Cluster Analysis

Table of Contents

[Applications 2](#_Toc106495092)

[Data Distribution Analysis 2](#_Toc106495093)

[Pre-Processing 3](#_Toc106495094)

[Quality of Clustering 4](#_Toc106495095)

[Similarity Metric 4](#_Toc106495096)

[Considerations 5](#_Toc106495097)

[Partitioning Algorithms 6](#_Toc106495098)

[K-Means Clustering Method 6](#_Toc106495099)

[Local Optima 8](#_Toc106495100)

[Number of Clusters 9](#_Toc106495101)

[Strengths and Weaknesses 10](#_Toc106495102)

[K-Medoid Clustering Algorithm 11](#_Toc106495103)

[Choosing 12](#_Toc106495104)

A **cluster** is a collection of data objects. Samples from the same cluster are similar to each other, while samples from difference clusters are dissimilar to each other.

**Cluster analysis**, also called clustering or data segmentation, is the process of finding similarities between data based on characteristics found in the data and grouping the similar objects into clusters. A simple example may be that we group together the data points that are ‘near’ each other. Of course, them being near each other just means that for some given feature, they have similar values.

Cluster analysis is an **unsupervised learning** algorithm, since we do not know beforehand which samples belong to which class. It can either work as a stand-alone tool to gain insight into the data distribution or as a pre-processing step for other algorithms.

## Applications

### Data Distribution Analysis

* Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
* Information retrieval: document clustering
* Land use: Identification of areas of similar land use in an earth observation database
* Marketing: Help marketers discover distinct groups in their customer bases, and then use this
* knowledge to develop targeted marketing programs
* City-planning: Identifying groups of houses according to their house type, value, and geographical
* location
* Earth-quake studies: Observed earthquake epicenters should be clustered along continent faults
* Web Search: Clustering can be used to organize the search results into groups and present the
* results in a concise and easily accessible way.
* Information Retrieval: Cluster documents into topics.

### Pre-Processing

* Summarization:
  + Preprocessing for regression, PCA, classification, and association analysis
* Compression:
  + Image processing: vector quantization
* Finding K-nearest Neighbors
  + Localizing search to one or a small number of clusters
* Outlier detection
  + Outliers are often viewed as those “far away” from any cluster. This can be used for things like credit card fraud detection (a very large transaction on an account that usually makes small transactions).

## Quality of Clustering

A good clustering algorithm will produce clusters with **high intra-class similarity** (cohesive within clusters) and **low inter-class similarity** (distinctive between clusters). Essentially, we want the clusters to be compact and also far away from each other.



The quality of clustering depends on:

* The **similarity measure** used by the algorithm
* The **implementation** of the algorithm
* The ability of the algorithm to discover some or all of the **hidden patterns**

### Similarity Metric

One of the most obvious ways to calculate the similarity between two data points is to measure the **distance** between them. However, not every feature we will be dealing with has a numeric value. This makes calculating the distance between two data points based on those features a little difficult, so we need some special methods to deal with those.

Additionally, we might also want to add some **weight** to specific features, ones we consider to be of greater importance than the others.

How good the clustering was is a separate measure entirely, usually done by some **quality function**. It is difficult to define what exactly we mean by ‘good’, so the answer to this is usually highly **subjective**.

## Considerations

There are a few things we need to keep in mind when performing cluster analysis:

* **Partitioning Criteria** – We need to consider whether the criteria we are partitioning based on is **single-level** (e.g. a news article belonging to either sports, or politics, etc.) or **hierarchical** (e.g. a news article firstly being a sports article and also being an article for a specific sport).
* **Separation of Clusters** – The clusters may be separated such that each data point belongs to a **single class** (exclusive) or to **multiple classes** (non-exclusive).
* **Similarity Measure** – The similarity of two data points may be based on **distance** or on **connectivity**. We will study the details of this soon.
* **Clustering Space** – Clusters can be created based on the complete feature space (**full space**), considering all the features, or based on **subspaces**, where only a few features are considered. Subspace clustering is done for large feature spaces.

We will mostly be concentrating on single-level, exclusive, distance-based, full-space clustering. Specifically, we will be dealing with the **K-Means** clustering algorithm and the **K-Medoids** clustering algorithm. Here, the stands for the number of clusters that the data will be divided into.

## Partitioning Algorithms

We will attempt to divide the data points into clusters such that the following equation is **minimized**:

i.e. for each class from to , we calculate the distance between each point in that cluster and the center of the cluster, .

