Fundamentals of Artificial Intelligence

Unsupervised Learning: Clustering

Shyamanta M Hazarika
Mechanical Engineering
Indian Institute of Technology Guwahati
s.m.hazarika@iitg.ac.in

http://www.iitg.ac.in/s.m.hazarika/

Supervised Learning vs. Unsupervised Learning

- Supervised learning: discover patterns in the data that relate data attributes with a target (class) attribute.
  - These patterns are then utilized to predict the values of the target attribute in future data instances.
- Unsupervised learning: The data have no target attribute; discover structure underlying the data.
  - Explore the data to find some intrinsic structures in them.
  - Unsupervised learning is more about creative endevours – exploration, understanding and refinement.

Clustering

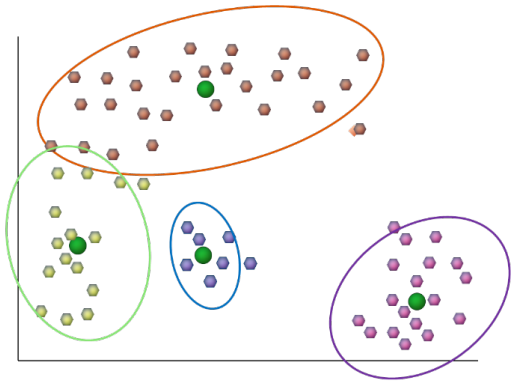
- Clustering is a technique for finding **similarity groups** in data, called **clusters** i.e.,
  - It groups data instances that are similar to (near) each other in one cluster and data instances that are very different (far away) from each other into different clusters.

    In clustering problems, class labels are not specified; only the feature vectors representing different objects/instances/ records or situations are known.

- Clustering is often called an **unsupervised learning** task as no class values denoting an *a priori* grouping of the data instances are given, which is the case in supervised learning.

Clustering

- A cluster is a collection of data items which are "similar" between them, and "dissimilar" to data items in other clusters



Examples of Clustering

- Example 1:Document Clustering

  Given a collection of text documents, we want to organize them according to their content similarities.

  - To produce a topic hierarchy.

    For instance, news reports can be further divided to those pertaining to politics, entertainment, sports, and so on.

- Example 2:Learning sequence of amino acids

  Clustering is used in bioinformatics in learning sequence of amino acids that occur repeatedly in protein; they may correspond to structural or functional elements within the sequence.

Examples of Clustering

- Example 3: Categorization

  For example group people of similar sizes together to make "small", "medium" and "large" T-Shirts.

  - Tailor-made for each person: too expensive
  - One-size-fits-all: does not fit all.

- Example 4: Customer segmentation

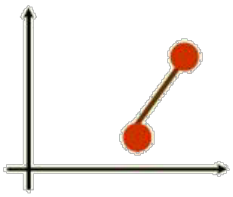
  In marketing, segment customers according to their similarities.

  - To do targeted marketing.

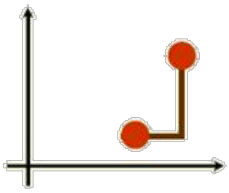
What do we need for Clustering?

- Proximity Measure
  - Similarity Measure
  - Dissimilarity Measure

Dissimilarity Measure - Distance



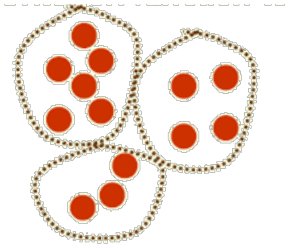
Euclidean Distance



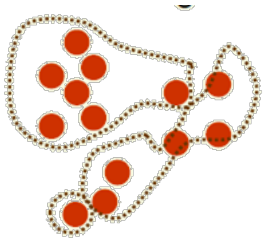
Manhattan Distance

What do we need for Clustering?

- Criterion function to evaluate a clustering



Good Clustering



Bad Clustering

- Algorithm to compute clustering
  - Optimizing the criterion function.

Basic Clustering Methods

- There are many clustering algorithms. A crisp categorization of the clustering methods is difficult.
- Major fundamental clustering methods include
  1. Partitional Clustering

     Partitioning is the most simple and basic version of cluster analysis.
  2. Hierarchical Clustering
  3. Spectral Clustering
  4. Clustering using Self-Organizing Maps

Partitional Clustering

- Partitioning is the most simple and basic version of cluster analysis.
- Formally, if a set of *N* objects is given, a partitioning technique will create *K* divisions of the data, where each division or partition is a representative of a cluster, $K \leq N$.

  It partitions the data into *K* groups in a way such that each group must comprise a minimum of one object.

