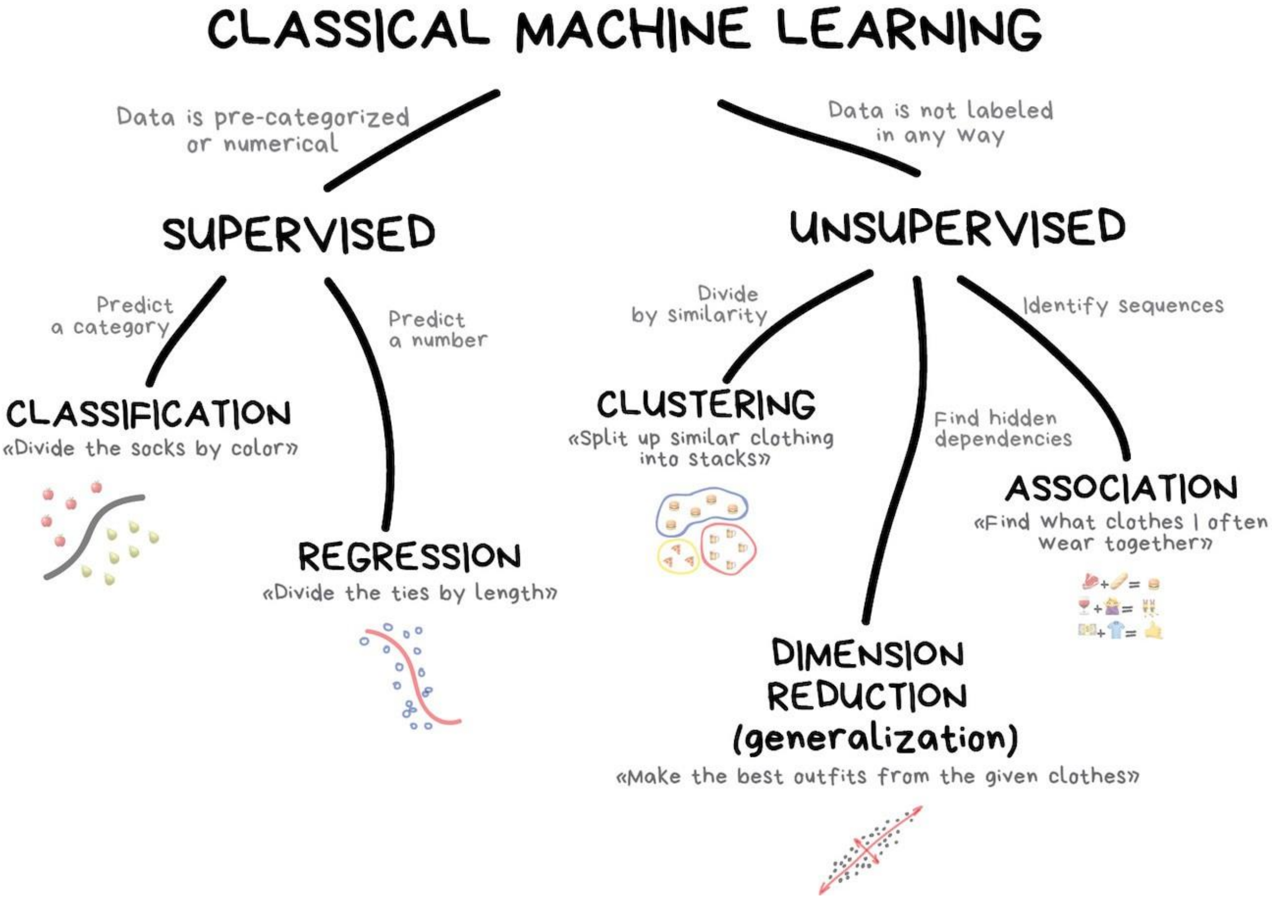
Unsupervised learning Clustering

Introduction to Unsupervised Learning

Objectives

- Understand unsupervised learning and its applications.
- Intro to clustering
- Hard and Soft Clustering
- K-means
- Hierarchical clustering



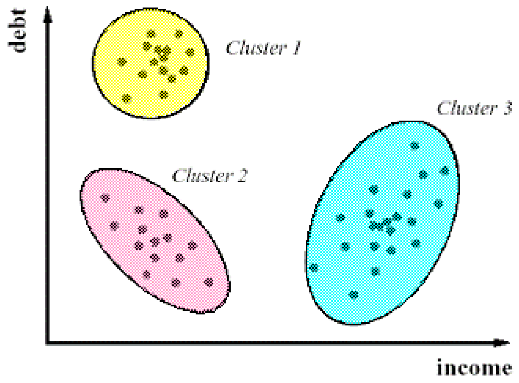
CLASSICAL MACHINE LEARNING
Data is pre-categorized or numerical
Data is not labeled in any way
SUPERVISED
UNSUPERVISED
Predict a category
Predict a number
CLASSIFICATION
«Divide the socks by color»
REGRESSION
«Divide the ties by length»
Divide by similarity
Identify sequences
Find hidden dependencies
CLUSTERING
«Split up similar clothing into stacks»
ASSOCIATION
«Find what clothes I often wear together»
DIMENSION REDUCTION (generalization)
«Make the best outfits from the given clothes»

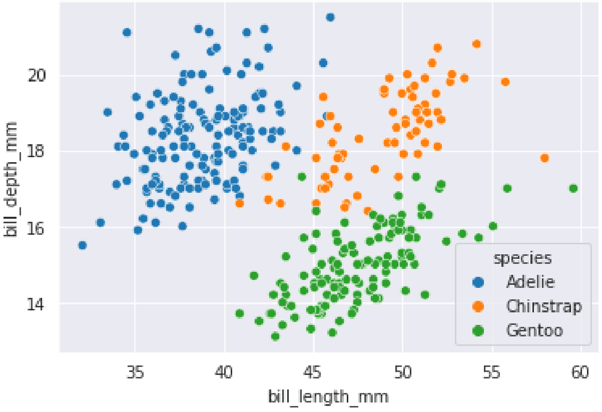
https://jovian.ai/adrian-g/sklearn-unsupervised-learning

What is Unsupervised Learning?