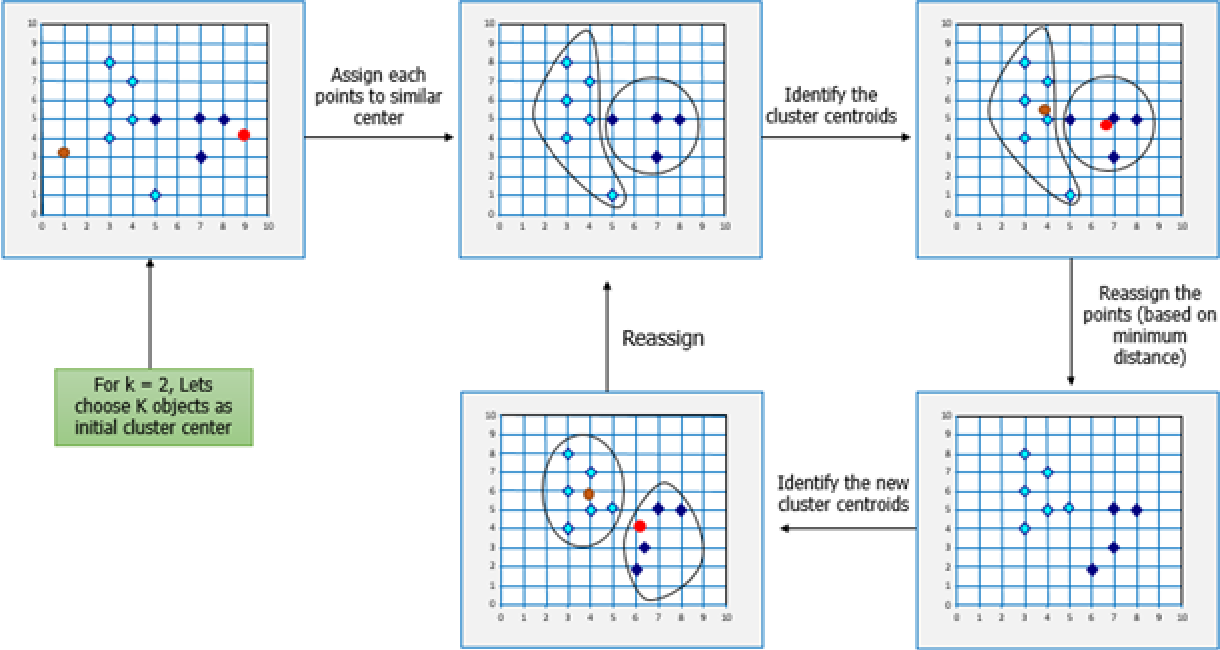
## K-Means Clustering Method

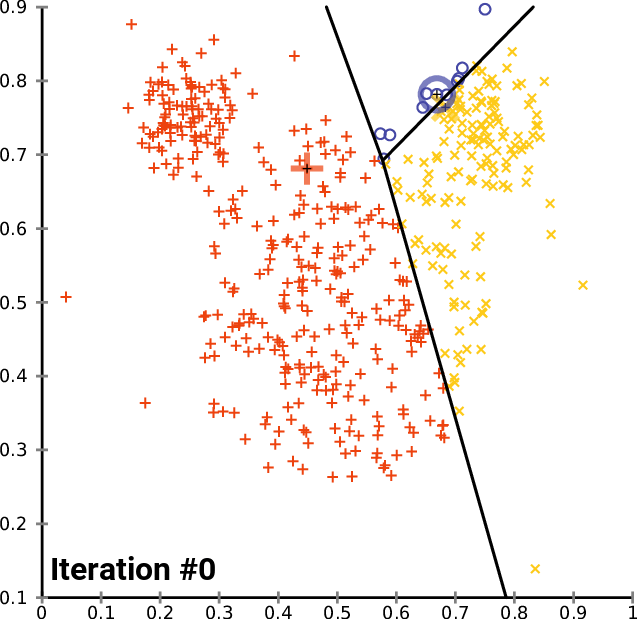
In the **K-Means Clustering Method**, the center of each cluster is represented by the **mean value** of the objects in the cluster. It takes two inputs, , the set of data points containing objects, and , the number of clusters to divide the objects into.

