Chapter 1

Introduction

This Chapter presents an overview of data clustering, similarity measures, some characteristics of a good clustering method, a brief history of cluster analysis and various applications of data clustering. Further, the Chapter describes different taxonomies of the clustering methods, various clustering evaluation techniques, followed by, a brief description of existing clustering methods along with some recent trends in clustering. Motivation and scope of the proposed work, organization of the thesis are presented towards the end of the Chapter.

1.1 Overview

The ability to form meaningful groups of objects is one of the most fundamental issues of intelligence and learning. Humans perform this task with remarkable ease. In early childhood one learns to distinguish, for example, between cats and dogs or apples and oranges. However, enabling the computer to do this task of grouping automatically is a difficult and often it is an ill-posed problem. Finding such inherent structure in the data is one of the objectives of data mining. Cluster analysis is a tool for exploring the structure of the data.

1.1.1 Data clustering in general

Data clustering is a data mining technique that enables the abstraction of large amounts of data by forming meaningful groups or categories of objects, formally known as clusters, such

that objects in the same cluster are similar to each other, and those in different clusters are dissimilar. Data objects, also called patterns [1] are often described in terms of measurements (*e.g.*, attributes, features), or by relationships with other objects (*e.g.*, pairwise distance, similarity).

The data clustering problem can be formulated as follows. Given a dataset $\mathcal{D}$ of n objects, each having dimensionality d, the dataset is divided into subsets (clusters) $C_i; i = 0, 1.., k$, such that the Intra-cluster distance is minimized and the Inter-cluster distance is maximized. Figure 1.1 illustrates inter-cluster and intra-cluster distances. The quality of the produced clusters is evaluated using different external and internal quality measures (The clustering quality measures will be discussed in subsequent sections).

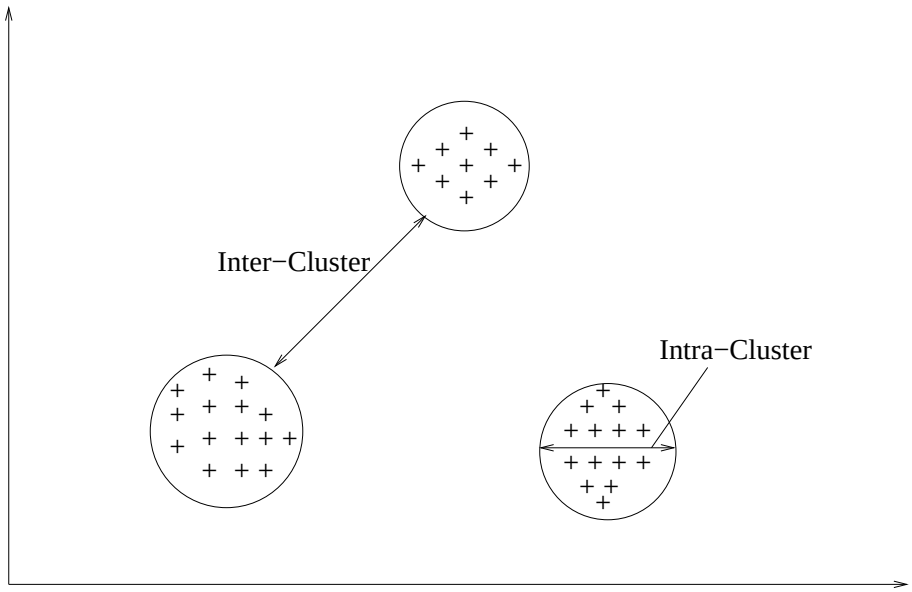
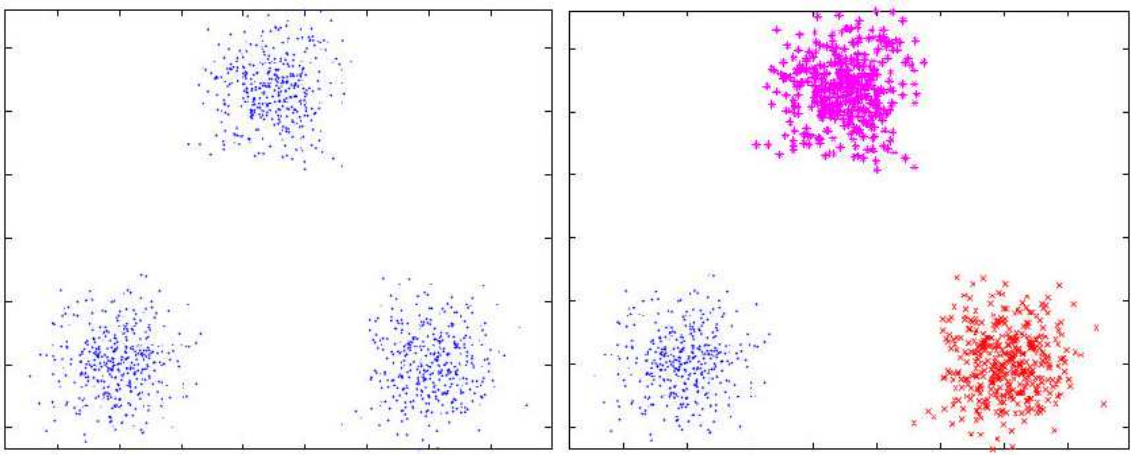


Figure 1.1: Inter-Cluster and Intra-Cluster distances

[1] The thesis uses the word "pattern" to mean a data object or data element which is basically a vector of feature-values. This is done because, much of the existing literature on clustering is from "pattern recognition" community.

Clustering is usually performed when no information is available concerning the membership of data objects to some known predefined classes. For this reason, clustering is traditionally seen as an unsupervised learning activity. An example of clustering can be seen in Figure 1.2. The dataset before clustering is shown in Figure 1.2 (a), whereas Figure 1.2 (b) shows the natural groupings that are discovered by a clustering procedure.



(a) (b)

Figure 1.2: Example for Clustering

1.1.2 Similarity measures

A key factor in the success of any clustering algorithm is the similarity measure adopted by the algorithm [1]. In order to group similar data objects, proximity metric has to be used to find which objects (or clusters) are similar. A large number of similarity metrics reported in the literature, only most of the common ones are reviewed in this subsection.

A data object (pattern) X described by a set of features, usually represented as a multidimensional vector, *i.e,* $X = (x_1, x_2, \ldots x_d)^T$. The features can be quantitative or qualitative, continuous or binary, nominal or ordinal, which determine the corresponding measure

mechanisms. Typically, distance functions are used to measure continuous features, while similarity measures are more important for qualitative variables [2].

In case of continuous features, the distance between two patterns $X = (x_1, x_2, \ldots x_d)^T$ and $Y = (y_1, y_2, \ldots, y_d)^T$ is most often based on L_1, L_2 or L_p norm:

- L_1 norm is also referred to as Manhattan or City-block distance is $\sum_{i=1}^{d} |x_i - y_i|$.
- L_2 norm is the standard Euclidean distance is $(\sum_{i=1}^{d} |x_i - y_i|^2)^{1/2}$.
- L_p norm or the Minkowski distance is a generalization of both Euclidean distance and Manhattan distance. It is $(\sum_{i=1}^{d} |x_i - y_i|^p)^{1/p}$ and it depends on the parameter p, which is a positive integer.

A more common similarity measure that is used specifically in document clustering is the *cosine correlation (Similarity)* measure [3]. It is computed as

$$\textit{CosSim}(X, Y) = \frac{X \cdot Y}{||X||||Y||}. \tag{1.1}$$

Where $(\cdot)$ indicates the vector dot product and $|| \cdot ||$ indicates the norm of the vector.