- **Definition:** Unsupervised learning is a type of machine learning where the algorithm is trained on data without explicitly labeled responses or target variables. Unlike supervised learning, which uses labeled data to predict outcomes, unsupervised learning seeks to identify patterns, relationships, or structures within the data without any guidance on what those patterns should be.
- **Key Characteristics:**
  - **No Labels or Outcomes:** The data used in unsupervised learning does not include labeled outcomes or target variables. The algorithm must discover patterns or groupings on its own.
  - **Data-Driven Insights:** It helps uncover hidden patterns or intrinsic structures within the data that are not immediately apparent.
  - **Exploratory Analysis:** Often used for exploratory data analysis, unsupervised learning can reveal new insights and guide further analysis or decision-making.

Clustering is a machine learning technique used to group similar data points together based on their features. The goal is to create clusters where data points within each cluster are more similar to each other than to those in other clusters.



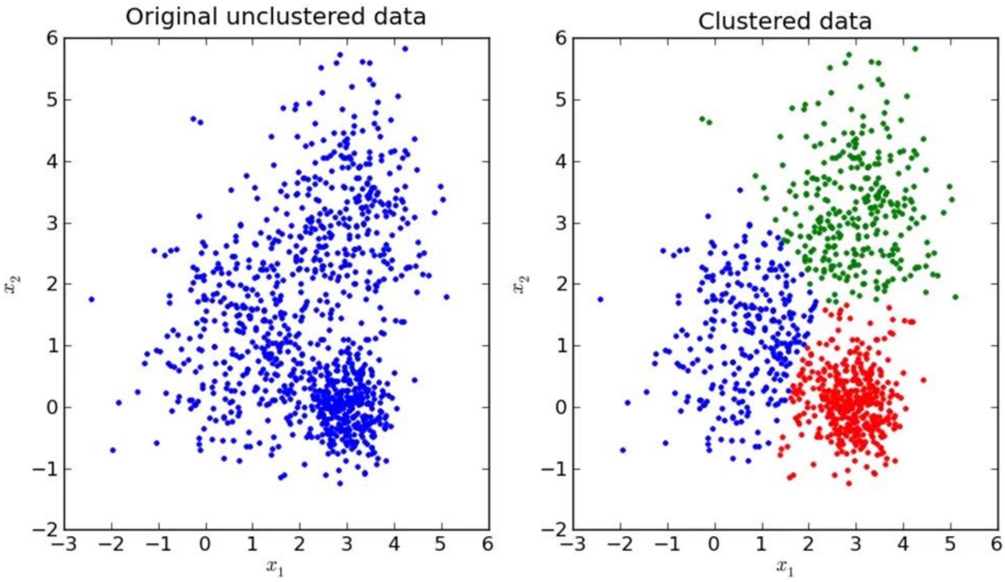


Example

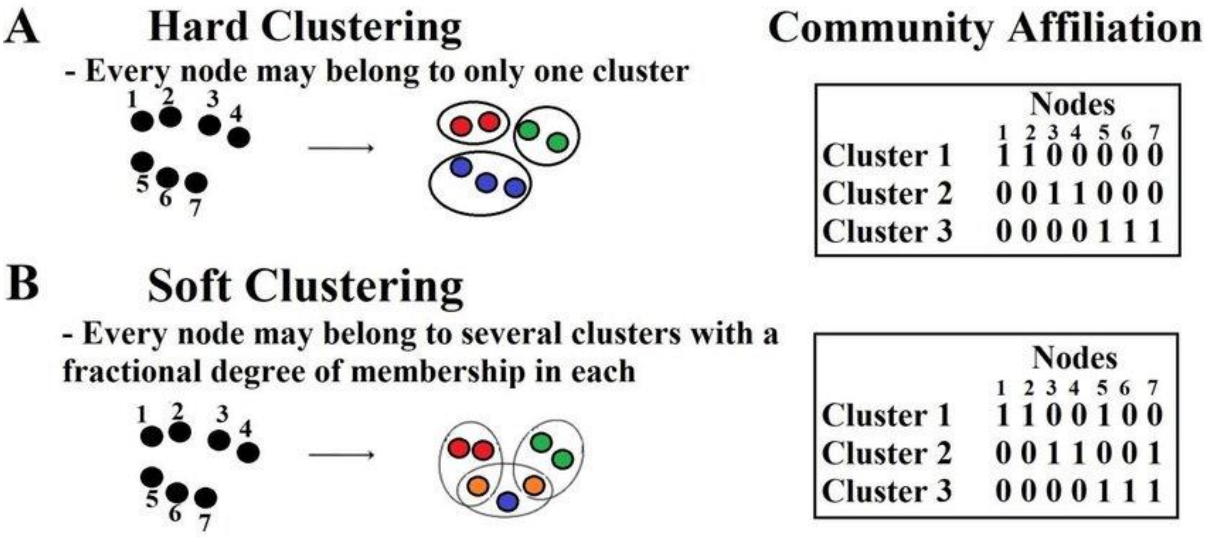
Imagine you have a dataset of various animals with features like size, weight, and habitat. Using clustering, you might group these animals into clusters such as:

1. **Mammals**: Animals like elephants, lions, and whales that share similar traits.
2. **Birds**: Animals like eagles, parrots, and sparrows with common features.
3. **Reptiles**: Animals like snakes, lizards, and turtles that are grouped based on their shared characteristics.