The method goes as follows:

1. Arbitrarily choose k objects from D as the initial cluster centers.
2. Repeat until there is no further change:
   1. (Re)assign each object to the cluster to which the object is most similar based on the mean value of the objects in the cluster.
   2. Update the cluster mean value for each cluster.

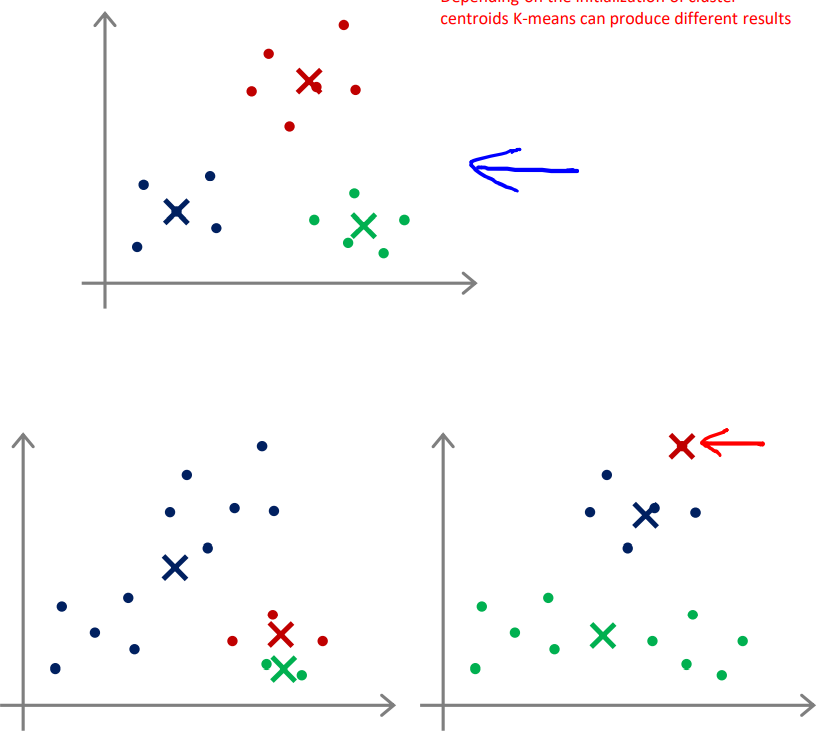
This gives us a set of clusters as the output.





### Local Optima

One issue with this approach is that, depending on the initial point chosen as the mean, we can end up with wildly different clusters:



Despite this issue, we will quickly figure out that the top-most case is the best one, since it will give us the lowest error. Thus, we can just run the algorithm times and then take the result which gave the lowest error.

### Number of Clusters

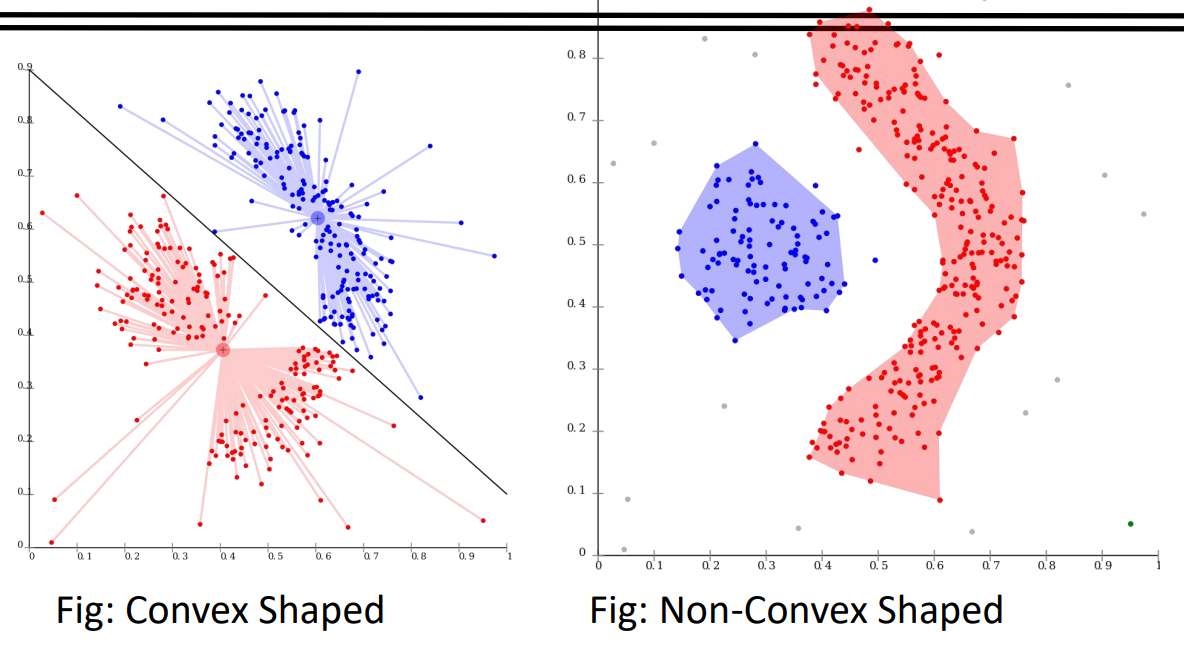
Another issue highlighted in the bottom-left image is, we have two separate clusters that are **very close together**. In cases like this, where the difference between the clusters is less than some predetermined threshold, we can consider **merging** the clusters, thus reducing the number of clusters we have to deal with. We need to be careful when doing this (i.e. deciding how many clusters there are), since it is also possible that we end up with fewer clusters than we should have.