For binary features, a similarity measure is commonly used. Suppose we use two binary subscripts to count features in two objects. Let n_{00} and n_{11} represents the number of simultaneous absence or presence of features in two objects, and n_{01} and n_{10} counts the features present only in one object. Then the following measures are commonly used to find the similarity between the two data objects X and Y.

- Simple matching coefficient: $Sim(X, Y) = \frac{n_{11}+n_{00}}{n_{11}+n_{00}+n_{10}+n_{01}}$
- Jaccard coefficient: $Sim(X, Y) = \frac{n_{11}}{n_{11}+n_{10}+n_{01}}$

Various other distance and similarity measures are discussed in [4], [1], [2].

Many algorithms employ the distance function (or similarity function) to calculate the similarity between two clusters, a cluster and an object, or two objects. Calculating the distance between clusters (or clusters and objects) requires a representative feature vector of that cluster (sometimes referred to as prototype, *e.g.*, centroid or medoid). Some clustering algorithms make use of a *similarity matrix*. A similarity matrix is an $n \times n$ matrix recording the distance (or degree of similarity) between each pair of objects. Obviously the similarity matrix is a positive definite symmetric matrix so we only need to store the upper right (or lower left) portion of the matrix.

1.1.3 Characteristics of a good clustering algorithm

A specific method can perform well on one dataset, but very poorly on another, depending on the problem domain, size and dimensionality of the data as well as on the objective function used. The following are typical requirements of a clustering algorithm in data mining [4].

- **Scalability**: The ability of the algorithm to perform well with a large number of data objects.
- **Analyze mixture of attribute types**: The ability to analyze dataset with mixtures of attribute types, as well as homogeneous ones.
- **Find arbitrary-shaped clusters**: Different types of algorithms will be biased toward finding different types of cluster structures/shapes and it is not always an easy task to determine the shape or the corresponding bias. Especially when categorical attributes are present, it may not be relevant to talk about cluster structures.

- **Minimal requirements for input parameters**: Many clustering algorithms require some user-defined parameters, such as the number of clusters, in order to analyze the data. However, with large datasets and higher dimensionality, it is desirable that a method requires only limited guidance from the user, in order to avoid biasing the result.

- **Handling of noise**: Clustering algorithms should be able to handle deviations, in order to improve cluster quality. Deviations are defined as data objects that depart from generally accepted norms of behavior and are also referred to as outliers. Deviation detection is considered as a separate problem.

- **Insensitivity to the order of input records**: The same data set, when presented to certain algorithms in different orders, may lead to dramatically different clusterings. The order of input mostly affects algorithms that perform only single scan over the data set, leading to locally optimal solutions at every step. Thus, it is crucial that algorithms to be insensitive to the order of input.

- **High dimensionality of data**: The number of attributes/dimensions in many data sets is large, and many clustering algorithms can produce meaningful results only when the number of dimensions is small. The appearance of a large number of attributes is often termed as the curse of dimensionality.

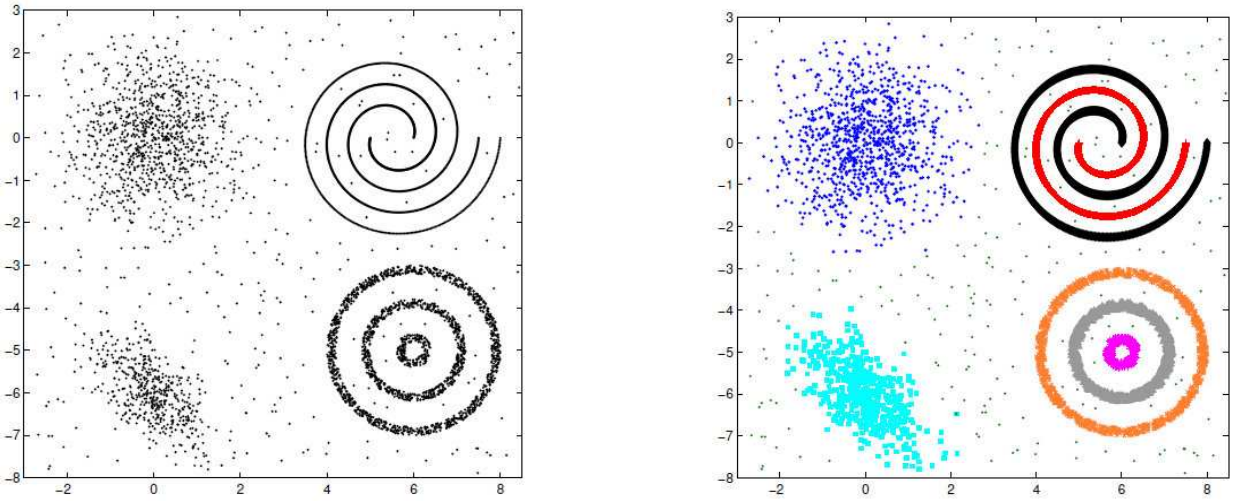
- **Interpretability and usability**: Most of the time, it is expected that clustering algorithms produce usable and interpretable results. But when it comes to comparing the results with preconceived ideas or constraints, some techniques fail to be satisfactory. Therefore, easy to understand results are highly desirable.

1.1.4 Clustering as an ill-posed Problem

Recently, A.K Jain *et.al.,* [5] stated that, clustering is inherently an ill-posed problem. This is because, there is no unique solution for any clustering problem *i.e.*, any given set of points can be clustered in drastically different ways, with no clear criterion available to prefer one clustering over another. Moreover, one clustering algorithm leads to different clustering results, based on its input parameters. However, because of the practical importance in various applications, many clustering methods have been proposed.

Vigorously growing amounts of data and data of various formats *viz.,* text, audio, video, web documents *etc.,* has been a driving force for making clustering a highly active research area. Different types of clustering methods have been proposed to cater these challenges. As increasingly complex and large data are common in current applications, it is more problematic to select a clustering criterion that leads to extract meaningful cluster structures. Clusters may have complex shapes, highly unbalanced sizes, different densities, and possible overlaps. For instance, clusters of complex shapes and of different sizes are shown in Figure 1.3. Figure 1.3(a) shows the original data and Figure 1.3(b) shows the desired clustering result. Furthermore, most current data collections are from high dimensional feature spaces.

The clustering method and other parameters like number of clusters, distance or similarity measure between data points, depends on the problem domain, and there is not even a single best known method suitable for all problem domains. Search for a unified theory behind clustering methods has recently revealed that there is no clustering function which simultaneously satisfies simple and obvious axioms *viz., scale-invariance, richness and consistency* [6]. Subsequently it is argued that, *clustering quality measure (CQM)* is a better



(a) (b)

Figure 1.3: (a) Data, (b) Desired clustering result

criterion to capture the theory behind the clustering methods [7].

Despite of these theoretical hurdles, because of its practical importance, several clustering methods have been found from classical k-means method [8] to kernel and spectral methods [9] [10], from hard partitioning methods to soft [11], [12], [2], fuzzy and rough partitioning methods [13], [14], [15], [16] etc.

