11.19.24

Clustering

Data 624 Presentation - Week 12

Presented by:

Will

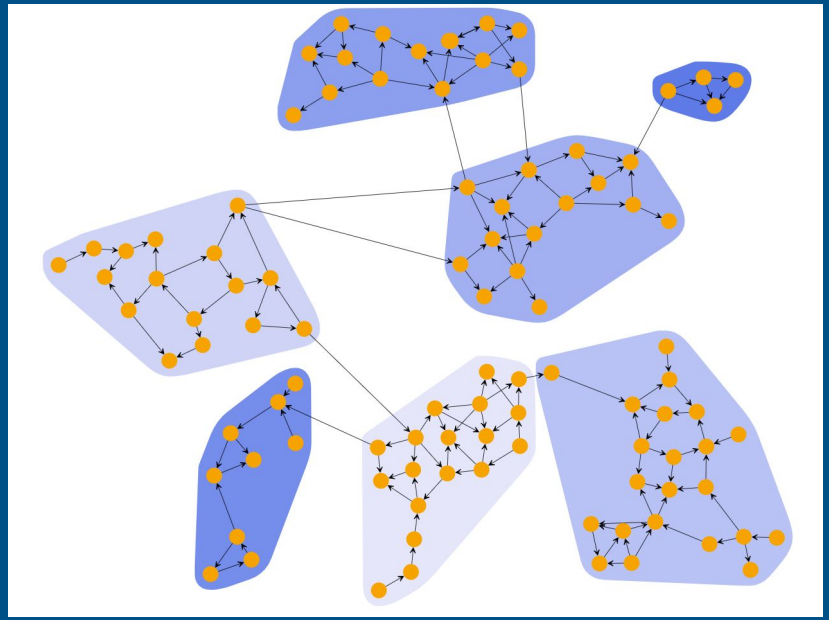
John

Darwhin

Tenzin

Chafiaa

Puja



Introduction to Clustering

Definition of Clustering

- Dividing data objects into groups that have more in common with each other than with those in other groups.
- Includes a family of algorithms such as hierarchical, density-based, or k-means, which group data in different ways.

Importance in Data Analysis

- Useful as a preliminary technique for exploring datasets without labels.
- Example: A company analyzing survey data (e.g., age, gender, past purchases) could use clustering to segment customers and tailor marketing campaigns or product features.

Supervised vs. Unsupervised Learning

Supervised Learning

- Training a model with labeled data.
- Examples: Email filtering (ham vs. spam), Sentiment Analysis,

Unsupervised Learning

- Analyzing unlabeled data to infer patterns.
- Examples: Reducing high-dimensional data (e.g., thousands of variables) to 2D or 3D for easier visualization and interpretation. Clustering is an example of an unsupervised technique.

Clustering in Unsupervised Learning

- Clustering helps reveal hidden structures in data by grouping similar elements.
- Guides decision-making with domain knowledge, though it doesn't explicitly define actions for each group.

Classification vs. Clustering

- **Classification:**
  - Supervised learning method.
  - Requires labeled data for training.
  - Predicts predefined categories or classes.
  - Examples: Email spam detection, disease diagnosis.
- **Clustering:**
  - Unsupervised learning method.
  - Works with unlabeled data.
  - Groups data based on similarities or patterns.
  - Examples: Customer segmentation, image grouping.
- **When to Use Clustering Instead of Classification:**
  - When labels are unavailable or impractical to define.
  - To explore and identify natural groupings or patterns in data.
  - For tasks like anomaly detection, market research, or exploratory data analysis.

Key Takeaway:

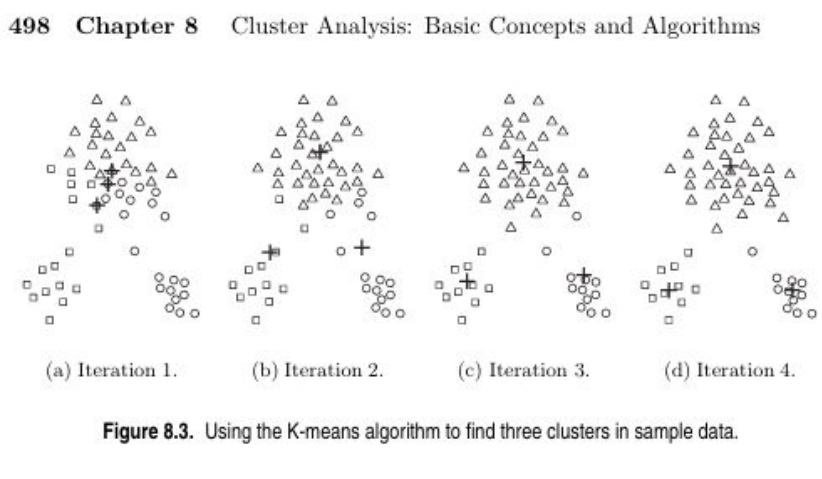
Use clustering to discover hidden patterns, and classification to assign predefined labels based on learned criteria.

Types of Clustering Algorithms

- K-means
- Gaussian Mixture Models
- Hierarchical Clustering
- Density Based Clustering (DBSCAN)

K-means

The k-means algorithm captures the insight that each point in a cluster should be near to the center of that cluster.



The K-Means algorithm operates through an iterative process to partition the dataset into k clusters.

Select the number of clusters (K): Decide the number of clusters you want to form.

Initialize centroids: Randomly select k points as initial centroids.

Assign data points to clusters: Assign each data point to the nearest centroid, forming k clusters.