K-means clustering

- K-means is a partitional clustering algorithm
- Let the set of data points (or instances) D be

  $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$,

  where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ is a vector in a real-valued space $X \subseteq R^r$, and r is the number of attributes (dimensions) in the data.
- The k-means algorithm partitions the given data into k clusters.
  - Each cluster has a cluster **center**, called **centroid**.
  - k is specified by the user

K-means algorithm

- Given k, the *k-means* algorithm works as follows:
  1. Randomly choose k data points (seeds) to be the initial centroids, cluster centers
  2. Assign each data point to the closest centroid
  3. Re-compute the centroids using the current cluster memberships.
  4. If a convergence criterion is not met, go to 2.

Convergence Criterion

1. no (or minimum) re-assignments of data points to different clusters,
2. no (or minimum) change of centroids, or
3. minimum decrease in the **sum of squared error** (SSE),

$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} dist(\mathbf{x}, \mathbf{m}_j)^2$$

- C_j is the jth cluster, $\mathbf{m}_j$ is the centroid of cluster C_j (the mean vector of all the data points in C_j), and $dist(\mathbf{x}, \mathbf{m}_j)$ is the distance between data point $\mathbf{x}$ and centroid $\mathbf{m}_j$.

K-Means: Strengths

- Simple: easy to understand and to implement
- Efficient: Time complexity: $O(tkn)$,
  where n is the number of data points,
  k is the number of clusters, and
  t is the number of iterations.
- Since both k and t are small. k-means is considered a linear algorithm.

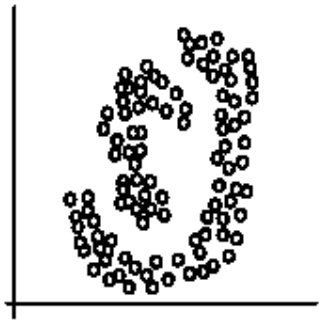
- K-means is the most popular clustering algorithm.
- Note that: it terminates at a local optimum if SSE is used. The global optimum is hard to find due to complexity.

K-means: Weaknesses

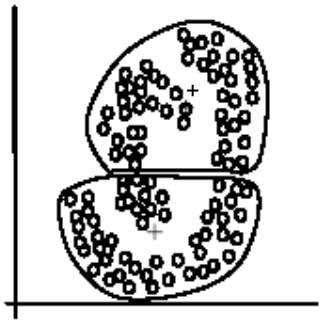
- The algorithm is only applicable if the mean is defined.
  - For categorical data, *k*-mode - the centroid is represented by most frequent values.
- The user needs to specify *k*.
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values.

K-means: Weaknesses

- The *k*-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).