Each cluster represents a group of animals with similar attributes, helping you understand patterns and relationships in the data.



Hard Clustering vs Soft Clustering



Hard Clustering

- Every node may belong to only one cluster

Community Affiliation

	Nodes						
	1	2	3	4	5	6	7
Cluster 1	1	1	0	0	0	0	0
Cluster 2	0	0	1	1	0	0	0
Cluster 3	0	0	0	0	1	1	1

Soft Clustering

- Every node may belong to several clusters with a fractional degree of membership in each

	Nodes						
	1	2	3	4	5	6	7
Cluster 1	1	1	0	0	1	0	0
Cluster 2	0	0	1	1	0	0	1
Cluster 3	0	0	0	0	1	1	1

Hard Clustering assigns each data point to exactly one cluster. Once a data point is assigned to a cluster, it is considered part of that cluster with a definitive membership.

Characteristics:

- **Exclusive Membership**: Each data point belongs to a single cluster.
- **Clear Boundaries**: Clusters have clear boundaries, and data points are distinctly categorized.
- **Example Algorithm**: K-Means Clustering is a classic example of hard clustering.

Example:

Imagine you have a set of animals and want to classify them into distinct categories like "mammals," "birds," and "reptiles." In hard clustering, each animal would be assigned to one category only. An eagle would be classified strictly as a bird, not as a mammal or reptile.

Soft Clustering (also known as **Fuzzy Clustering**) allows data points to belong to multiple clusters with varying degrees of membership. Each data point has a probability or degree of membership for each cluster.

Characteristics:

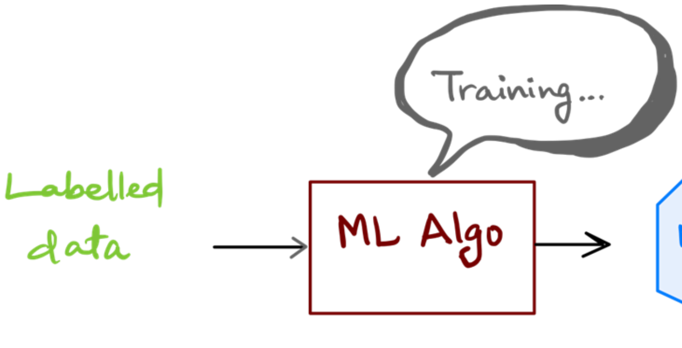
- **Probabilistic Membership**: Each data point can belong to multiple clusters, with a certain degree of membership (e.g., 70% to cluster A, 30% to cluster B).
- **Overlapping Clusters**: Clusters can overlap, and there are no strict boundaries.
- **Example Algorithm**: Fuzzy C-Means Clustering is a common example of soft clustering.

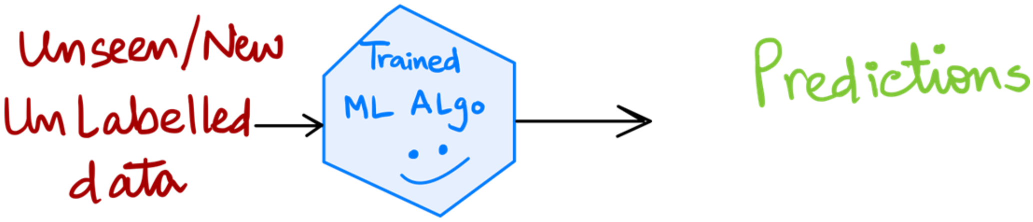
Example:

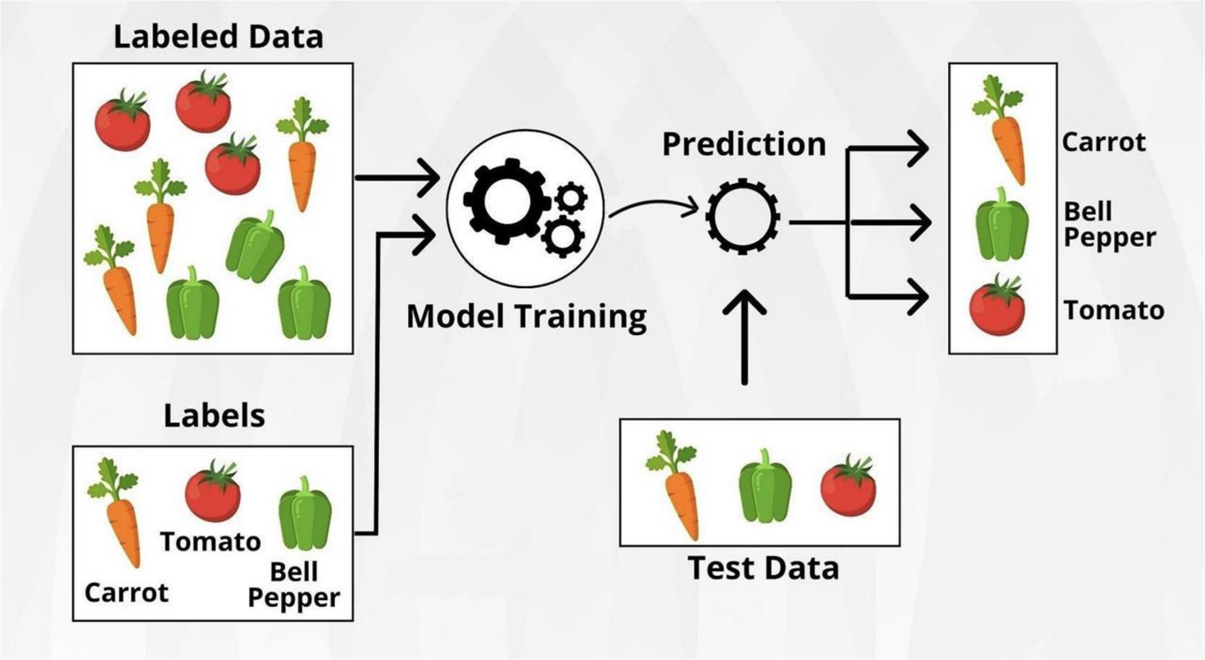
Using the same set of animals, a cat might have a high probability of being in the "mammals" cluster and a lower probability of being in the "reptiles" cluster if it has some traits that overlap with reptiles. Soft clustering allows for this overlap and provides a more nuanced classification.