Update centroids: Calculate the new centroids by taking the mean of all data points assigned to each cluster.

Repeat: Reassign data points to the nearest centroid and update centroids until the centroids no longer change or a maximum number of iterations is reached[12].

K-means:

Minimizing the Sum of Squared Errors

- (SSE) Key Concept: Objective: The main goal of K-Means is to minimize the Sum of Squared Errors (SSE) within clusters.

- A lower SSE indicates that points are closer to their respective centroids, resulting in more cohesive clusters.

- What is Sum of Squared Errors (SSE)?Definition: SSE measures the total squared distance between each data point and the centroid of its assigned cluster.

Table 8.1. Table of notation.

Symbol	Description
$\mathbf{x}$	An object.
C_i	The i^{th} cluster.
$\mathbf{c}_i$	The centroid of cluster C_i.
$\mathbf{c}$	The centroid of all points.
m_i	The number of objects in the i^{th} cluster.
m	The number of objects in the data set.
K	The number of clusters.

$$\mathrm{SSE} = \sum_{i=1}^{K} \sum_{\mathbf{x} \in C_i} dist(\mathbf{c}_i, \mathbf{x})^2$$

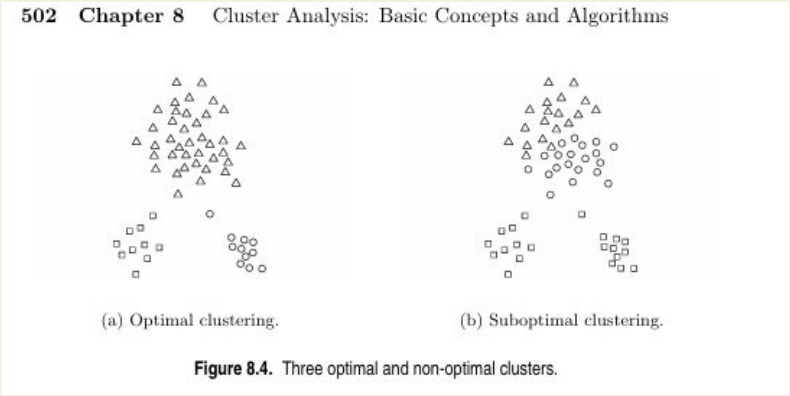
$$\mathbf{c}_i = \frac{1}{m_i} \sum_{\mathbf{x} \in C_i} \mathbf{x}$$

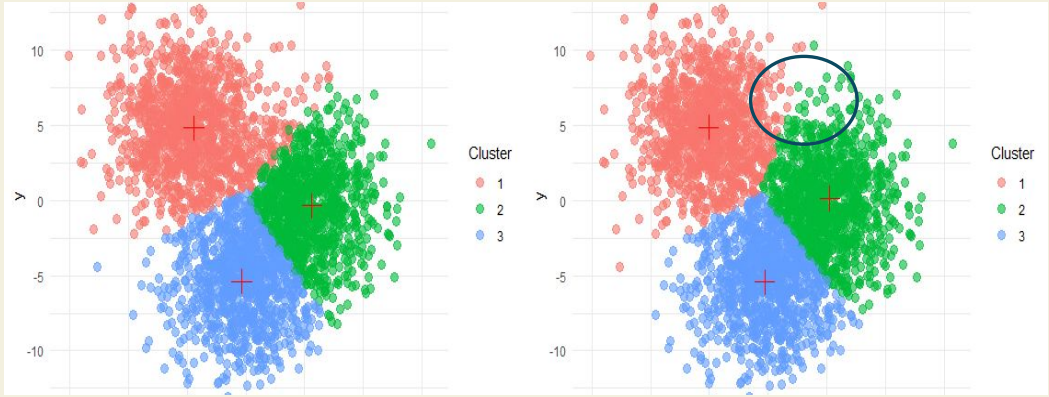
Sum of Squared Errors (SSE)

Why Minimize SSE?

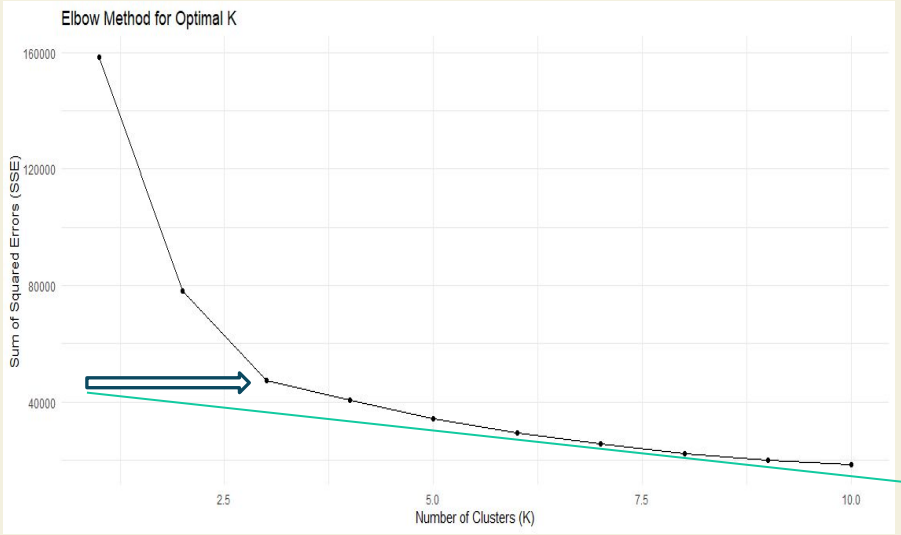
Tighter Clusters: Lower SSE means that points within a cluster are closer together, making the clusters more compact.