Two Natural Clusters

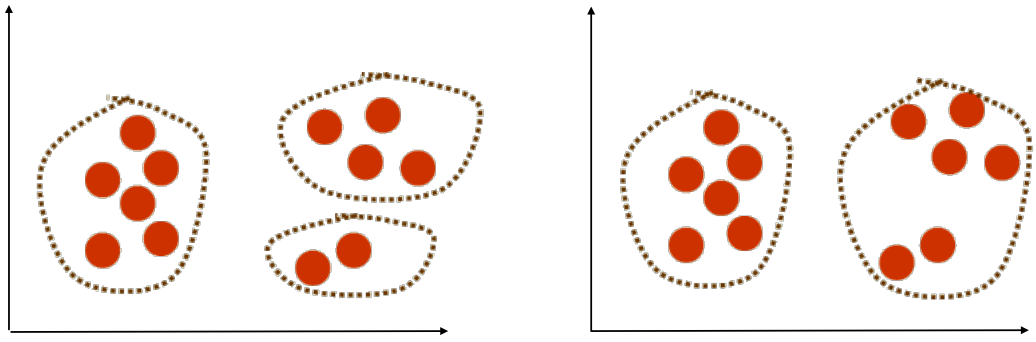


K-means Clusters

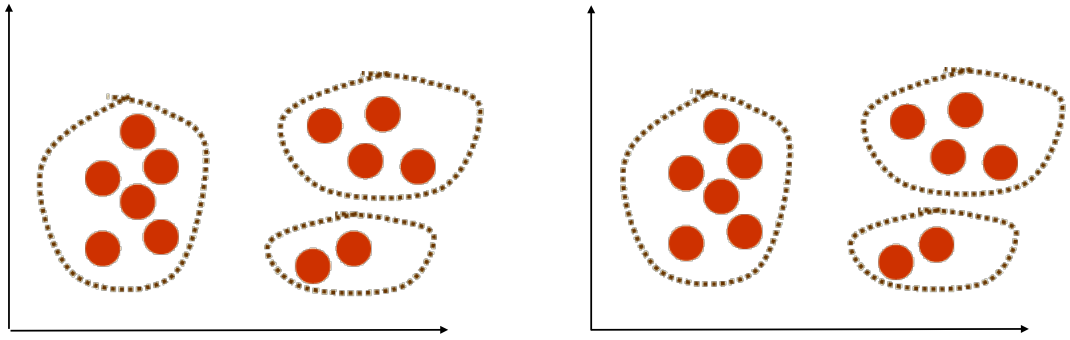
K-means: Summary

- Despite weaknesses, *k*-means is still the most popular algorithm due to its simplicity, efficiency and
  - other clustering algorithms have their own lists of weaknesses.
- No clear evidence that any other clustering algorithm performs better in general
  - although they may be more suitable for some specific types of data or applications.
- Comparing different clustering algorithms is a difficult task. No one knows the correct clusters!

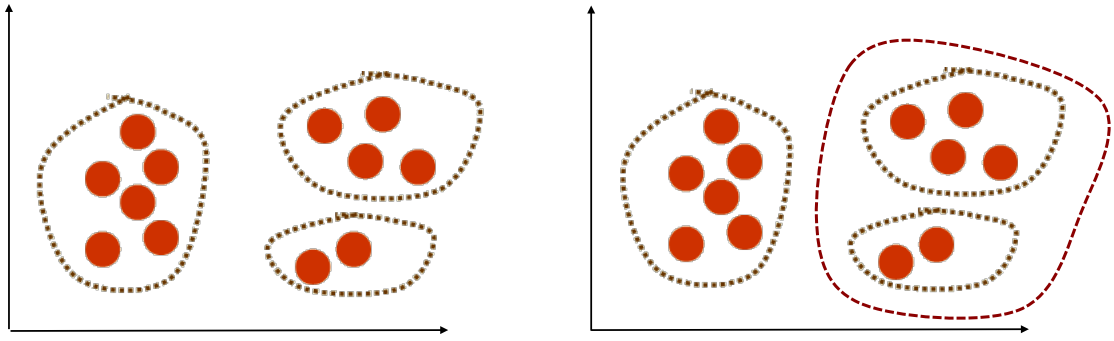
Hierarchical Clustering



Hierarchical Clustering



Hierarchical Clustering



Hierarchical Clustering

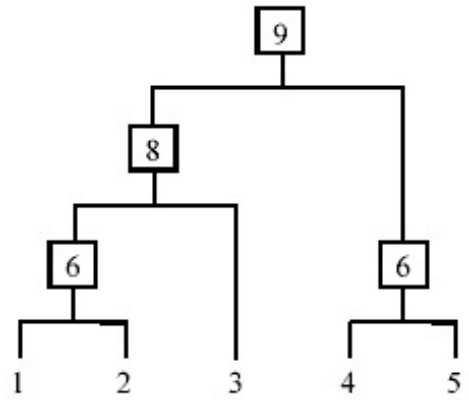
Partitioning-based techniques are not based on the assumption that substructures exist in the clusters. There may be instances when data is organized in a hierarchical manner, that is, clusters have subclusters within subclusters and so on.

- Produce a nested sequence of clusters, a tree, also called Dendrogram.

Given a dataset X with N items, consider a sequence of divisions of its elements into K clusters; an integer between 1 and N; a number not fixed a priori.

First possible division is one that divides the data into N groups. Second partition divides X into N-1 clusters by merging closest observations into a cluster. Third into N-2 clusters ..... progressively combining or agglomerating two closest clusters.

Level L when K = N – L + 1 ; at Level 1 we have N clusters and at Level N we have 1 cluster.



Types of hierarchical clustering

- Agglomerative (bottom up) clustering:
  - It builds the dendrogram from the bottom level
    - merges the most similar (or nearest) pair of clusters
    - stops when all the data points are merged into a single cluster (i.e., the root cluster).
- Divisive (top down) clustering:
  - It starts with all data points in one cluster, the root.
    - Splits the root into a set of child clusters. Each child cluster is recursively divided further
    - stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point

Agglomerative vs. Divisive

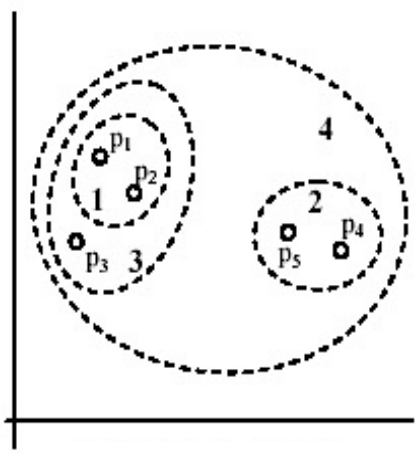
Agglomerative is more popular then divisive methods.