Supervised vs. Unsupervised Learning

- **Supervised Learning:** Involves training a model on labelled data, where the outcomes are known. The goal is for the model to learn the mapping from inputs to outputs so it can predict the label for new, unseen data.







Key Characteristics:

- **Labeled Data:** The training data includes both inputs and corresponding correct outputs (labels).
- **Objective:** Learn the relationship between input features and the target variable to make predictions.
- **Training Process:** The model is trained by minimizing the error between the predicted outputs and the actual outputs.

Common Algorithms:

- **Classification:** Logistic Regression, Support Vector Machines (SVM), Decision Trees, Random Forest, Neural Networks.

- **Regression:** Linear Regression, Ridge Regression, Lasso Regression, Decision Trees, Neural Networks.

Examples:

- **Email Spam Detection:** The model learns from emails labeled as "spam" or "not spam" to classify new emails.
- **Credit Scoring:** The model predicts whether a loan applicant is likely to default based on labeled historical data.

Advantages:

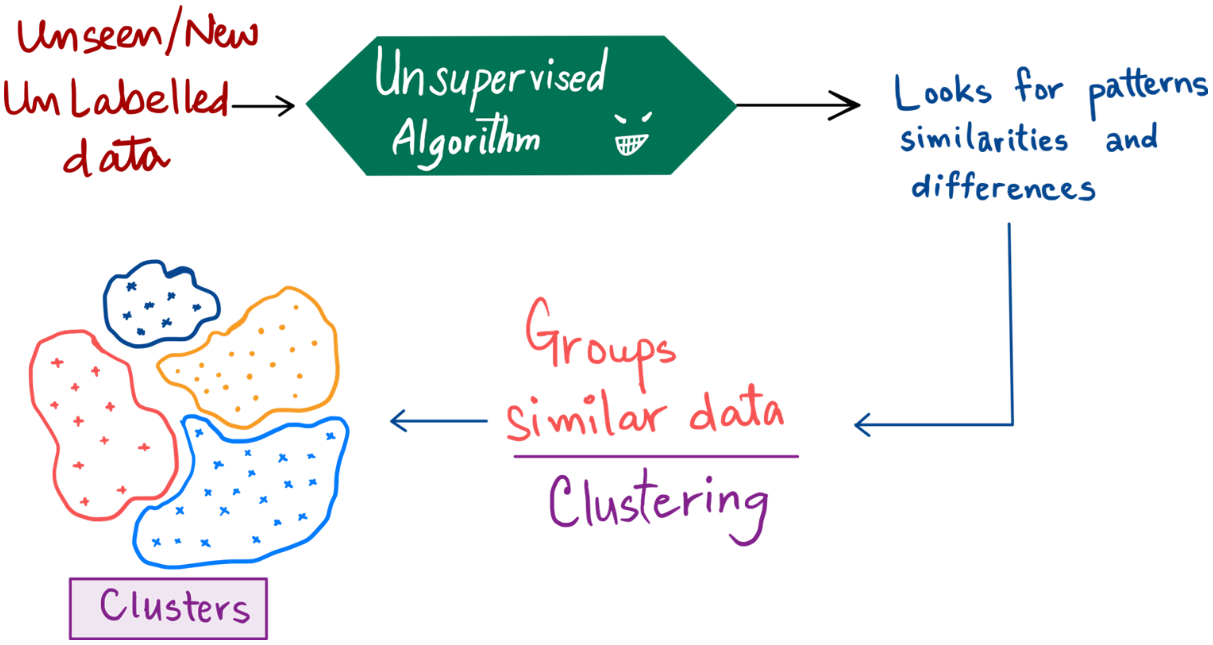
- Produces highly accurate models for prediction tasks.
- The relationship between input and output is clear and well-defined.