Better Representation: Centroids are more representative of their clusters, improving the interpretability and quality of the clustering.





Choosing K



Elbow Method: A technique to find the optimal number of clusters (K) by balancing simplicity and fit quality.

How it works:

Run K-Means with different values of K, calculating Sum of Squared Errors (SSE) for each.

Plot SSE vs. K: Observe how SSE decreases as K increases.

Interpreting the Plot:

Elbow Point: The optimal K is where the SSE reduction slows down sharply, creating an "elbow" shape.

The elbow appears around K=3 where additional clusters bring minimal improvement.

Reasons to use the elbow method for selecting K

- Simplicity: Chooses the smallest number of clusters that represents the data well.
- Avoid Overfitting: Prevents adding unnecessary complexity with extra clusters.

Gaussian Mixture Models

Gaussian Mixture Models (GMMs) are probabilistic clustering methods that assume data is generated from a mixture of Gaussian distributions, offering flexible cluster shapes, soft assignments of data points to clusters, and parameter estimation through the Expectation-Maximization algorithm; they are widely discussed in foundational machine learning texts like *Pattern Recognition and Machine Learning* by Bishop and practical guides like the Scikit-learn documentation

Gaussian mixture distribution

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

Key Concepts of Gaussian Mixture Models (GMMs)

1. **Mixture of Gaussians:** Models data as a combination of Gaussian distributions.
2. **Soft Clustering:** Assigns probabilities of belonging to multiple clusters.
3. **Cluster Parameters:**
    a. μk: Mean (cluster center).
    b. Σk: Covariance matrix (shape/orientation).
    c. *πk*: Mixing weight (proportion of data in each cluster).
4. **Expectation-Maximization (EM)**:
    a. E-Step: Computes cluster probabilities for each point.
    b. M-Step: Updates μk,Σk,πk to maximize likelihood.
5. **Flexible Clusters**: Handles clusters of varying shapes, sizes, and orientations.
6. **Applications**: Clustering, density estimation, image segmentation, and anomaly detection

Expectation-Maximization (EM) Algorithm

EM for Gaussian Mixtures

Given a Gaussian mixture model, the goal is to maximize the likelihood function with respect to the parameters (comprising the means and covariances of the components and the mixing coefficients).

1. Initialize the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k, and evaluate the initial value of the log likelihood.
2. **E step**. Evaluate the responsibilities using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \tag{9.23}$$

The Expectation-Maximization (EM) algorithm is an iterative method used in Gaussian Mixture Models (GMMs) to estimate the parameters (μk, Σk, πk) of each Gaussian component. In the E-step, it computes the probabilities (responsibilities) that each data point belongs to a cluster. In the M-step, it updates the parameters to maximize the likelihood of the observed data. This process repeats until convergence, refining the model to best fit the data.

3. **M step**. Re-estimate the parameters using the current responsibilities

$$\boldsymbol{\mu}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \tag{9.24}$$

$$\boldsymbol{\Sigma}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^{\mathrm{T}} \tag{9.25}$$

$$\pi_k^{\text{new}} = \frac{N_k}{N} \tag{9.26}$$

where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}). \tag{9.27}$$

4. Evaluate the log likelihood

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \tag{9.28}$$

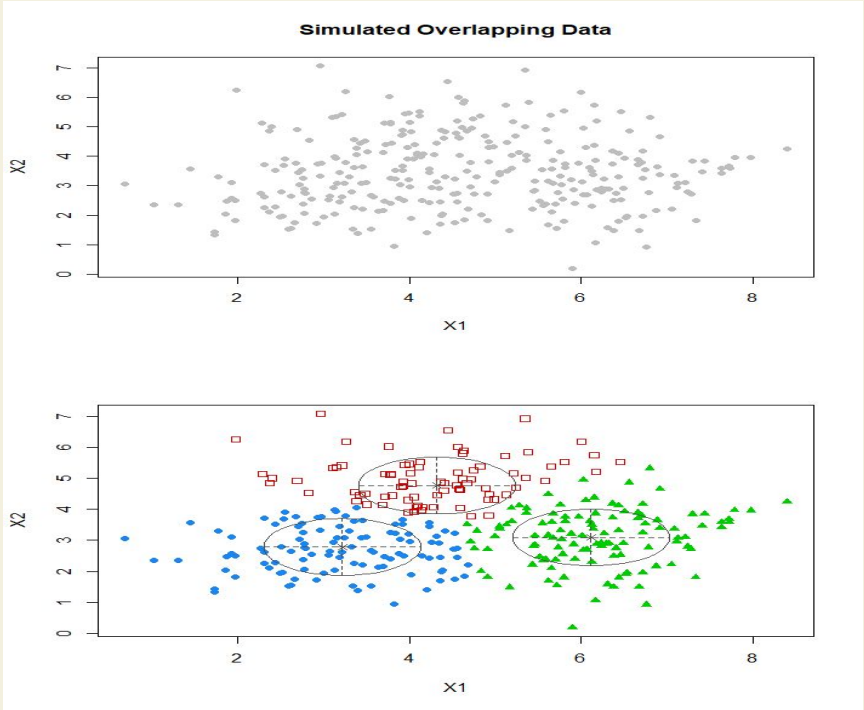
and check for convergence of either the parameters or the log likelihood. If the convergence criterion is not satisfied return to step 2.