1.1.5 A brief history of cluster analysis

According to the scholarly journal archive JSTOR, the word “cluster” was first used in its general sense to denote a group by Bartram *et.al.*, in 1739 [12]. The first hierarchical clustering method was developed by biologists to create hierarchy of various species for analyzing their relationship systematically [17]. Single-link clustering [18], complete-link clustering [19], and average-link clustering [20] first appeared in 1957, 1948, and 1958, respectively. Ward’s method [21] was proposed in 1963. Partitional clustering, on the other

hand, is closely related to data compression and vector quantization. The most popular partitional clustering algorithm, k-means [8], was proposed by J. MacQueen in 1967 and later an iterative approach to minimize the objective criterion of k-means method was proposed by Lloyd [22]. Two well-known versions of k-means in Pattern Recognition literature are ISODATA [23] proposed by Ball and Hall, and FORGY [24] proposed by Forgy. In 1971, Zahn proposed a graph-theoretic clustering method [25], which is closely related to single-link clustering. The first spectral clustering algorithm was proposed by Shi and Malik [26] in 1997. A summary of the important results in spectral graph theory can be found in [27]. The EM algorithm [28], which is the standard algorithm for estimating a finite mixture model for mixture-based clustering, is attributed by Dempster *et.al.,* in 1977. Interest in mean-shift clustering [29] was revived in 1995 by Cheng. Comaniciu and Meer further popularized it in 2002 [30]. Fischer and Buhmann modified the idea of connectedness in single-link method that led to path-based clustering [31].

The emergence of data mining leads to a new line of clustering research that emphasizes efficiency when dealing with huge database. DBSCAN by Ester *et. al.,* [32] for density-based clustering and CLIQUE by Agrawal *et al.,* [33] for subspace clustering are two well-known algorithms in this community. Incremental clustering methods are designed to operate in a single pass over the data points to improve the speed and scalability of data clustering. The COBWEB system [34] is an incremental conceptual clustering algorithm. The leaders clustering method [35] is another simple and popular incremental clustering method which produces a partition of a dataset in linear time with respect to the size of the dataset.

Along with the clustering techniques that produce hard or crisp clustering, which means that each object is assigned to only one cluster, fuzzy clustering methods are developed,

wherein this restriction is relaxed. That is, in fuzzy clustering the object may belong to all of the clusters with a certain degree of membership [36]. This is particularly useful when the boundaries among the clusters are not well separated and ambiguous. FCM(Fuzzy C-Means) [37] is one of the most popular fuzzy clustering algorithms in 1990s. FCM can be regarded as a generalization of ISODATA [38] and was studied by Bezdek [39]. FCM variants were also developed to deal with other data types, such as symbolic data [40] and data with missing values [41]. Recently, Rough set theory [42] became quite prominent to explore uncertainties. Lingras *et. al.,* [15] presented Rough k-means in 2004, which was later improved by Peters *et.al.,* [43] in 2006. Many clustering methods have been evolved which use the fuzzy set and rough set theories in the recent years [13], [16], [14].

Kernel based methods have been proposed to identify non-isotropic and linearly inseparable clusters in the input space [44], [45], [46]. These techniques transform the given data in the input space to a high dimensional feature space called the induced space and find the clustering in that space. The kernel trick, arising from Mercers theorem proposed in [47] is used to compute the distance between the pattens in the kernel induced feature space. Schlkopf *et.al.,* [45] depicted a kernel k-means algorithm in the online mode. Dhillon *et. al.,* [48] presented a unified view of the kernel and spectral methods.

Hybrid clustering methods combine two or more methods where the result of one method is given as input to the subsequent method. *l*-DBSCAN [49], *Rough*-DBSCAN [16], *Leader-single-link* [50] are some examples of this type. Ensemble based clustering methods [51] are inspired by the ensemble of supervised learning methods [52]. These hybrid and ensemble methods have been used not only to improve the clustering quality but also to speed-up the clustering process.

The literature on cluster analysis is vast, and hundreds of clustering algorithms have been proposed in the literature. It requires a tremendous effort to list and summarize all the major clustering algorithms. A detailed study on some other recent trends in clustering is available in [2], [9], [53], [54].

1.1.6 Applications of clustering

Data clustering is prevalent in any discipline that involves analysis of multivariate data. Clustering has been widely used in various application domains which include artificial intelligence, pattern recognition, machine learning, information retrieval, image processing, biology, data mining, marketing, medicine, psychology, recommender systems and statistics.

In image processing, it is used to segment texture in images to differentiate between various regions or objects [55], [56]. It is also used for data compression in image processing, which is also known as *vector quantization* [57]. In computer vision, several important problems can be formulated as clustering problems [58].

Clustering has always been used in statistics [59] and science [60]. The classic introduction into pattern recognition framework is given in [61]. Typical applications include speech [62] and character recognition [63]. For statistical approaches to pattern recognition see [64], [65]. Documents can be clustered [66] to generate topical hierarchies for information access or retrieval [67]. Clustering is also used to perform market analysis [68], [69].

Clustering in data mining was brought to life by intense developments in text mining [70], [71], [72], spatial database applications, such as, GIS or astronomical data [73], [74], sequence and heterogeneous data analysis [75], Web applications [76], DNA analysis in computational biology [77], and many others.

1.2 Different taxonomies of data clustering algorithms

Different starting points and criteria usually lead to different taxonomies of clustering algorithms [12], [78], [11], [79]. Clustering algorithms can be classified along different independent dimensions. For instance, different starting points, methodologies, algorithmic point of view, clustering criteria, and output representations, usually lead to different taxonomies of clustering algorithms [80]. Different properties of clustering algorithms can be described as follows:

- **Agglomerative vs. Divisive Clustering**: This concept relates to algorithmic structure and operation. An agglomerative approach begins with each object in a distinct (singleton) cluster, and starts merging clusters together until a stopping criterion is satisfied (bottom-up hierarchical clustering). On the other hand, a divisive method begins with all objects in a single cluster and iteratively performs splitting until a stopping criterion is met (top-down hierarchical clustering).

- **Monothetic vs. Polythetic Clustering**: Both the monothetic and polythetic issues are related to the sequential or simultaneous use of features in the clustering algorithm. Most algorithms are polythetic; that is, all features enter into the computation of distances (or similarity functions) between objects, and decisions are based on those distances, whereas, a monothetic clustering algorithm uses the features one by one.

- **Hard vs. Fuzzy Clustering**: A hard clustering algorithm allocates each object to a single cluster during its operation and outputs a Boolean membership function either 0 or 1. A fuzzy clustering method assigns degrees of membership for each input object

to each cluster. A fuzzy clustering can be converted to a hard clustering by assigning each object to the cluster with the largest degree of membership.

- **Distance vs. Density Clustering**: A distance-based clustering algorithm assigns an object to a cluster based on its distance from the cluster or its representative(s), whereas a density-based clustering grows a cluster as long as the density (or number of objects) in the neighborhood satisfies some threshold. Distance-based clustering algorithms fails at discovering clusters of arbitrary shape, whereas density-based clustering algorithms are capable of finding arbitrary shape clusters.

- **Partitional vs. Hierarchical Clustering**: A Partitional clustering algorithm obtains a single partition of the data instead of a clustering structure, such as the dendogram produced by a hierarchical technique. Partitional methods have advantages in applications involving large data sets for which the construction of a dendogram is computationally prohibitive.

- **Deterministic vs. Stochastic Clustering**: This issue is most relevant to Partitional techniques designed to optimize a squared error function. Deterministic optimization can be accomplished using traditional techniques in a number of deterministic steps. Stochastic optimization randomly searches the state space consisting of all possible solutions.