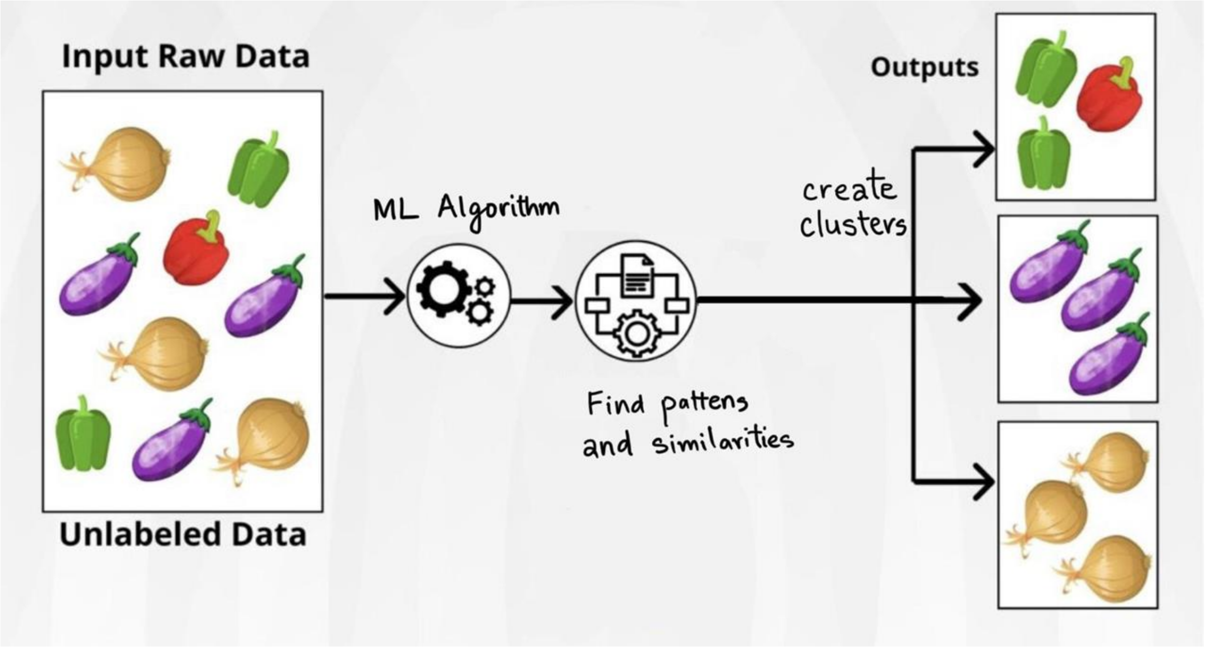
Disadvantages:

- Requires a large amount of labeled data, which can be costly and time-consuming to obtain.
- Performance is limited by the quality and quantity of the labeled data.

Unsupervised Learning

Definition: Unsupervised learning involves training a model on data without labeled responses or targets. The objective is to find hidden patterns, groupings, or structures within the data.





Key Characteristics:

- **Unlabeled Data:** The data used in training does not include labels or predefined outcomes.
- **Objective:** Discover hidden structures or patterns in the data.
- **Training Process:** The model identifies patterns and groupings without any guidance from labelled outputs.

Common Algorithms:

- **Clustering:** K-means, Hierarchical Clustering, DBSCAN.
- **Dimensionality Reduction:** Principal Component Analysis (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE).
- **Association Rules:** Apriori Algorithm, often used for market basket analysis.

Examples:

- **Customer Segmentation:** Group customers into segments based on purchasing behavior without prior knowledge of these groups.
- **Anomaly Detection:** Identify unusual transactions in financial systems that deviate from the norm, potentially indicating fraud.

Advantages:

- Can work with unlabeled data, which is often easier to collect.
- Useful for exploring data to find patterns that are not immediately obvious.

Disadvantages:

- The results can be less interpretable than supervised learning.
- There is no guarantee that the identified patterns will be meaningful or useful.

Applications of Unsupervised Learning

Unsupervised learning has a wide range of applications across various industries, where its ability to find patterns, group data, and detect anomalies provides valuable insights. Here are some key applications:

1. Customer Segmentation

- **Purpose:** Group customers into segments based on similar behaviors or characteristics.
- **Use Case:** In marketing, businesses use clustering techniques (e.g., K-means clustering) to segment customers based on their purchase history, browsing behavior, demographics, etc. This allows companies to tailor marketing strategies, personalize recommendations, and improve customer retention.

2. Anomaly Detection

- **Purpose:** Identify unusual patterns that do not conform to expected behavior.
- **Use Case:** In cybersecurity, unsupervised learning algorithms like DBSCAN or Isolation Forests are used to detect fraudulent activities, such as unusual login attempts or transactions, that deviate from normal patterns, helping in fraud detection and prevention.

3. Recommendation Systems

- **Purpose:** Suggest products, content, or services to users based on their behavior and preferences.
- **Use Case:** In e-commerce and streaming services, clustering and association rules (e.g., Apriori algorithm) are used to recommend items to users based on the behavior of similar users, enhancing user experience and increasing engagement.

4. Dimensionality Reduction for Data Visualization

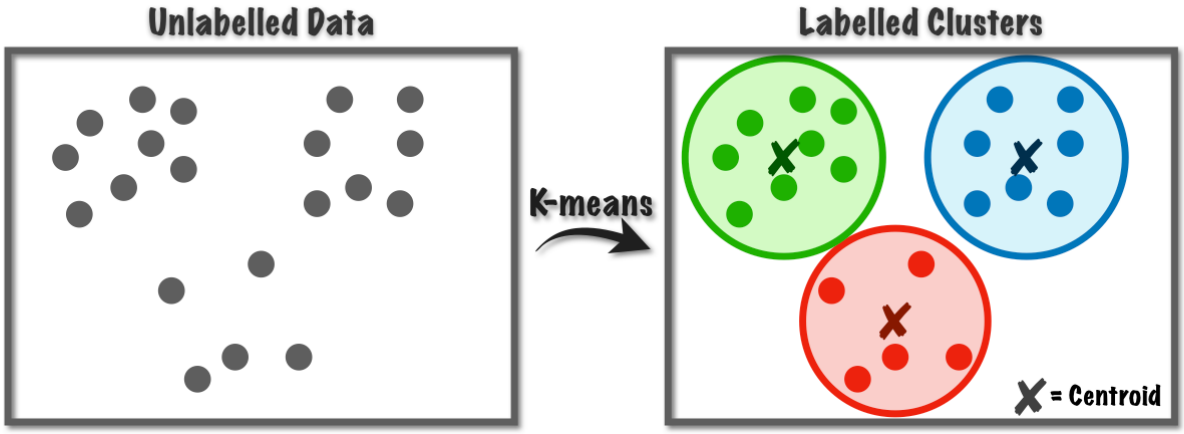
- **Purpose:** Reduce the complexity of data while retaining essential information, making it easier to visualize and understand.
- **Use Case:** Techniques like Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE) are used in data science to reduce the number of variables in datasets, which is especially useful for visualizing high-dimensional data in 2D or 3D spaces.