Evaluating a GMM

- Log-Likelihood:
  - Measures how well the model explains the data. A higher log-likelihood indicates a better fit.
  - Can be compared across iterations to monitor convergence.
- Information Criteria:
  - Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) penalize model complexity to avoid overfitting.
  - Lower AIC or BIC values indicate a better balance between model fit and complexity.

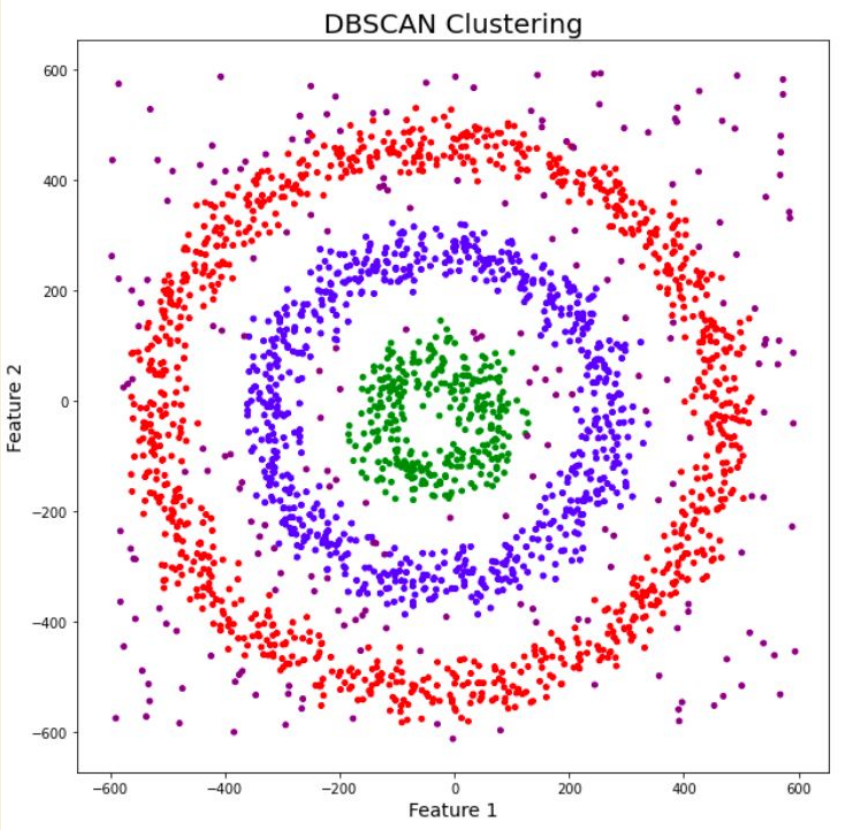
```
library(mclust)
gmm_model <- Mclust(data_overlap)

# Summary of the model
summary(gmm_model)

# Visualize clusters
plot(gmm_model, what = "classification", main = "GMM Clustering")
```

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust EII (spherical, equal volume) model with 3 components:

 log-likelihood   n df       BIC       ICL
      -1017.743 300  9 -2086.82 -2189.53