- **Incremental vs. Non-incremental Clustering**: This issue arises when the objects set to be clustered is large, and constraints on execution time or memory space need to be taken into consideration in the design of the clustering algorithm. Incremental clustering algorithms minimize the number of scans through the objects set, reduce

the number of objects examined during execution, or reduce the size of data structures used in the algorithms operations. Also incremental algorithms do not require the full data set to be available beforehand. New data can be introduced without the need for re-clustering.

- **Intermediate vs. Original Representation Clustering**: Some clustering algorithms use an intermediate representation for dimensions reduction when clustering large and high dimensional datasets. It starts with an initial representation, considers each data object and modifies the representation. These classes of algorithms use one scan of the dataset and its structure occupies less space than the original representation of the dataset, so it may fit in the main memory.

1.3 Clustering evaluation criteria

The previous subsection has reviewed different types of clustering algorithms proposed in the literature. However, different clustering algorithms, or even a single clustering algorithm using different parameters, generally gives different clustering results. Therefore, it is necessary further evaluation of these resulting clustering.

Cluster validation is the process of assessing the quality and reliability of the clusters derived from various clustering processes [80]. Generally, cluster validity has two aspects: (1) first, the quality of clusters can be measured in terms of *homogeneity* and *separation* on the basis of the definition of a cluster: objects within one cluster are similar to each other, while objects in different clusters are dissimilar. Thus if the data is not previously classified, *internal quality measures* are used to compare different sets of clusters without reference

to external knowledge, and (2) the second aspect relies on a given "ground truth" of the clusters. The "ground truth" could come from domain knowledge, such as known function families of objects, or from other knowledge repositories (*e.g.*, such as the clinical diagnosis of normal or cancerous tissues for gene expression datasets). Thus, cluster validation is based on the agreement between clustering results and the "ground truth". Consequently, the evaluation depends on a prior knowledge about the classification of data objects, *i.e.*, class labels. This labeling is used to compare the resulting clusters with the original classification; such measures are known as *external quality measures* [81].

1.3.1 External quality measures

This subsection reviews some external quality measures which assume that a prior knowledge about the data objects (*i.e.*, class labels) is given [82].

- **F-measure**:

  One external measure is the *F*-measure, a measure that combines the *Precision* and *Recall* ideas from the literature of information theory. The precision and recall of a cluster C_j with respect to a class R_i, where $i, j = 1, 2, \ldots, k$ are defined as:

$$\textit{Precision}(R_i, C_j) = \frac{L_{ij}}{|C_j|} \tag{1.2}$$

$$\textit{Recall}(R_i, C_j) = \frac{L_{ij}}{|R_i|} \tag{1.3}$$

  Where L_{ij} is the number of objects of class R_i in cluster C_j, $|R_i|$ is the number of

objects in class R_i and $|C_j|$ is the number of objects in cluster C_j.

The *F*-measure of a class R_i is defined as:

$$F(R_i) = \max_j \left\{ \frac{2 \times \textit{Precision}(R_i, C_j) \times \textit{Recall}(R_i, C_j)}{\textit{Precision}(R_i, C_j) + \textit{Recall}(R_i, C_j)} \right\} \tag{1.4}$$

With respect to class R_i we consider the cluster with the highest *F*-measure to be the cluster C_j that is mapped to class R_i, and that *F*-measure becomes the score for class R_i. The overall *F*-measure for the clustering result of k clusters is the weighted average of the *F*-measure for each class R_i:

$$\textit{F-measure(k)} = \frac{\sum_{i=1}^{k} (|R_i| \times F(R_i))}{\sum_{i=1}^{k} |R_i|} \tag{1.5}$$

The higher the *F*-measure the better the clustering due to the higher accuracy of the resulting clusters mapped to the original classes.

- **Entropy**:

  The second external quality measure is the Entropy. It tells us how homogeneous a cluster is. The higher the homogeneity of a cluster, the lower the entropy is, and vice verse. The entropy of a cluster containing only one object (perfect homogeneity) is zero. Assume a partitioning result of a clustering algorithm consisting of k clusters. For every cluster C_j we compute pr_{ij}, the probability that a member of cluster C_j belongs to class R_i. The entropy of each cluster C_j is calculated using the following

standard formula, where the sum is taken over all classes:

$$E(C_j) = -\sum_{i=1}^{k} pr_{ij} \log(pr_{ij}), j = 1,2,\ldots,k \tag{1.6}$$

The overall entropy for a set of k clusters is calculated as the sum of entropies for each cluster weighted by the size of each cluster.

$$Entropy(k) = \sum_{j=1}^{k} \left(\frac{|C_j|}{n} \right) \times E(C_j) \tag{1.7}$$

Where $|C_j|$ is the size of cluster C_j, and n is the total number of objects. As mentioned earlier, we would like to generate clusters of lower entropy, which is an indication of the homogeneity (or similarity) of objects within the clusters. The overall weighted entropy formula avoids favoring smaller clusters over larger clusters.

- **Purity**:

  The Purity of a clustering solution is the average precision of the clusters relative to their best matching classes. For a single cluster C_j, Purity is defined as the ratio of the number of objects in the dominant cluster to the total number of objects in the cluster:

$$P(C_j) = \frac{1}{|C_j|} \max_{i=1,2\ldots,k.\ i \neq j} (L_{ij}) \tag{1.8}$$

  Where L_{ij} is the number of objects from class R_i into cluster C_j, and $|C_j|$ is the number of objects in cluster C_j. To evaluate the total purity for the entire k clustering, the

cluster-wise purities are weighted by the cluster size and the average value is calculated:

$$\textit{Purity}(k) = \frac{1}{n}\sum_{j=1}^{k} \max_{i=1,2,...,k.\ i\neq j}(L_{ij}) \tag{1.9}$$

Higher values of the Purity measure which indicate better partitioning of objects.

- **Rand-Index**:

  An alternative to the information-theoretic interpretation of clustering is to view it as a series of decisions, one for each of thepairs of data points in the collection. We want to assign two data points to the same cluster if and only if they are similar. A true positive (TP) decision assigns two similar data points to the same cluster, a true negative (TN) decision assigns two dissimilar data points to different clusters.

  There are two types of errors we commit. False positive (FP) decision assigns two dissimilar data points to the same cluster and false negative (FN) decision assigns two similar data points to two different clusters. The Rand-Index (RI) measures the percentage of decisions that are correct, and it is defined as follows.

$$RI = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1.10}$$

Some other external quality measures, such as, Adjusted Rand-Index, Normalized Mutual Information (NMI), Jaccard Coefficient, Variation Information (VI), Rand Static (R) *etc.*, are discussed in [83], [84] [82]. The thesis uses the Rand-Index as the clustering quality measure.

1.3.2 Internal quality measures

Internal quality measures are used to verify whether the clustering structure produced by a clustering algorithm fit the data, but using only information inherent to the dataset. This subsection reviews some internal quality measures.

- **Dunn Index (DI)**: An internal quality index for crisp clustering that aims at the identification of "compact" and "well separated" clusters. The Dunn Index is defined in the following equation for a specific number of k clusters.

$$DI(k) = \min_{i=1,2,\ldots,k} \left\{ \min_{j=1,2,\ldots,k.\ i \neq j} \left\{ \frac{D(C_i, C_j)}{\max_{r=1,2,\ldots,k} diam(C_r)} \right\} \right\}. \tag{1.11}$$

  where the dissimilarity function between two clusters C_i and C_j is

$$D(C_i, C_j) = \min_{X \in C_i, Y \in C_j} \{||X - Y||\}$$

  and *diam*(C_r) is the diameter of cluster C_r, which may be considered as a measure of clusters' dispersion. If the dataset contains compact and well-separated clusters, the distance between clusters is expected to be large and the diameter of the cluster is expected to be small, thus large values of DI indicate the presence of compact and well-separated clusters.