### Strengths and Weaknesses

The main strength of the K-Means Clustering Algorithm is that it is very **efficient**. It has a time complexity of , where is the number of iterations, is the number of clusters and is the number of objects. Other clustering algorithms like PAM or CLARA have and time complexities respectively.

The weaknesses however are manyfold:

* Often terminates at a local optima
* Applicable only to objects in a continuous -dimensional space
* Must specify beforehand
* Sensitive to noise and outliers
* The algorithm is good for **convex-shaped clusters**, but not for **non-convex-shaped** ones.



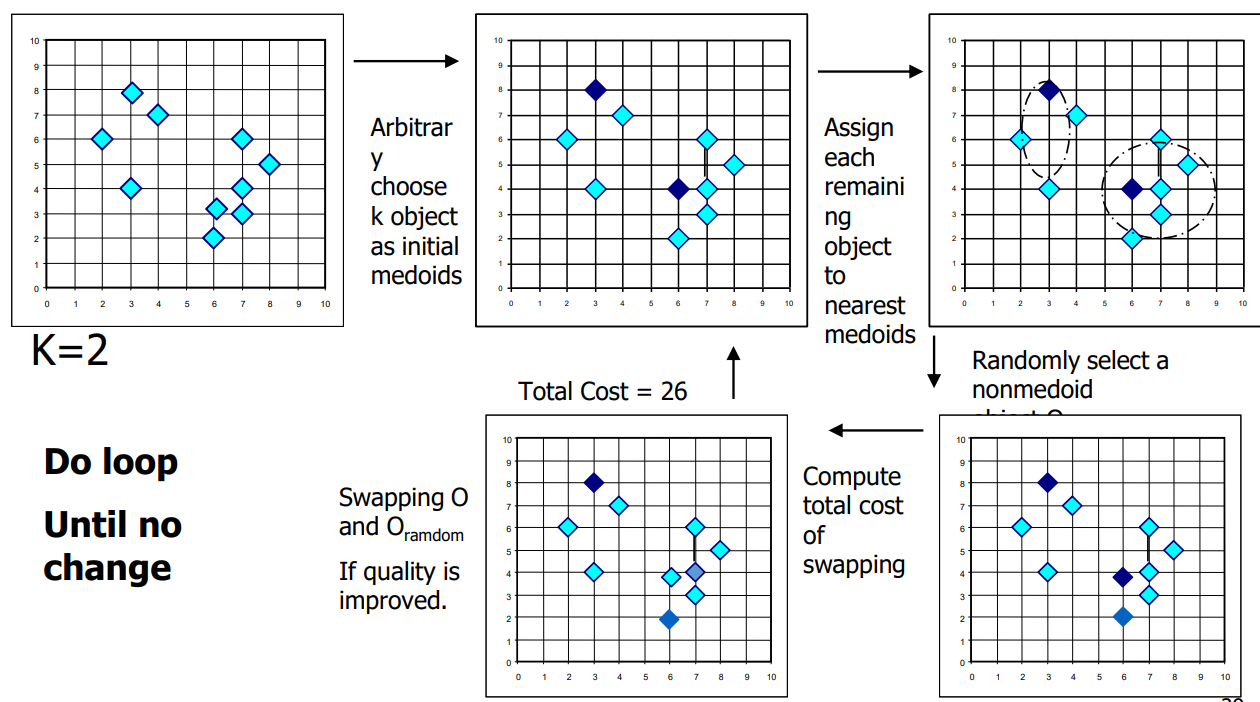
For the last point, **continuity-based** or **density-based** algorithms are able to find both convex-shaped and non-convex-shaped clusters.