Clustering table:
  1   2   3
108  79 113
```



DBSCAN

- AKA Density-Based Spatial Clustering of Applications with Noise
- Considers clusters as areas of high density separated by areas of low density.
- Does not require you to specify the number of clusters beforehand.
- Efficient at marking outlier points that sit far away in low-density regions.
- Useful at working with weirdly shaped distributions.



DBSCAN

Parameter Definitions:

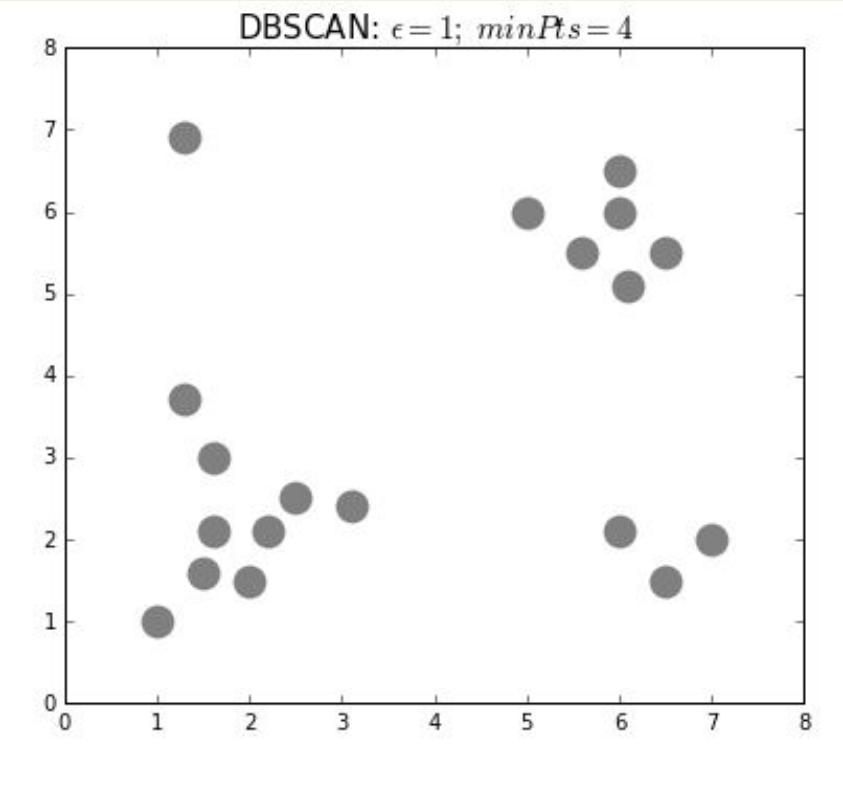
ε -> Distance threshold to determine the “neighborhood” of a single point.

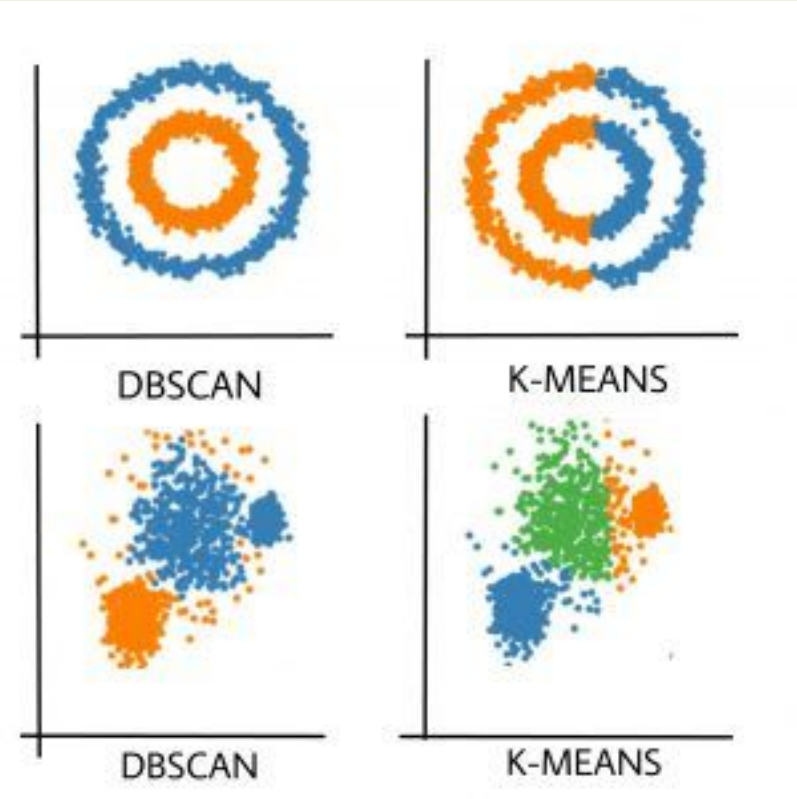
n -> The minimum number of points needed to form a cluster.

Algorithm:

1. Choose a point and determine all points within a distance ε.
2. If the point has at least *n* it is a “core point” (a new cluster is created).
3. Check if neighboring points are also core points.
4. Repeat steps 2-3 until all core points are found. Points that are within distance ε of a core point but do not have enough neighbors themselves are called border points. Finding border points ends the recursive process. process for all neighbor points, continue to expand cluster while new core points are found.
5. Repeat steps 1-4 until all points are classified as being part of a cluster or noise (neither a core point nor a border point).

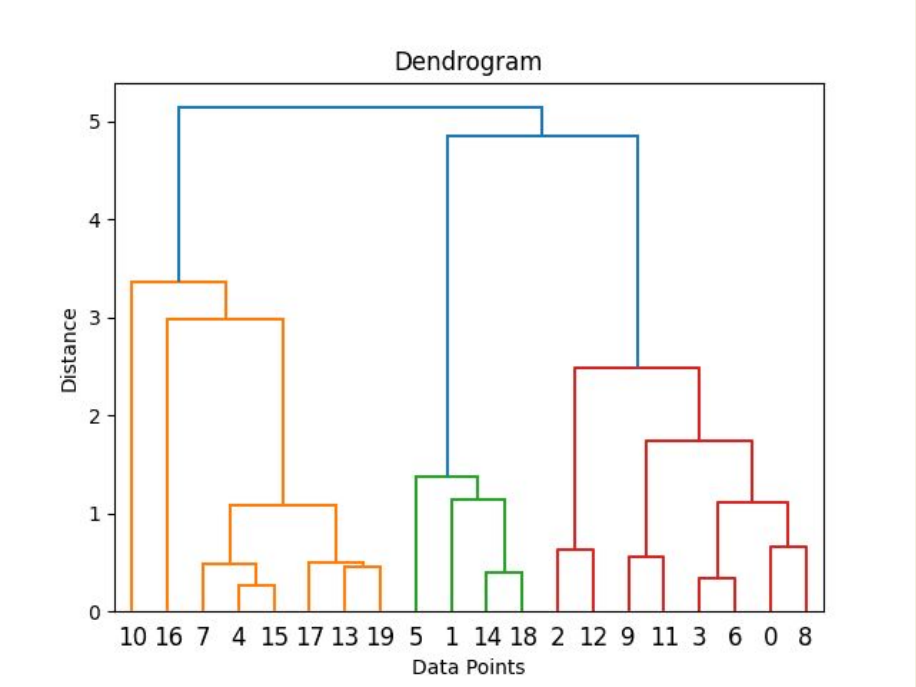
DBSCAN


DBSCAN: $\epsilon = 1;\ minPts = 4$
0
1
2
3
4
5
6
7
8


DBSCAN
K-MEANS
DBSCAN
K-MEANS

HCA

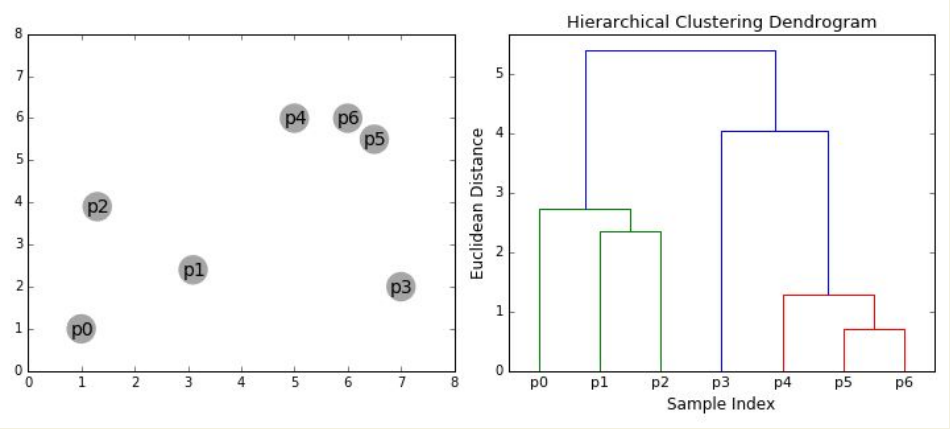
- AKA Hierarchical Clustering Analysis.
- Creates a hierarchical tree of objects to be clustered (a dendogram).
- Does not require you to specify the number of clusters beforehand.
- Very useful in providing information about data structure and patterns.
- Not really useful for large datasets as it is hard to interpret.



HCA (Agglomerative)

Algorithm :

1. Agglomerative (bottom up) HCA involves first treating each individual point as its own cluster.
2. A distance metric is used to calculate the clusters closest to one another and join them.