  The problems with the Dunn Index are (1) its considerable time complexity for large n, and (2) its sensitivity to the presence of noise in the dataset, since these are likely to increase the values of the *diam(C)*.

- **Separation Index (SI)**:

  Separation Index is a cluster validity measure that utilizes cluster prototypes to measure the dissimilarity between clusters, as well as between objects in a cluster to their respective cluster prototypes.

$$SI(k) = \frac{\sum_{i=1}^{k} \sum_{\forall X_j \in C_i} ||X_j - m_i||}{n^* \min_{r,l=1,2,\ldots,k.\ r \neq l} ||m_l - m_r||} \tag{1.12}$$

  Where $||X - m_i||^2$ is the Euclidean distance between object X and the cluster prototype m_i. For centroid-based clustering, m_i is the corresponding centroid of the cluster C_i while in medoid-based clustering, m_i refers to the medoid of the cluster C_i. Clustering solutions with more compact clusters and larger separation have lower separation index, thus lower values indicates better solutions. The index is more computationally efficient than Dunns index, and is less sensitive to noisy data.

- **Cluster Distance Index (CD)**:

  The CD index measures the distance between the two clusters that are merged in a given step of a hierarchical clustering algorithm. This distance depends on the selected representatives for the hierarchical clustering performed. For instance, for centroid-based hierarchical clustering the representatives of the formed clusters are the centroids of each cluster, so CD index is the distance between the centroids of the clusters as shown in the following equation.

$$CD(C_i, C_j) = ||m_i - m_j||^2 \tag{1.13}$$

Where C_i and C_j are the merged clusters and m_i and m_j are the centroids of the clusters C_i and C_j, respectively. In the *Single linkage* (SL) hierarchical clustering, the CD index is defined as the minimum Euclidean distance between all possible pairs of points.

$$CD(C_i, C_j) = \min_{\forall X \in C_i,\, \forall Y \in C_j} ||X - Y||^2 \tag{1.14}$$

The *Complete linkage* (CL) hierarchical clustering defines the CD index as the maximum Euclidean distance between all pairs of data points.

$$CD(C_i, C_j) = \max_{\forall X \in C_i,\, \forall Y \in C_j} ||X - Y||^2 \tag{1.15}$$

Some other Internal Quality Measures include, *Silhouette Coefficient*, R-Square measure, Root Mean Square Standard Deviation (RMSSTD) Index, Bayesian Information Criterion, Partition Entropy, Xu-Index, Hartigan, Ball&Hall index, Means Square Error (MSE), Sum of Squares Within Clusters (SSW), Sum of Squares Between Clusters (SSB) *etc*. A detailed study on these measures is presented in [85], [84].

1.4 Existing clustering methods: A short review

Clustering methods can be broadly categorized into hierarchical clustering and partitional clustering methods, as shown in Figure 1.4. The other categories of clustering methods include density-based methods, grid-based methods, model-based methods *etc*. Hybrid and Ensemble schemes are the new trends in data clustering. This section outlines some of the

existing clustering techniques in each category.

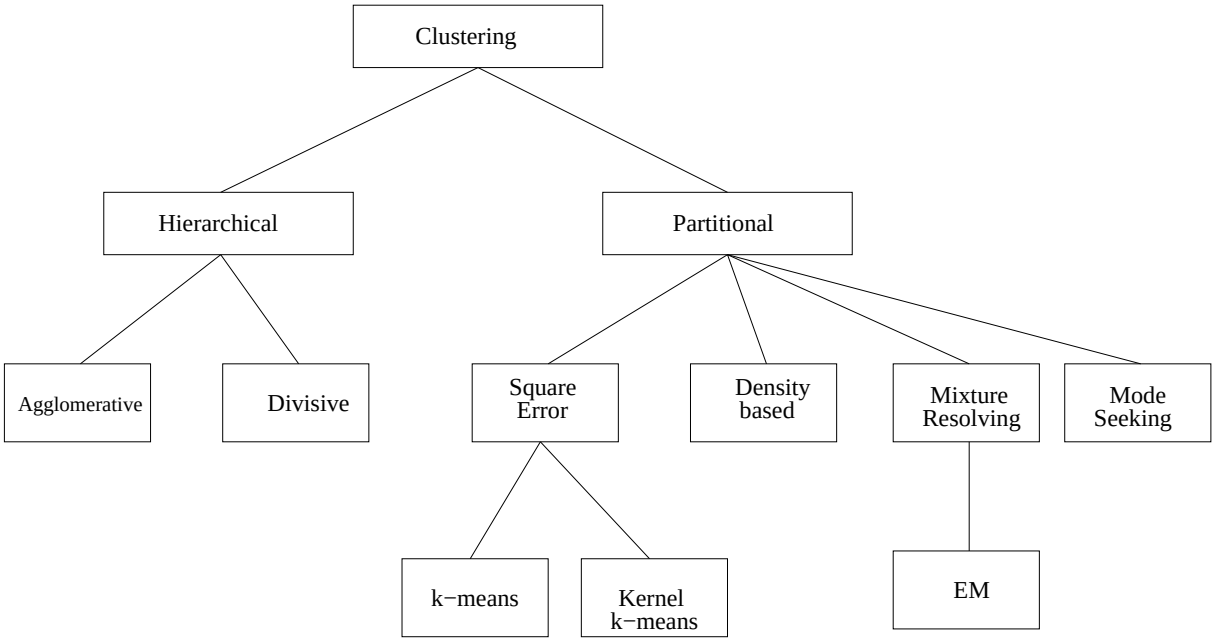


Figure 1.4: A Possible Taxonomy of Clustering Algorithms

1.4.1 Hierarchical methods

Hierarchical clustering algorithms generate a tree of nested clusters represented by a dendogram. A hierarchical method can be classified as being *agglomerative or divisive* method, based on how the hierarchical decomposition is formed. In agglomerative approach, each pattern is considered as individual cluster at the initial step and the clusters which are more similar are *merged* into one cluster in the next step and this procedure will be repeated until all patterns are grouped into a single cluster. Divisive methods start with a single cluster with all pattern in it, this cluster is *split* in to small clusters and the process of splitting is repeated till each pattern forms an individual cluster [4]. Figure 1.5 illustrates both the agglomerative and divisive approaches in hierarchical clustering methods. Two divisive clustering

algorithms, named MONA and DIANA, are described in [84].