## K-Medoid Clustering Algorithm

One of the issues we mentioned with the K-Means Clustering Algorithm is that it was **sensitive to outliers**. This was because it was calculating the mean value for the clusters, and having an outlier could throw off that calculation. To deal with this, the **K-Medoid Clustering Algorithm**, also called the **Partition Around Medoids** (PAM) algorithm, uses the **most centrally located object** as the centre point. This point is called the **medoid**.

The algorithm is:

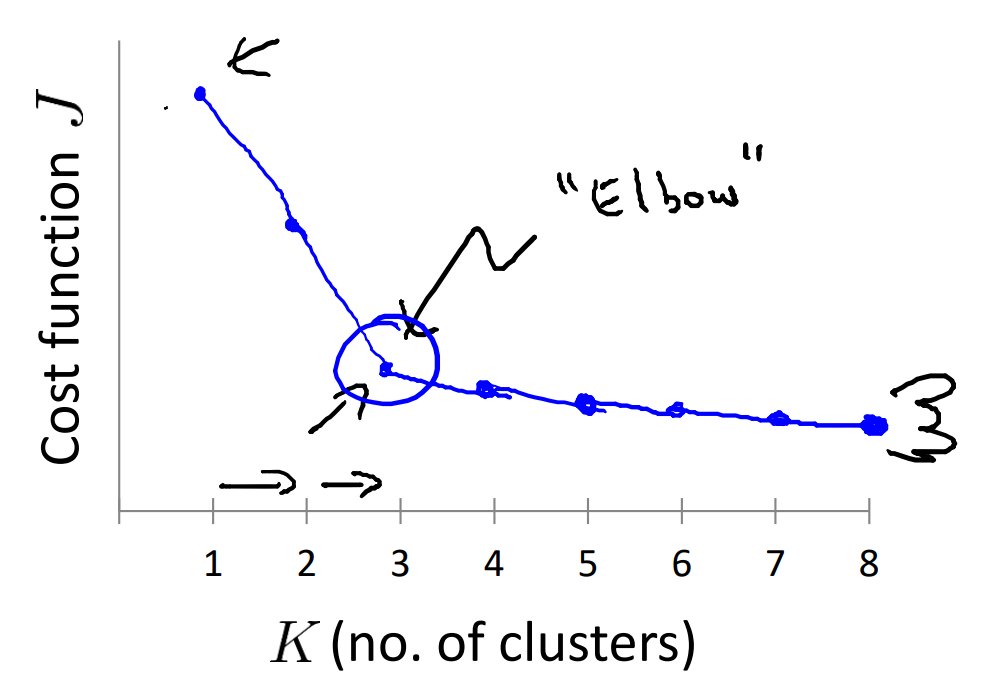
1. Randomly choose objects from as the initial representative objects (seeds).
2. Repeat until there is no further change:
   1. Assign each remaining object to the nearest cluster.
   2. Randomly select a non-representative object, .
   3. Compute the difference, , of the error if we use as the cluster center, , and the current error, (i.e. ).
   4. If , perform the swap and recompute.



For this algorithm as well, we give and as the input, the number of classes and the complete dataset respectively, and we get a set of clusters as an output.

## Choosing

The number of clusters in a dataset is very subjective. Even when looking at a dataset, different people may argue that a different number of clusters are present. To deal with this, we should try increasing values of and compare the different error values we get.



However, there is a point at which the reduction in error is **minimal**, which makes further clustering arguably unnecessary. At one point, we will be arguing that each data point is a separate cluster. To make sure we do not go too far in this direction, we can just stop at the point where we see the decrease is insignificant. The point at which the error starts to decrease slowly is called the **elbow**. Sometimes however, the decrease may be subtle, which results in there not being a clear elbow. At that point, it is up to us to figure out what degree of change is too little.

The value of might also depend on the application we are developing. For example, if the clusters correspond to T-shirt sizes, then the value of depends on whether we want to have 3 different sizes or 5 different sizes.