- At the beginning, each data point forms a cluster (also called a node).
- Merge nodes/clusters that have the least distance.
- Go on merging
- Eventually all nodes belong to one cluster

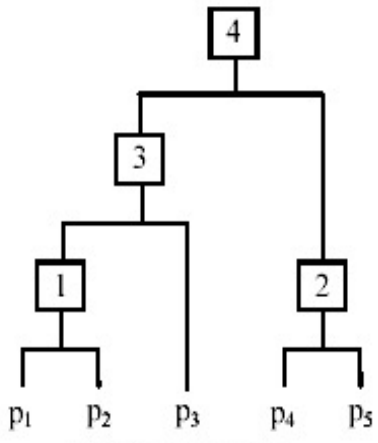
Agglomerative processes begin with N singletons and form the sequence with successive merging of clusters. Simpler computation required from one level to next.

Divisive processes begin with all the sample in a single cluster and create the sequence by consecutively separating clusters.

An example



Nested Clusters



Dendogram

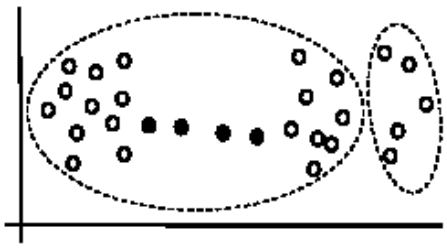
Distance of Two clusters

Whether employing agglomerative or divisive technique, required to measure the distance between two clusters; where each is usually a set of objects.

- Four commonly used measures for distance between two clusters:
  - Single linkage
  - Complete linkage
  - Average linkage
  - Centroid linkage

The four measures lead to FOUR variants of the Hierarchical Clustering Algorithm.

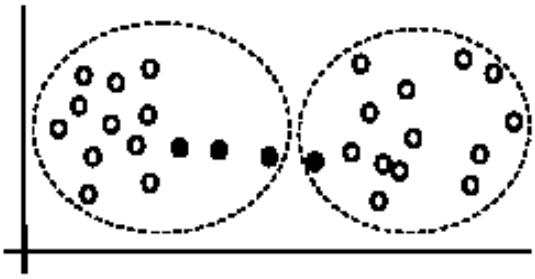
Single-link Method



Two natural clusters are split into two

- The distance between two clusters is the distance between two closest data points in the two clusters, one data point from each cluster.
- It can find arbitrarily shaped clusters, but
  - It may cause the undesirable "chain effect" by noisy points

Complete-link Method



- The distance between two clusters is the distance of two furthest data points in the two clusters.
- It is sensitive to outliers because they are far away

Other Methods

- Average Link: A compromise between
  - the sensitivity of complete-link clustering to outliers and
  - the tendency of single-link clustering to form long chains that do not correspond to the intuitive notion of clusters as compact, spherical objects.
  - In this method, the distance between two clusters is the average distance of all pair-wise distances between the data points in two clusters.
- Centroid Method: In this method, the distance between two clusters is the distance between their centroids

Data Standardization

- Data often calls for general transformations of a set of attributes selected for the problem.
  - Might be useful to define NEW attributes by applying speficied mathematical functions to the existing ones.
  - New variable derived OFTEN express the information inherent in the data in ways that make the information more useful; thereby improving model performance.
- In the Euclidean space, standardization of attributes is recommended so that all attributes can have equal impact on the computation of distances.
- Standardize attributes: to force the attributes to have a common value range

Interval-scaled Attributes

- Their values are real numbers following a linear scale.
  - For example: The difference in Age between 10 and 20 is the same as that between 40 and 50.
  - The key idea is that intervals keep the same importance through out the scale
- Two main approaches to standardize interval scaled attributes, **range** and **z-score**. *f* is an attribute

$$range(x_{if}) = \frac{x_{if} - \min(f)}{\max(f) - \min(f)}$$

Interval-scaled Attributes

- Z-score: transforms the attribute values so that they have a mean of zero and a **mean absolute deviation** of 1. The mean absolute deviation of attribute *f*, denoted by s_f is computed as follows

$$s_f = \frac{1}{n}\left(|x_{1f} - m_f| + |x_{2f} - m_f| + \ldots + |x_{nf} - m_f|\right)$$

$$m_f = \frac{1}{n}\left(x_{1f} + x_{2f} + \ldots + x_{nf}\right)$$

Z-score: $z(x_{if}) = \frac{x_{if} - m_f}{s_f}$

Ratio-scaled attributes

- Numeric attributes, but unlike interval-scaled attributes, their scales are exponential,
- For example, the total amount of microorganisms that evolve in a time *t* is approximately given by Ae^{Bt} where *A* and *B* are some positive constants.
- Do log transform: $\log(x_{if})$
  - Then treat it as an interval-scaled attribuete