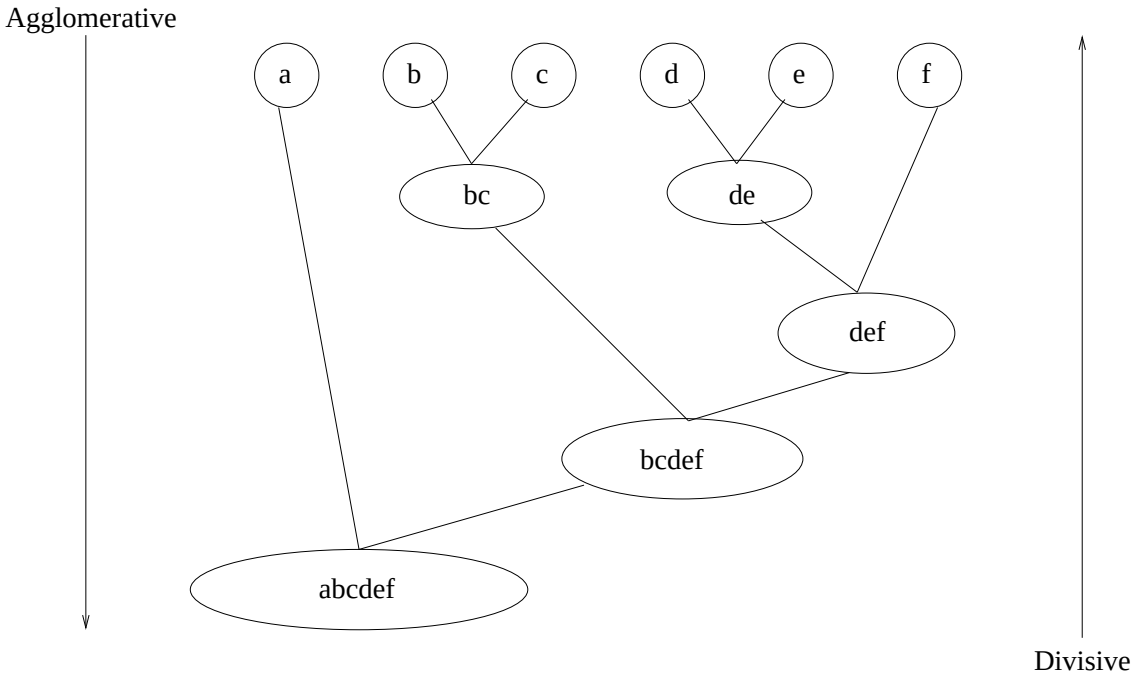


Figure 1.5: Agglomerative and Divisive Hierarchical Clustering

Similarity plays main role in the process of splitting and merging. The similarity between two clusters can be measured in different ways. Majority of agglomerative clustering methods use the minimum distance d_{min} as the distance measure. If C_i and C_j are the two clusters, then $d_{min}(C_i, C_j) = \min\{||X_i - X_j||\}$, where X_i and X_j are patterns that belong to the clusters C_i and C_j respectively. Two clusters to be merged are those which are at a smaller d_{min} distance. *Single-link* clustering method [18] uses this distance to find the dendogram of clusterings. Other similar methods are *average-link* [19] and *complete-link* [20] methods which differ from the single-link by the distance method by which the distance between two clusters is found. In the case of average-link, the distance between clusters C_i and C_j is the average distance between a pattern from C_i to a pattern in C_j. Whereas, for complete-link, the distance is the maximum distance between a pair of patterns X_i and X_j where X_i is from

C_i and X_j is from C_j. Single-link is good at handling clusters of non-elliptical shapes, but is sensitive to noise and outliers. The complete-link method is less susceptible to noise and outliers, but can break large clusters, and has trouble with convex shaped clusters. But in both approaches fixing merge or split criterion is a critical problem and the process of merging or splitting is never undone. Some other agglomerative clustering algorithms include *group average-linkage, median-linkage, centroid-linkage* and *Ward's method* [78].

BIRCH uses a hierarchical data structure called CF-tree for partitioning the data in an incremental way. CF-tree is a height-balanced tree, which stores the clustering features and it is based on two parameters, *viz.,* branching factor B and threshold τ, which refer to the diameter of a cluster. A CF-tree is built as the data is scanned. The CF-tree structure captures the important clustering information of the original data while reducing the required storage. After the CF-tree is built, an agglomerative hierarchical clustering method is applied to perform the global clustering. BIRCH typically finds a good clustering with a single scan of the data and improves the quality further with a few additional scans. It also handles noisy outliers effectively. Outliers are eliminated from the summaries by identifying the objects sparsely distributed in the feature space. BIRCH has a computational complexity of $O(n)$ where n is the dataset size. However, it has one drawback, *viz.,* it may not work well when clusters are not "spherical" because it uses the concept of radius or diameter to control the boundary of a cluster. Further, it is order-sensitive as it generates different clusters for different scanning orders of the same input data.

Guha *et al.,* [86] proposed an agglomerative hierarchical clustering algorithm, called CURE (Clustering Using REpresentatives) which can find arbitrary shaped clusters. In

CURE, instead of using a single centroid to represent a cluster, a constant number of representative points are chosen to represent a cluster. Unlike centroid/medoid based methods, CURE is capable of finding clusters of arbitrary shapes (*e.g.* ellipsoidal, spiral, cylindrical, non-convex) and of arbitrary sizes. Shrinking the representative points towards the centroid helps CURE in avoiding the problem of noise. CURE utilizes random sample (and partition) strategy to reduce its computational cost. ROCK (RObust Clustering using linKs) is another agglomerative clustering algorithm for categorical and boolean data which uses the *Jaccard coefficient* to measure similarity between data items. ROCK constructs a sparse graph from the given data using a similarity threshold and the concept of shared neighbors. It then performs agglomerative clustering over the sparse graph [87].

CHAMELEON [88] is a hierarchical agglomerative algorithm which utilizes dynamic modeling in cluster aggregation. The algorithm is based on the k-nearest-neighbor graph. It finds the clusters in the dataset by using a two-phase algorithm. In the first step it generates the k-nearest neighbor graph that contains only links between a pattern and its nearest neighbors. Later, CHAMELEON uses a graph-partitioning algorithm to cluster the data items into a large number of relatively small sub-clusters. During the second phase, it uses agglomerative clustering algorithm to obtain a clustering by repeatedly combining these sub-clusters. The algorithm does not depend on assumptions about the data model. The computational cost of the method is quadratic with respect to the size of the dataset. Some other hierarchical clustering methods are presented in [2].

1.4.2 Partitional clustering methods

Partitional clustering algorithms partition the dataset into a pre-defined number of clusters. They can further be classified into two categories based on their input parameters: algorithms which explicitly take the number of clusters k as input and algorithms that take as input, a threshold τ on the radius of the clusters which indirectly determines the number of clusters [89]. k-means [8] and PAM [84] are examples of algorithms in the first category. Algorithms like Leader [35], DBSCAN [32], *etc*., fall in the second category.

k-means clustering method is an iterative method, which partitions the dataset into k clusters such that all patterns in a given cluster are closest to the center of that cluster. Each cluster C_j is represented by its mean (centroid) M_j. The clustering has to be found such that the criterion $J = \sum_{j=1}^{k} \sum_{X_i \in C_j} ||X_i - M_j||^2$ is minimized. It is simple to implement and has linear time complexity with respect to the size of the dataset. But the quality of the clustering result is highly influenced by initial conditions and the parameter k. Further, it is inefficient in identifying non-convex and linearly inseparable clusters. Kernel k-means [90] is a nonlinear extension of the k-means clustering method, which can identify non-convex and linearly inseparable clusters. It first maps the data points into a higher dimensional feature space and tries to minimize the criterion which is the same as in k-means clustering method, but in the induced space. The k-means and kernel k-means methods will be discussed elaborately in the next chapter.

The basic k-means algorithm has been extended in many different ways. Bradley *et. al.* [91] proposed an approach to refine the initial seed points which helps in getting quick convergence. Some algorithms are proposed in [92] [93] for computing initial cluster centers.

A hybrid approach to find an initial cluster centers is given in [94]. Two well-known variants of the k-means method are ISODATA [23] and FORGY [24]. In k-means, each data point is assigned to a single cluster (called hard assignment). Dunn *et al.,* [95] proposed Fuzzy c-means wherein each data point can be a member of multiple clusters with a membership value (soft assignment). Bezdek *et.al.,* [39] proposed some other improvements over fuzzy c-means. A good overview of fuzzy set based clustering is available in [96].