Clustering Algorithms Overview

- **Clustering:** A method of grouping objects so that objects in the same group are more similar to each other than to those in other groups.

K-means Clustering

- **Description:** It is a popular algorithm used in unsupervised machine learning to partition data into k distinct clusters. Each cluster is defined by a centroid, which is the average of all points in the cluster.



K-Means Clustering is a method used to group similar items together. Think of it as sorting objects into different boxes based on their similarities.

- **Strengths:** Simple, efficient.
- **Limitations:** Requires K to be specified, sensitive to initial placement of centroids.

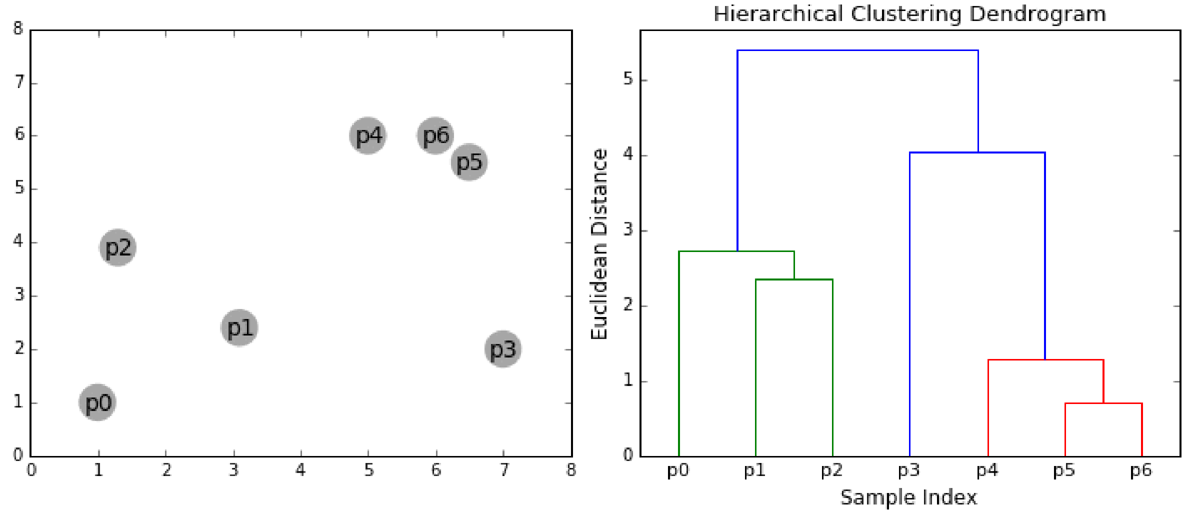
Steps of K-Means Clustering

1. **Choose the Number of Clusters (kkk)**: Decide how many clusters you want to divide your data into. This number **k** is specified by you.
2. **Initialize Centroids**: Randomly select k data points from your dataset to serve as the initial **centroids** (the **central points of each cluster**).
3. **Assign Data Points to Clusters**: Each data point is assigned to the cluster whose centroid is closest to it. This is typically done using Euclidean distance, but other distance metrics can be used.
4. **Update Centroids**: Recalculate the centroids of each cluster by taking the average of all data points assigned to that cluster. This new centroid represents the mean position of all points in the cluster.
5. **Repeat**: Repeat the assignment and update steps until the centroids no longer change significantly or converge (i.e., the assignments of data points to clusters stabilize).

6. **Convergence**: The algorithm stops when the centroids no longer change or the changes are minimal, indicating that the clusters are stable.

Hierarchical Clustering

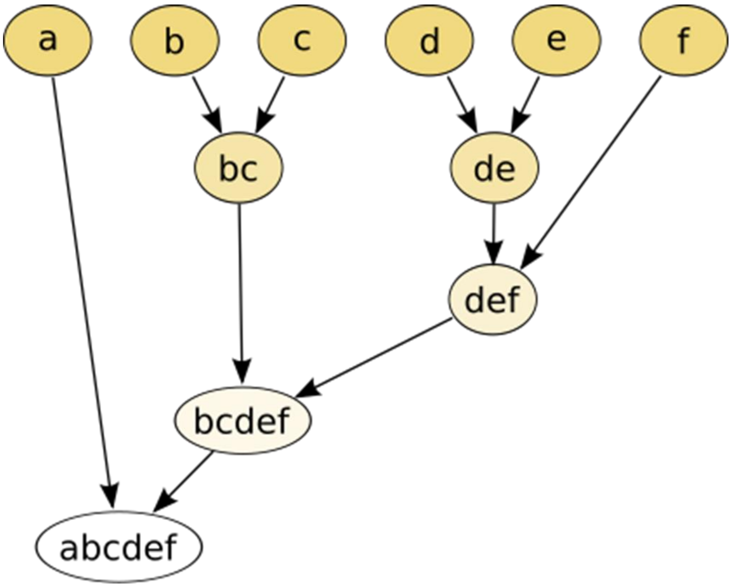
- **Description:** Builds a hierarchy of clusters with agglomerative (bottom-up) and divisive (top-down) methods. Unlike K-Means, which requires you to specify the number of clusters in advance, hierarchical clustering does not. It can produce a tree-like structure called a **dendrogram** to visualize the arrangement of clusters.