Nominal Attributes

- Sometime, we need to transform nominal attributes to numeric attributes.
- Transform nominal attributes to binary attributes.
  - The number of values of a nominal attribute is *v*.
  - Create *v* binary attributes to represent them.
  - If a data instance for the nominal attribute takes a particular value, the value of its binary attribute is set to 1, otherwise it is set to 0.
- The resulting binary attributes can be used as numeric attributes, with two values, 0 and 1.

Nominal Attributes: An Example

- Nominal attribute *fruit*: has three values,
  - Apple, Orange, and Pear
- We create three binary attributes called, Apple, Orange, and Pear in the new data.
- If a particular data instance in the original data has Apple as the value for *fruit*,
  - then in the transformed data, we set the value of the attribute Apple to 1, and
  - the values of attributes Orange and Pear to 0

Ordinal Attributes

- Ordinal attribute: an ordinal attribute is like a nominal attribute, but its values have a numerical ordering.

  E.g.,
  - Age attribute with values: Young, MiddleAge and Old. They are ordered.
  - Common approach to standardization: treat is as an interval-scaled attribute.

Which Clustering Algorithm?

- Clustering research has a long history. A vast collection of algorithms are available.
  - We only introduced several main algorithms.
- Choosing the "best" algorithm is a challenge.
  - Every algorithm has limitations and works well with certain data distributions.
  - It is very hard, if not impossible, to know what distribution the application data follow.
    - The data may not fully follow any "ideal" structure or distribution required by the algorithms.
    - Decide how to standardize the data, to choose a suitable distance function and to select other parameter values.

Which Clustering Algorithm?

- Due to these complexities, the common practice is to
  - run several algorithms using different distance functions and parameter settings, and
  - then carefully analyze and compare the results.
- The interpretation of the results must be based on insight into the meaning of the original data together with knowledge of the algorithms used.
- Clustering is highly application dependent and to certain extent subjective (personal preferences).

Cluster Evaluation: A Difficult Problem

- The quality of a clustering is very difficult to evaluate because
  - We do not know the correct clusters
- Some methods are used:
  - User inspection
    - Study centroids, and spreads
    - Rules from a decision tree.
    - For text documents, one can read some documents in clusters.

Cluster Evaluation: Ground Truth

- We use some labeled data (for classification)
- Assumption: Each class is a cluster.
- After clustering, a confusion matrix is constructed. From the matrix, we compute various measurements, entropy, purity, precision, recall and F-score.
  - Let the classes in the data D be $C = (c_1, c_2, \ldots, c_k)$. The clustering method produces k clusters, which divides D into k disjoint subsets, $D_1, D_2, \ldots, D_k$.

Evaluation based on Internal Information

- **Intra-cluster cohesion** (compactness):
  - Cohesion measures how near the data points in a cluster are to the cluster centroid.
  - Sum of squared error (SSE) is a commonly used measure.
- **Inter-cluster separation** (isolation):
  - Separation means that different cluster centroids should be far away from one another.
- Expert judgments are still the key!

Indirect Evaluation

- In some applications, clustering is not the primary task, but used to help perform another task.
- We can use the performance on the primary task to compare clustering methods.
- For instance, in an application, the primary task is to provide recommendations on book purchasing to online shoppers.
  - If we can cluster books according to their features, we might be able to provide better recommendations.
  - We can evaluate different clustering algorithms based on how well they help with the recommendation task.
  - Here, we assume that the recommendation can be reliably evaluated.

Summary

- Clustering is has along history and still active
  - There are a huge number of clustering algorithms
  - More are still coming every year.
- We only introduced a couple of main algorithms. There are many others, e.g.,
  - density based algorithm, sub-space clustering, scale-up methods, neural networks based methods, fuzzy clustering, co-clustering, etc.
- Clustering is hard to evaluate, but very useful in practice. This partially explains why there are still a large number of clustering algorithms being devised every year.
- Clustering is highly application dependent and to some extent subjective.