The k-modes algorithm is an extension of k-means algorithm for large datasets with categorical data [97]. The k-prototypes algorithm [97], through the definition of a combined dissimilarity measure, further integrates the k-means and k-modes algorithms to allow for clustering instances described by mixed attributes. Recently, Yiu-Ming Cheung *et.al.,* [98] proposed a generalized version of the k-means method, called k^*-clustering which can find ellipse-shaped data clusters as well as ball-shaped ones without predetermining the exact number of clusters.

PAM (Partitioning Around Medoids) is similar to the k-means method [84]. PAM uses *medoid* as the cluster prototype. Medoid is the most centrally located data item of the cluster. PAM avoids the effect of outliers and noise. Finding medoid in each iteration is an expensive computation and hence PAM is not well suited for large datasets. CLARA (Clustering LARge Applications) and CLARANS (Clustering Large Applications based on RANdomized Search) are the extensions of PAM to work with large datasets. CLARA applies PAM on a sample portion of data selected as a representative of the entire dataset. So, the quality of the clustering depends on the selected sample. CLARANS shows better performance than PAM and CLARA and it selects sample with some randomness at each step of the search [99], [100].

In contrast to all the above methods, leaders clustering method [35] is a partitioning method which takes a threshold τ on the radius of the clusters, which indirectly determines the number of clusters, as input to generate a partition of the dataset. This method can be applied for large datasets because of its linear time complexity, but the result of this method heavily depends on the order of scanning of the given dataset and also on the input parameter τ. The leaders clustering method will be discussed elaborately in the next chapter.

1.4.3 Graph-based methods

Graph theoretic clustering, sometimes referred as spectral clustering, represents the data points as nodes in a weighted graph [2]. The edges connecting the nodes are weighted by their pair-wise similarity. The central idea is to partition the nodes into two subsets A and B such that the cut size, *i.e.,* the sum of the weights assigned to the edges connecting between nodes in A and B, is minimized. Initial algorithms solved this problem using the minimum cut algorithm, which often results in clusters of imbalanced sizes. A cluster size (number of data points in a cluster) constraint was later adopted by the ratio cut algorithm [101]. Shi and Malik proposed an efficient approximate graph-cut method called the Normalized Cut [55]. Meila and Shi presented a Markov Random Walk view of spectral clustering and proposed the Modified Normalized Cut(MNCut) [9] algorithm that can handle an arbitrary number of clusters. Another variant of spectral clustering algorithm was proposed by Ng *et.al.,* [102], where a new data representation is derived from the normalized eigen vectors of a kernel matrix. Dhillon *et.al.,* [48] established a unified view of kernel, spectral and graph based methods. Extensive study on graph based and spectral clustering methods can be found in [9], [103] [104].

1.4.4 Density-based methods

Density-based clustering methods can be treated as a type of partitioning algorithms. Density-based clustering methods perform clustering by identifying the high density regions in the object space. A cluster, defined as a connected dense component, grows in any direction that density leads. Therefore, density-based algorithms are capable of discovering clusters of arbitrary shapes. Also this provides a natural protection against outliers. An excellent introduction to density-based methods can be found in the book [4].

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is one of the most well known density-based clustering algorithms [32]. A data object is called a *core* object, if the neighborhood around it within a given radius *EPSILON* has at least the minimum number of objects *MINPTS*. Otherwise, the object is a *non-core* object. A core object along with its neighboring objects forms a cluster. These clusters are expanded by merging near-by core clusters. DBSCAN is efficient in identifying arbitrary shaped clusters. But it has a quadratic time complexity, and hence is not a suitable one for large datasets. One of the ways to overcome this problem is to build an index over the data set like a R^*-tree index. But this solution is suitable only when the dimensionality of the data is low. A generalization over DBSCAN called GDBSCAN [105] can work with both numerical and categorical data. An improvement over the DBSCAN was proposed by Yesser El-sonbaty *et.al.,* [106] where the dataset partitioning is done as a pre-processing step to reduce the search space of the clustering method and detects clusters in an efficient way. More recently, Viswanath *et.al.,* [16] proposed a faster version of the DBSCAN, wherein the size of the dataset that is used with DBSCAN is reduced by selecting few prototypes of the dataset.

OPTICS (Ordering Points to Identify the Clustering Structure) proposed by Ankerst *et. al.* [107] is a cluster-ordering method that facilitates density-based clustering without worrying about parameter specification. It has same computational complexity as DBSCAN. DBCLASD(Distribution Based Clustering of Large Spatial Databases) [108] is a non-parametric approach that discovers clusters of different densities. DENCLUE (DENsity-based CLUstEring) [109] is yet another density based method that works well even on noisy data.

1.4.5 Grid-based methods

Grid-based clustering algorithms first quantize the clustering space into a finite number of cells (hyper-rectangles) and then perform the required operations on the quantized space. The advantage of grid-based methods is the fast processing time, which is independent of the size of the dataset, yet it depends on the number of cells in each dimension in the quantized space.

STING [110](STatistical INformation Grid) is a grid-based clustering algorithm, in which the spacial area is divided into rectangular cells. It generates a hierarchical structure of the grid cells so as to represent the clustering information at different levels. STING produces good clustering results in a smaller running time. The performance of STING relies on the granularity of the lowest level of the grid structure and the resulting clusters are all bounded horizontally or vertically, but never diagonally. This shortcoming might greatly affect the cluster quality.

CLIQUE [33](CLustering In QUEst) is one of the early algorithms proposed for clustering in high-dimensional space. CLIQUE generates the potential dense units in k-dimensional space from dense units found in (k-1)-dimensional space. These dense units are examined

to determine the clusters. Empirical evaluation shows that CLIQUE scales linearly with the number of instances, and has good scalability as the number of dimensions are increased.

Wave-Cluster [111] is a multiresolution clustering method that first summarizes the data by imposing a multidimensional grid structure on the data space. Then it uses a wavelet transformation on the original feature space, finding the dense regions in the transformed space. It has linear time complexity. It does not require the number of clusters as an input parameter. It is a very powerful method to work with large datasets and can find arbitrary shaped clusters. It is insensitive to the order of the input data items and insensitive to outliers. But, it is not efficient to work with high dimensional data.

1.4.6 Model-based methods

Model-based methods try to fit the data to a mathematical model. The underlying assumption is that the patterns in the dataset are generated by an underlying probability distribution. Often these distributions are mixtures of several distributions. The problem of clustering is reduced to the problem of estimating the parameters of the underlying probability distribution. Expectation-Maximization (EM) [112] is a popular model-based method. EM can be considered as an extension of the k-means clustering method [11]. EM method adopts an iterative approach to estimate the parameters, such as *mean* and *standard deviation* of the probability distributions so as to best fit the data. The process starts with the initial parameters and it iteratively refines the parameters of the clusters by calculating the expected cluster memberships for each data object and uses this estimate to re-estimate the model parameters.

SOM (*Self Organizing Map*) [113] is one of the famous neural network methods for cluster analysis. The SOM method is an unsupervised neural network mapping of the high

dimensional input data onto a usually two-dimensional output space while preserving relations between the data items. The cluster structure within the data as well as the inter-cluster similarity is visible from the resulting topology. The SOM model can be used in particular to visualize the dataset and explore its properties. COBWEB [34] is another model based clustering method. COBWEB creates a classification tree, in which each node refers to the concept and contains a probabilistic description of the concept, which summarizes the objects classified under that node.