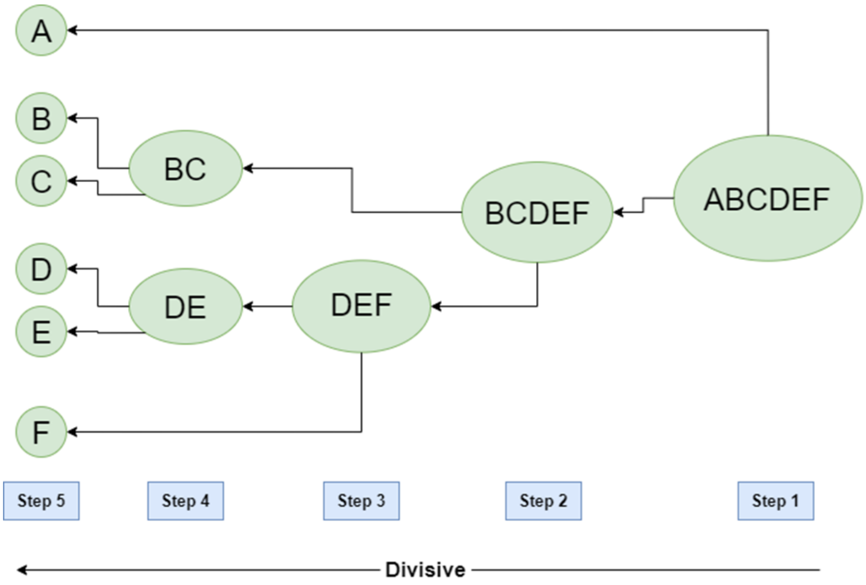
https://dashee87.github.io/images/hierarch.gif

Two Main Types

1. **Agglomerative Hierarchical Clustering** (Bottom-Up Approach):
   - **Start**: Begin with each data point as its own cluster.
   - **Merge**: Repeatedly merge the two closest clusters based on a distance metric until all points are in a single cluster or a stopping criterion is met.
   - **Result**: A dendrogram is created showing how clusters are merged at each step.



2. **Divisive Hierarchical Clustering** (Top-Down Approach):
    - **Start**: Begin with all data points in a single cluster.
    - **Split**: Repeatedly split the cluster into smaller clusters until each data point is in its own cluster or a stopping criterion is met.
    - **Result**: A dendrogram is created showing how clusters are split at each step.



How It Works

1. **Calculate Distances**: Compute the distances between all pairs of data points or clusters using a distance metric (e.g., Euclidean distance).
2. **Create a Dendrogram**:
    - **Agglomerative**: Create a tree by initially considering each data point as a cluster and then successively merging the closest clusters based on distance.

    - **Divisive**: Create a tree by starting with all points in one cluster and iteratively splitting the cluster into smaller clusters.
3. **Cut the Dendrogram**: To determine the final clusters, you cut the dendrogram at a certain level. The clusters at this level represent the final grouping of data points.

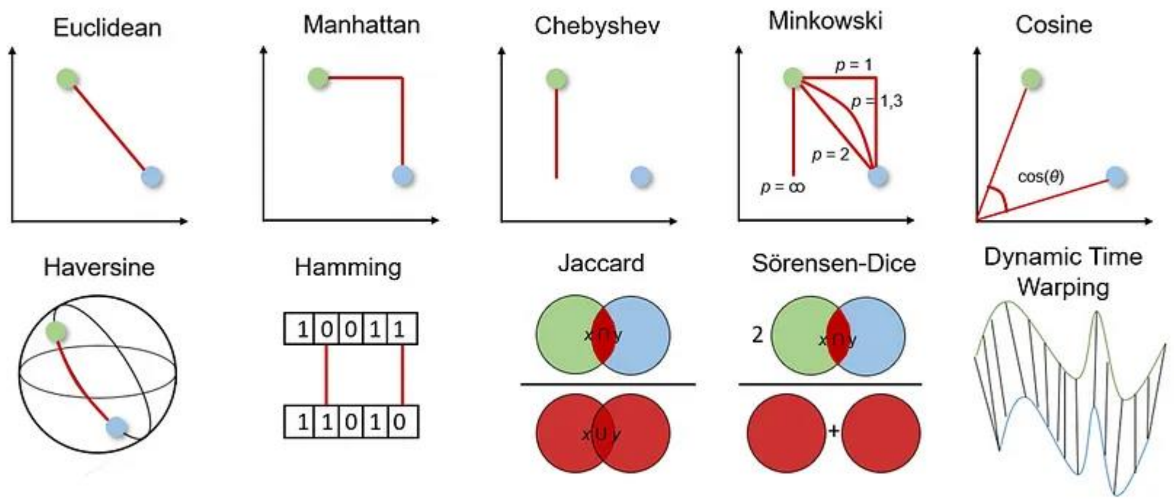
Example

Imagine you have a dataset of animals and you want to group them into a hierarchy based on their similarities:

1. **Start**: Each animal is its own cluster (e.g., lion, tiger, bear, etc.).
2. **Merge**: Find the closest clusters (e.g., lions and tigers) and merge them.
3. **Repeat**: Continue merging clusters (e.g., big cats with bears) based on proximity until all animals are grouped into one cluster.
4. **Dendrogram**: The result is a tree structure showing how animals are grouped together.

- **Strengths:** No need to specify clusters. Produces visual representation of cluster relationships.
- **Limitations:** Computationally expensive for large datasets. The choice of distance metric and linkage criteria can affect results.

