science*, *295*(5552), 7-7.

Bank, D., Koenigstein, N., & Giryes, R. (2023). Autoencoders. *Machine learning for data science handbook: data mining and knowledge discovery handbook*, 353-374.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bolin, J. H., Edwards, J. M., Finch, W. H., & Cassady, J. C. (2014). Applications of cluster analysis to the creation of perfectionism profiles: a comparison of two clustering approaches. *Frontiers in psychology*, *5*, 343.

Burt, R. S., Kilduff, M., & Tasselli, S. (2013). Social network analysis: Foundations and frontiers on advantage. *Annual review of psychology*, *64*(1), 527-547.

Caliński, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, *3*(1), 1-27.

Carreira-Perpinán, M. A. (2015). A review of mean-shift algorithms for clustering. *arXiv preprint arXiv:1503.00687*.

Chellatamilan, T., & Suresh, R. M. (2012, January). Automatic classification of learning objects through dimensionality reduction and feature subset selections in an e-learning system. In *2012 IEEE International Conference on Technology Enhanced Education (ICTEE)* (pp. 1-6). IEEE.

Chhikara, P., Jain, N., Tekchandani, R., & Kumar, N. (2022). Data dimensionality reduction techniques for Industry 4.0: Research results, challenges, and future research directions. *Software: Practice and Experience*, *52*(3), 658-688.

Cizek, G. J. (2003). *Detecting and preventing classroom cheating: Promoting integrity in assessment*. Corwin Press.

Cohen-Addad, V., Kanade, V., Mallmann-Trenn, F., & Mathieu, C. (2019). Hierarchical clustering: Objective functions and algorithms. *Journal of the ACM (JACM)*, *66*(4), 1-42.

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2), 224-227.

Davison, M. L., & Sireci, S. G. (2000). Multidimensional scaling. In *Handbook of applied multivariate statistics and mathematical modeling* (pp. 323-352). Academic Press.

Doersch, C. (2016). Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.

Donoho, D. (2003). Hessian eigenmaps: new tools for nonlinear dimensionality reduction. *Proc. National Academy of Science*, *100*, 5591-5596.

Görür, D., & Edward Rasmussen, C. (2010). Dirichlet process gaussian mixture models: Choice of the base distribution. *Journal of Computer Science and Technology*, *25*(4), 653-664.

Huang, X., Wu, L., & Ye, Y. (2019). A review on dimensionality reduction techniques. *International Journal of Pattern Recognition and Artificial Intelligence*, *33*(10), 1950017.

Huysmans, D., Smets, E., De Raedt, W., Van Hoof, C., Bogaerts, K., Van Diest, I., & Helic, D. (2018). Unsupervised learning for mental stress detection-exploration of self-organizing maps. *Proc. of Biosignals 2018*, *4*, 26-35.

Ij, H. (2018). Statistics versus machine learning. *Nat Methods*, *15*(4), 233.

Izakian, H., & Pedrycz, W. (2013, June). Anomaly detection in time series data using a fuzzy c-means clustering. In *2013 Joint IFSA world congress and NAFIPS annual meeting (IFSA/NAFIPS)* (pp. 1513-1518). IEEE.

Kaur, N. K., Kaur, U., & Singh, D. (2014). K-Medoid clustering algorithm-a review. *Int. J. Comput. Appl. Technol*, *1*(1), 42-45.

Kline, P. (2014). *An easy guide to factor analysis*. Routledge.

Kriegel, H. P., Kröger, P., Sander, J., & Zimek, A. (2011). Density-based clustering. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, *1*(3), 231-240.

Liu, R. (2022). Data Analysis of Educational Evaluation Using K-Means Clustering Method. *Computational Intelligence and Neuroscience*, *2022*(1), 3762431.

Masci, J., Meier, U., Cireşan, D., & Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. In *Artificial neural networks and machine learning–ICANN 2011: 21st international conference on artificial neural networks, espoo, Finland, June 14-17, 2011, proceedings, part i 21* (pp. 52-59). Springer Berlin Heidelberg.

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.

Moon, T. K. (1996). The expectation-maximization algorithm. *IEEE Signal processing magazine*, *13*(6), 47-60.

Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. MIT Press.

Naik, G. R., & Kumar, D. K. (2011). An overview of independent component analysis and its applications. *Informatica*, *35*(1).

Niu, X., Wang, L., & Yang, X. (2019). A comparison study of credit card fraud detection: Supervised versus unsupervised. *arXiv preprint arXiv:1904.10604*.

Reynolds, D. A. (2009). Gaussian mixture models. *Encyclopedia of biometrics*, *741*(659-663), 3.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *science*, *290*(5500), 2323-2326.

Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems (TODS)*, *42*(3), 1-21.

Shahapure, K. R., & Nicholas, C. (2020, October). Cluster quality analysis using silhouette score. In *2020 IEEE 7th international conference on data science and advanced analytics (DSAA)* (pp. 747-748). IEEE.

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge University Press.

Shchurenkova, E. (2017). *Dimension reduction using Independent Component Analysis with an application in business psychology* (Doctoral dissertation, University of British Columbia).

Shetty, P., & Singh, S. (2021). Hierarchical clustering: a survey. *International Journal of Applied Research*, *7*(4), 178-181.

Sorzano, C. O. S., Vargas, J., & Montano, A. P. (2014). A survey of dimensionality reduction techniques. *arXiv preprint arXiv:1403.2877*.

Thompson, J. R., Feng, L., Reesor, R. M., & Grace, C. (2021). Know Your Clients' behaviours: a cluster analysis of financial transactions. *Journal of Risk and Financial Management*, *14*(2), 50.

Vardakas, G., & Likas, A. (2024). Global k-means++: an effective relaxation of the global k-means clustering algorithm. *Applied Intelligence*, *54*(19), 8876-8888.

Vesanto, J., & Alhoniemi, E. (2000). Clustering of the self-organizing map. *IEEE Transactions on neural networks*, *11*(3), 586-600.

Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural computation*, *23*(7), 1661-1674.

Wattenberg, M., Viégas, F., & Johnson, I. (2016). How to use t-SNE effectively. *Distill*, *1*(10), e2.

Yin, H., Aryani, A., Petrie, S., Nambissan, A., Astudillo, A., & Cao, S. (2024). A rapid review of clustering algorithms. *arXiv preprint arXiv:2401.07389*.

Zhang, Z., & Wang, J. (2006). MLLE: Modified locally linear embedding using multiple weights. *Advances in neural information processing systems*, *19*.