# Cluster analysis

**What is cluster analysis?**

Cluster analysis groups data objects based only on information found in the data that describes the objects and their relationships. The goal is to have objects within a cluster to be similar to one another and different to the objects in other clusters. Clustering (cluster analysis) is the process of grouping a set of objects into multiple clusters. Clustering is a way to discover previously unknown groups in the data. Clustering is unsupervised learning since no information of class label is given.

A cluster is a collection of data objects. The objects are similar to one another within the same cluster and dissimilar to the objects in other clusters. Similarity/Dissimilarity is defined with distance measures.

**Requirements for clustering**

* Scalability: many clustering algorithms work well on smaller data sets, but data may contain vast amounts of data. Clustering on a sample of data may give biased results. Therefore, highly scalable clustering algorithms are needed.
* Ability to deal with different types of attributes: many algorithms work only on numeric (interval-based) data. But we might need to deal with other data types.
* Discovery of clusters with arbitrary shapes: many clustering algorithms only find spherical shapes (the ones based on Euclidean or Manhattan distance). Sometimes algorithms that find other shapes are needed.
* Minimal requirements for domain knowledge to determine input parameters: Many algorithms require that we specify domain knowledge in the form of input parameters, such as the desired number of clusters. The clustering algorithm may be sensitive to such parameters, which can be hard to determine.
* Able to deal with noise and outliers: real world data can have a lot dirty data. Some algorithms can be sensitive to noise. We need algorithms that are robust to noise.
* Insensitive to order of input records: some algorithms may be sensitive to input order, given a set of objects they may return different clusterings depending on the order in which the objects are presented. We need algorithms that are insensitive to input order.
* High dimensionality: most clustering algorithms can handle low-dimensional data. Finding clusters in a higher dimensional space is hard.
* Incorporation of user-specified constraints: we may need to perform clustering under some constraints. It is challenging to find clusterings that satisfy specified constraints.
* Interpretability and usability: we want clusterings that are easy to interpret and use

**Major clustering approaches:**

* Partitioning approach: construct various partitions and then evaluate them by som criterion (eg minimizing the sum of squares). Typical methods: K-Means, K-Medoids, CLARANS
* Hierarchical approach: create a hierarchical decomposition of the set of data (or objects) using some criterion. Typical methods: Diana, Agnes, BIRCH, ROCK, CHAMELEON
* Density-based approach: Based on connectivity (anslutning) and density functions. Typical methods: DBSCAN, OPTICS

## Types of data in cluster analysis

|  |  |
| --- | --- |
| Data matrix:  Structure that stores objects over attributes. matrix.  Each row corresponds to an object. |  |
|  |  |
| Dissimilarity matrix:  Structure that stores a collection of proximities that are available for all pairs of objects.  is dissimilarity between objects i and j.  The similarity between objects i and j is |  |

**Interval-scaled variables (numeric)**

Continuous measurements. Sometimes we need to standardize the data:

* Calculate the mean:
* Calculate the mean absolute deviation:
* Standardized measurement (z-score):

is the f:th variable

Distances are normally used to measure the similarity/dissimilarity between two objects.

* distance is a non-negative number
* distance of an object to itself is 0
* distance is a symmetric function
* going directly from i to j in space is no more than making a detour over any k.

Usual distance measurements:

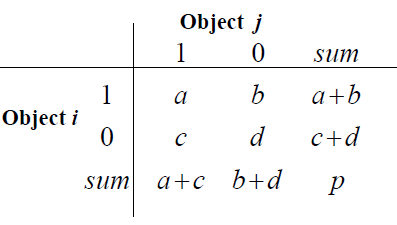
* Euclidean:
* Manhattan:

where and are two p-dimensional objects

**Binary variables**

* Symmetric binary variables: both states are equally important; 0/1
* Asymmetric binary variables: one state is more important than the other; 1 is the important state, 0 the other

To compute dissimilarity for binary variables, we make a contingency table for binary variables:



* is number of attributes having 1 for object i and 1 for object j
* is number of attributes having 1 for object i and 0 for object j
* is number of attributes having 0 for object i and 1 for object j
* is number of attributes having 0 for object i and 0 for object j

For symmetric binary variables where both states matter, distance measure is

For asymmetric binary variables where only state 1 is important, distance measure is  
where the Jaccard coefficient is

**Nominal attributes**

Categorical data. Each value represents some kind of category. The values have no order. A nominal attribute can take on two or more states. The number of states of a nominal attribute is M. The states are denoted by 1,2,…,M (only for data handling, doesn’t represent any specific ordering).

Dissimilarity between objects i and j is computed based on the ratio of mismatches:

where and

We can also calculate distance with another method. Nominal variable is encoded using asymmetric binary attributes by creating a new binary attribute for each of the M states (think dummy but keep all dummys).

**Ordinal variables**

Can be both discrete or continuous but order (rank) is important for the values. M is the number of states in the variable. These ordered states define the ranking . To compute distance for ordinal variables:

1. The value of variable for the i:th object is and has ordered states representing the ranking . Replace with its corresponding rank
2. Standardize the rankings. Map the range of each variable onto [0,1] by replacing the i:th object in the :th variable by
3. Use interval-based methods (Euclidean or Manhattan) to calculate dissimilarity

**Ratio-scaled variables**

Apply logarithmic transformation and treat them as continuous ordinal data. Treat their rank as interval-scaled

**Variables of mixed types**

A database may contain all of these types of variables. One may use a weighted formula to combine their effects:

Indicator if and only if:

* or is missing, or
* and variable is asymmetric binary

Otherwise

* If is binary or nominal: if or otherwise
* If is numeric (interval-based): use the (normalized) distance where runs over all non-missing objects for attribute
* If is ordinal or ratio-scaled: compute ranks and and treat as numeric

**Cosine similarity**

Measure of similarity that can be used to compare documents. Let **x** and **y** be two vectors. Cosine similarity is

where is the Euclidean norm for vector **x**.

## Calculate the distance between clusters

Proximity between two cluster is computed with linkage measures.

We have two clusters and . The distance between clusters is with…

* Single link: smallest distance between an element in one cluster and an element in the other cluster
* Complete link: largest distance between an element in one cluster and an element in the other
* Average: average distance between an element in one cluster and an element in the other
* Centroid: distance between the centroids of two clusters
* Medoid: A medoid is one chosen, centrally located point in each cluster. The distance here is between two medoids of two clusters

**For numerical measures:**

Centroid: the “middle” of a cluster

Radius: square root of average distance from any point of the cluster to its centroid

Diameter: square root of average mean squared distance between all pair of points in the cluster