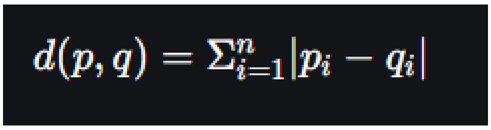
Common Distance Measures



The **Euclidean distance** is the most widely used distance measure in clustering. It calculates the straight-line distance between two points in n-dimensional space. The formula for Euclidean distance is:

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

The **Manhattan distance** is also called the Taxicab or City-Block distance as the distance between two real-valued vectors is calculated as if one could only move at right angles. This distance measure is often used for discrete and binary attributes to get a realistic path.



$$d(p, q) = \Sigma_{i=1}^{n} |p_i - q_i|$$

DBSCAN Clustering

- **Description:** Clusters data based on density, identifying core, reachable points, and noise.
- **Advantages:** Can find arbitrarily shaped clusters, robust to noise.

Dimensionality Reduction

- **Description:** A technique used to reduce the number of features (or dimensions) in a dataset while retaining as much information as possible. This is useful for simplifying models, improving computational efficiency, and visualizing high-dimensional data.
- **Applications:**
  - **Simplify Models**: Fewer features mean simpler models that are easier to interpret and faster to train.
  - **Reduce Overfitting**: Fewer dimensions can help reduce the risk of overfitting by removing noise and irrelevant features.
  - **Improve Performance**: Reducing dimensions can speed up training and inference times.
  - **Visualization**: High-dimensional data can be projected into 2D or 3D for visualization, making it easier to understand patterns and relationships.

Common Techniques

1. **Principal Component Analysis (PCA)**
   - **Description**: PCA is a technique that transforms the data into a new coordinate system where the greatest variance (information) is captured in the first few dimensions.
   - **How It Works**:
     - Compute the covariance matrix of the data. (Covariance - how much two random variables gets change together)

        - Calculate the eigenvectors (principal components) and eigenvalues.
        - Project the data onto the eigenvectors corresponding to the largest eigenvalues.
    - **Use Case**: Reducing dimensions while preserving the most significant features.
2. **t-Distributed Stochastic Neighbor Embedding (t-SNE)**
    - **Description**: t-SNE is a technique for visualizing high-dimensional data by reducing it to 2 or 3 dimensions while preserving the local structure.
    - **How It Works**:
        - Compute pairwise similarities between data points in high-dimensional space.
        - Map these similarities to a lower-dimensional space using probabilistic techniques.
    - **Use Case**: Visualizing clusters and relationships in high-dimensional data.

Strengths:

- Simplifies data and models.
- Improves performance and interpretability.
- Useful for visualization.

Limitations:

- Can lose some information in the reduction process.
- Some methods, like t-SNE, may not preserve global structure.
- Choosing the right technique and number of dimensions can be challenging.

Evaluation of Clustering Models

- **Metrics:** Silhouette Score (measures similarity within clusters), Elbow Method (finds optimal number of clusters).

Advanced Techniques: Gaussian Mixture Models (GMMs)

- **Description:** Probabilistic models assuming data points are from a mixture of Gaussian distributions.
- **Difference from K-means:** Considers the probability of belonging to each cluster.

Practical Coding Example

Summary and Key Takeaways

Reference: Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
Reference: Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
Reference: Aggarwal, C. C., & Zhai, C. (2012). *Mining Text Data*. Springer.
Reference: Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning*. Springer.
Reference: Jain, A. K. (2010). *Data clustering: 50 years beyond K-means*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(8), 645-656. Link
Reference: Murtagh, F., & Contreras, P. (2012). *Algorithms for hierarchical clustering: An overview*. Wiley Encyclopedia of Operations Research and Management Science. Link
Reference: Ester, M., et al. (1996). *A density-based algorithm for discovering clusters in large spatial databases with noise*. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD), 226-231. Link
Reference: van der Maaten, L., & Hinton, G. (2008). *Visualizing data using t-SNE*. Journal of Machine Learning Research, 9, 2579-2605. Link
Reference: Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer.
Reference: Agrawal, R., et al. (1993). *Mining association rules between sets of items in large databases*. Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, 207-216. Link
Reference: Rousseeuw, P. J. (1987). *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis*. Journal of Computational and Applied Mathematics, 20, 53-65. Link
Reference: Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
Reference: Kohonen, T. (2001). *Self-Organizing Maps*. Springer.

Reference: Agrawal, R., & Srikant, R. (1994). *Fast algorithms for mining association rules*. Proceedings of the 20th International Conference on Very Large Data Bases (VLDB), 487-499. Link

Reference: Tan, P.-N., Steinbach, M., & Kumar, V. (2019). *Introduction to Data Mining* (3rd ed.). Pearson. Link

Reference: Rousseeuw, P. J. (1987). *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis*. Journal of Computational and Applied Mathematics, 20, 53-65. Link

Reference: Davies, D. L., & Bouldin, D. W. (1979). *A cluster separation measure*. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-1(2), 224-227. Link

Reference: Lloyd, S. (1982). *Least squares quantization in PCM*. IEEE Transactions on Information Theory, 28(2), 129-137. Link

Reference: Rand, W. M. (1971). *Objective criteria for the evaluation of clustering methods*. Journal of the American Statistical Association, 66(336), 846-850. Link

Reference: Vincent, D., & Dubes, R. (1975). *Normalized mutual information as a measure of association*. Journal of the American Statistical Association, 70(350), 445-455. Link

Reference: Kang, H., & Hong, J. (2021). *Dimensionality reduction techniques for clustering: A comparative review*. IEEE Access, 9, 114034-114048. Link

Reference: Cheng, C., & Li, L. (2006). *Domain knowledge-based clustering*. Proceedings of the 2006 SIAM International Conference on Data Mining. Link