3. Repeat step two until all points belong to a single cluster.



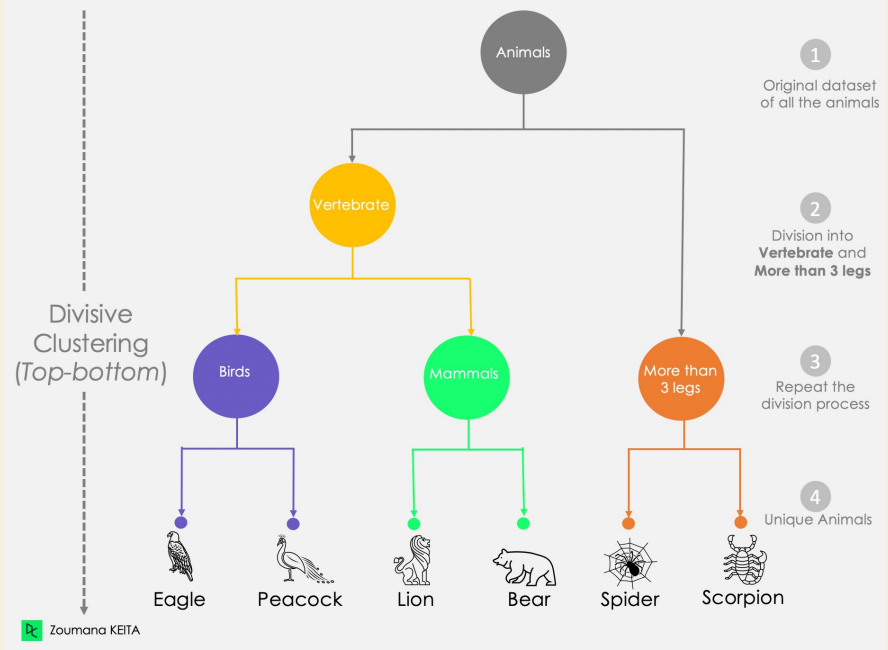
Distance Metrics:

- Euclidean distance, manhattan distance, etc. can be used for points.
- Single linkage, complete linkage, average linkage can be used for clusters.

HCA (Divisive)

Algorithm :

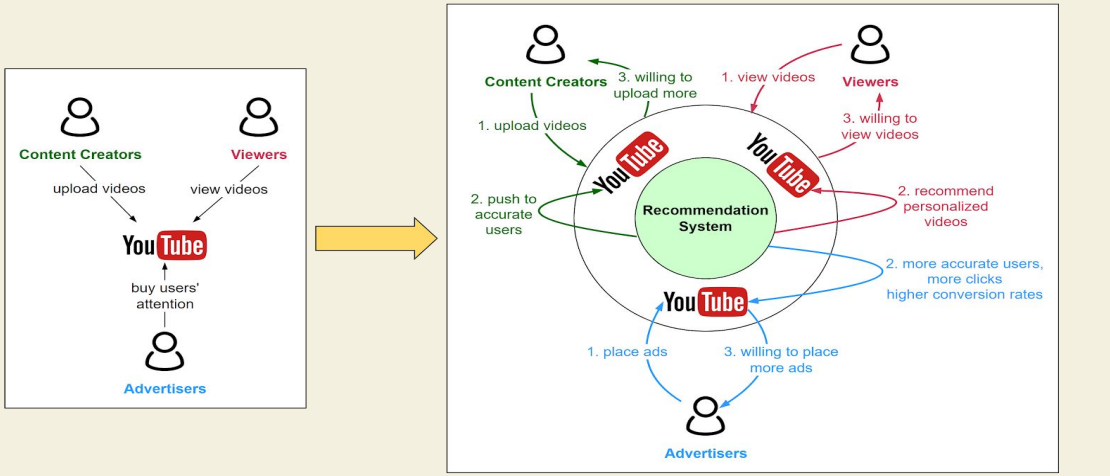
1. Divisive (top down) HCA involves starts with all points as a single cluster.
2. A distance metric is used to determine dissimilarity between points.
3. Use data from Step 2 to split the cluster using another clustering method (i.e. k-means).
4. Repeat step two until all points are their own cluster.



Areas of Application

1. **Recommendation Engines (Example: Amazon, YouTube)**

- Recommendation systems like Amazon and YouTube use clustering to group users with similar preferences, making it easier to suggest products or content.

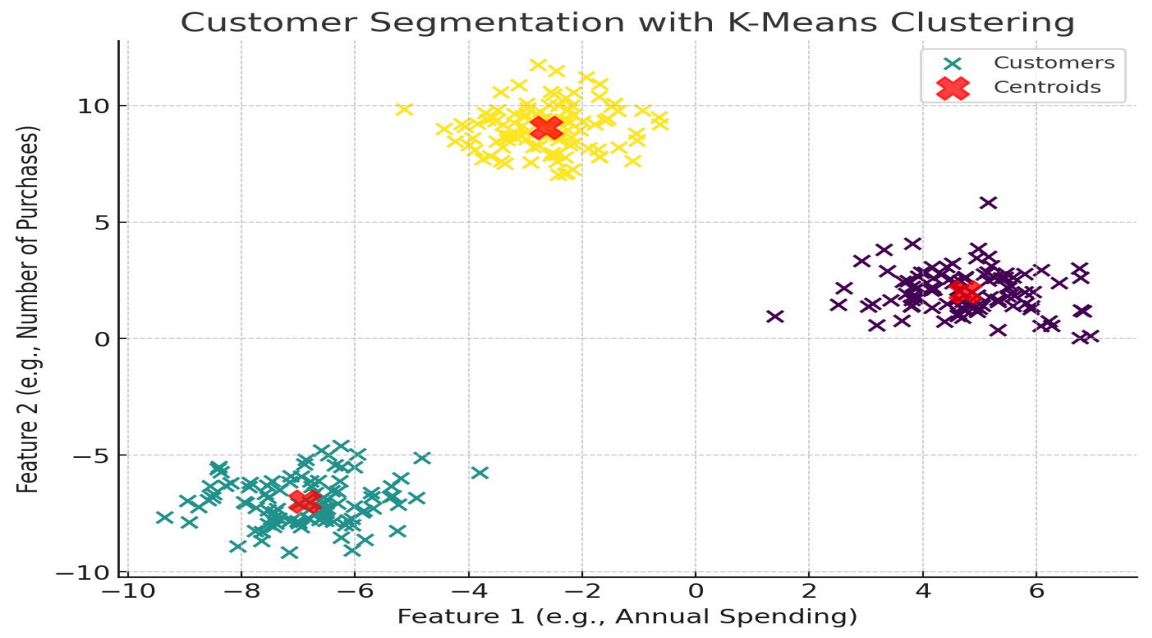


ByteByteGo

Areas of Application

2. Market and Customer Segmentation (Example: Marketing Campaigns)

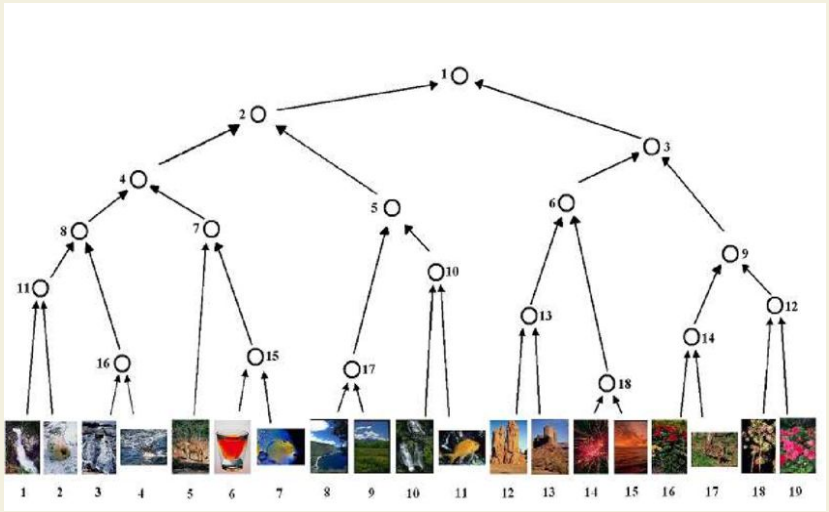
- Group customers based on purchasing behavior, demographics, or browsing patterns

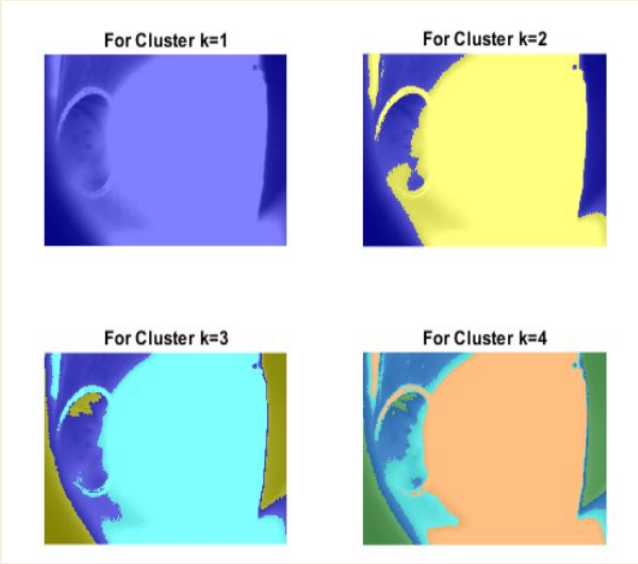


Areas of Application

3. Image Processing (Example: Object Recognition)

- In image processing, clustering helps group similar features in images, which is useful in tasks like object recognition.

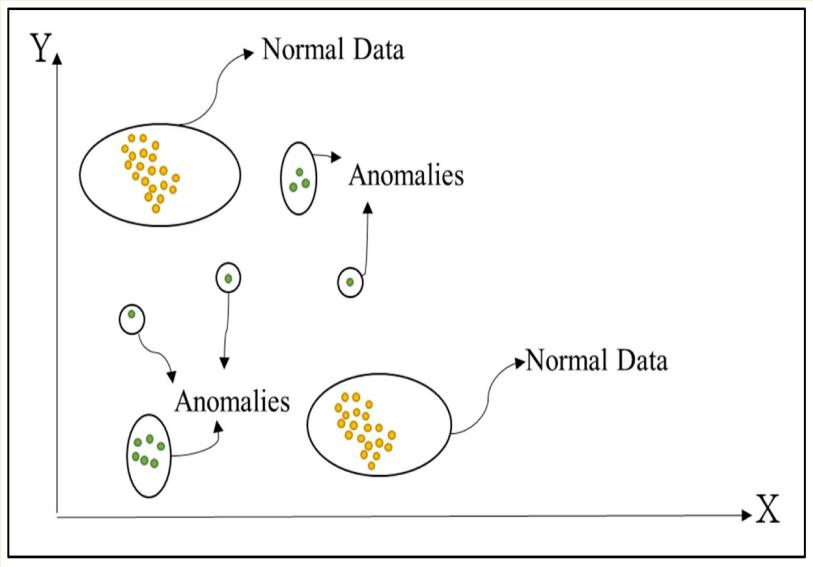




Areas of Application

4. Fraud Detection (Example: Banking Transactions)

- In fraud detection, clustering helps identify unusual patterns or behaviors.



Types of Anomalies

Point Anomalies: Transactions that don't fit any cluster (e.g., an unusually large purchase) are flagged as fraud.

Contextual Anomalies: Transactions that are normal in general but out of place in context (e.g., a late-night purchase) are detected.

Collective Anomalies: A group of unusual transactions can form a separate cluster, indicating fraud.

Challenges in Clustering:

1. **Determining the Number of Clusters:** Deciding the optimal number of clusters is challenging. It often involves using techniques like the elbow method, silhouette analysis, or gap statistics, but these methods may yield conflicting results depending on the dataset.

2. **Sensitivity to Noise and Outliers:** Clustering algorithms can be highly sensitive to noise, leading to distorted clusters or misclassification. For instance, K-means is particularly vulnerable to outliers because it minimizes variance, which can be skewed by extreme values.

3. **Choosing the Right Clustering Method:** No single clustering method works universally. Methods like K-means are effective for spherical clusters, but others like DBSCAN are better for arbitrary shapes. Choosing an unsuitable method can lead to inaccurate clustering.

4. **Interpreting Nonlinear Models:** Algorithms like spectral clustering or deep learning-based clustering produce complex, nonlinear models that are harder to interpret compared to simpler methods like K-means.

5. **Computational Complexity:** Clustering large datasets can be computationally intensive, especially with algorithms like hierarchical clustering or Gaussian Mixture Models, which may not scale well.

Considerations for Clustering:

1. **Data Preprocessing:** Proper data preprocessing is essential for quality results. This includes:
    - Handling missing values.
    - Normalizing or scaling features to ensure equal contribution.
    - Addressing imbalanced data distributions if they exist.
2. **Model Validation:**
    - Ensuring the model generalizes well to new data involves techniques like: Cross-validation.
    - Evaluating cluster stability by bootstrapping or repeated sampling.
3. **Model Interpretability:** The clusters must make sense in the real-world context of the problem. Highly complex models might give technically accurate clusters but are difficult to explain or apply practically.
4. **Scalability:** With large datasets, consider algorithms optimized for scalability, such as MiniBatch K-means or distributed implementations of clustering methods.
5. **Avoiding Overfitting:** Overfitting in clustering can occur when the model fits the noise or outliers in the data. Regular validation and noise reduction techniques can help mitigate this risk, By keeping these challenges and considerations in mind, clustering can be applied more effectively in real-world scenarios.

Results and Conclusion

Effectiveness of Clustering in Predictive Analytics

- Unsupervised learning allows for discovering hidden patterns without labeled data.
- Clustering is effective for segmenting data into meaningful groups, such as customer segmentation, anomaly detection, or market trend analysis.
- It helps generate insights that drive decisions in marketing, business strategies, and product development.
- Results from clustering models can enhance predictive models by revealing patterns and relationships that were not initially obvious.

Final Thoughts

- Clustering remains a crucial tool in analytics, offering value in exploratory data analysis and when labels are not available.
- As data becomes more complex, clustering will continue to evolve with advancements in algorithms and integration with other AI techniques like deep learning.
- Future Developments: Continued improvement in clustering techniques could enable even more accurate groupings in larger, more complex datasets, enhancing decision-making in real-time applications.

References

1. Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
2. Kuhn, M., & Johnson, K. (2013). *Applied Predictive Modeling*. Springer.
3. Tan, P.-N., Steinbach, M., & Kumar, V. (2005). *Introduction to Data Mining*. Pearson.
4. R Documentation: mclust Package: URL: https://cran.r-project.org/web/packages/mclust/index.html
5. Eric J, Why Do We Use Clustering? 5 Benefits and Challenges In Cluster Analysis **https://datarundown.com/why-clustering/**
6. geeksforgeeks, DBSCAN Clustering in ML | Density based clustering **https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/**