To enumerate some other clustering methods which do not fit into the above mentioned categorization are: (i) Constraint-based clustering methods [4], (ii) Spectral methods [103], [114], (iii) kernel-based clustering methods [9], (iv) Rough and Fuzzy clustering methods [13], [15], [13], [115].

1.5 Recent trends in data clustering

This subsection summarizes two recent trends in data clustering *viz.,* ensemble based clustering methods and hybrid clustering methods [50], [116].

- **Ensemble based clustering**

  Ensemble based clustering methods are recent research advancement motivated from ensemble based classifiers [54]. Ensemble of classifiers is relatively a well studied area than ensemble of clustering methods. Ensemble clustering methods combine various clustering results, called cluster ensembles, using a *consensus function* to produce a single clustering result [116].

  Cluster ensembles can be formed in a number of different ways [51], such as, (1)

the use of a number of different clustering techniques (either deliberately or arbitrarily selected), (2) the use of a single technique many times with different initial conditions, and (3) the use of different partial subsets of features or patterns. Similarly, there are so many *consensus functions* proposed in the literature to combine these individual clusterings. Recent consensus methods may be roughly classified into proximity-based methods [51], [31], methods based on a (categorical) feature-space representation of the objects [54], [117], and methods based on voting (ensemble relabeling) as in [31], [118], [119]. How to select clustering method which generates ensemble members and how to select a suitable consensus function have been under study [51], [54], [53].

A simple framework for combining the results of multiple clusterings to produce a consistent clustering is presented in [120]. Graph-theoretic methods are widely used in cluster ensemble problems [114], [121]. Ensemble methods are also meant to speed-up the clustering process, like *ensemble of leaders*, proposed in [116] and it is shown that ensemble of weak clustering methods like leaders can produce results that are comparable to a strong clustering result and it consumes lesser time. A detailed information on several other ensemble based methods is presented in [122].

- **Hybrid clustering methods**

  Hybrid clustering methods combine two or more methods where the result of one method is given as input to the subsequent method. A hybrid method which combines hierarchical clustering and k-means clustering method is proposed in [94]. The method reduces the running time of the k-means method by providing better initial seed-points. Further, it is suitable for high dimensional datasets and reduces the impact of noise.

Rough-DBSCAN [16] proposed by Viswanath *et.al.,* is a faster version of DBSCAN which falls into this category. In *Rough*-DBSCAN, a fast clustering method like leaders clustering method is applied at first from which prototypes called leaders are generated. Later these leaders are given as input to the DBSCAN method. For large datasets *rough*-DBSCAN can find a similar clustering as found by the DBSCAN, but is consistently faster than DBSCAN.

l-DBSCAN [49] is similar to *rough*-DBSCAN except that *l*-DBSCAN derives two types of leaders (prototypes) called coarse leaders and fine leaders and achieves a very similar result as that of the DBSCAN which uses the dataset in a very reduced time, where as *rough*-DBSCAN uses only one type of leaders.

Some other recent trends include: Semi-supervised clustering [123], Large-scale clustering [5], [12], Multi-way clustering [5], [2], Parallel and Distributed clustering [124], [125], Sub-space clustering [126], Co-operative clustering methods [80], [127] *etc*..

1.6 Motivation and scope of the present work

k-means clustering method is a popular and simplest method that has been widely used in various application domains [11]. But, it fails to identify arbitrary shaped clusters in the data. kernel k-means method is a generalization of the k-means method which is proved to be efficient to identify arbitrary shaped clusters [90]. However, for large datasets, like in data mining applications, applying k-means or kernel k-means method will be a time consuming process. Because of the simplicity and practical importance of these methods in various applications, there is a need to speed-up the k-means and kernel k-means clustering methods

in order to work with large datasets. This is the motivating step for the proposed research.

The objective of the proposed thesis is to speed-up the k-means and kernel k-means clustering methods. To speed-up the k-means method, the thesis proposes two prototype based hybrid approaches, which gives the same clustering result as that obtained using the conventional k-means method. To speed-up the kernel k-means method, the thesis proposes two prototype based approaches and a single-pass approach, where, the clustering result obtained by these methods is similar to that obtained using the conventional kernel k-means method.

The key idea of the proposed prototype based hybrid approaches is to reduce the size of the dataset that is used with k-means or kernel k-means by selecting only few prototypes of the dataset. However, the computational requirements to derive these prototypes is also a main concern. Hence, the thesis argues to use several improved versions of the conventional leaders clustering method [35] to derive the prototypes in a single database scan. As the proposed methods are working with only prototypes, there may be some deviations in their final clustering result when compared to that obtained by using the entire dataset. Finally, some correcting steps are proposed to handle such deviations. The thesis establishes that, in some favorable cases the correcting step can be eliminated. Figure 1.6 (See towards the end of the Chapter) gives a schematic diagram of the proposed hybrid clustering methods.

The single-pass approach proposed in the thesis to speed-up the kernel k-means method is similar to the two-step kernel k-means clustering method [128]. The method works in two stages. Initially, the method selects a small random sample from the dataset. A partition of the selected sample is obtained by using the conventional kernel k-means method. For each cluster in the derived partition, the pattern in the input space which represents the center

(mean) of the cluster (in the induced space) is obtained by using the *gradient descent method*. Finally, each unsampled data point in the dataset is assigned to its nearest cluster center to derive a partition of the entire dataset. The experimental results shows that the proposed method is much faster than the prototype based methods, but the efficiency of this method depends on the size of the random sample. Figure 1.7 (See towards the end of the Chapter) gives a schematic diagram of the proposed single-pass kernel k-means clustering method.

1.6.1 Scope of the thesis

Given the same initial conditions, like the same dataset, the same seed points (or initial partition), the same number of clusters, *i.e.*, *k*, kernel function (in the case of kernel k-means clustering method) the thesis proposes some techniques to derive a clustering result which is same or similar to the clustering result obtained by the conventional k-means or kernel k-means clustering method but in a reduced time requirement. That is, the scope of the thesis is to speed-up the *k*-means or kernel *k*-means clustering process.

1.7 Organization of the thesis

The thesis is organized as follows.

Chapter 2 presents a review of the conventional k-means and kernel k-means clustering methods along with their limitations. Later, this chapter gives an outline of the existing methods in the literature, that are proposed to speed-up the k-means and kernel k-means clustering methods. Finally, the focus of the thesis and an outline of the proposed methods to speed-up the k-means and kernel k-means clustering methods are presented towards the end of the Chapter.

Chapter 3 explains the concept of prototypes, various prototype selection schemes, and how these prototypes are and used in various supervised and unsupervised schemes. Later, the leaders clustering method [35], which is used to derive prototypes in the proposed hybrid schemes, is discussed in detail along with its basic properties and limitations. Some modified versions of the conventional leaders clustering method, that are proposed in this thesis in order to work with the k-means or kernel k-means method, are explained towards the end of the Chapter.

Chapter 4 presents the proposed prototype based hybrid methods to speed-up the k-means clustering method *viz., lk*-means-CMFT and *lk*-means-CMVT. Theoretically and experimentally it is shown that, both the proposed methods give the same clustering result as produced by the conventional k-means method, but in a reduced time, and *lk*-means-CMVT is more efficient than *lk*-means-CMFT. Finally, the Chapter shows that the proposed methods outperforms the other recent index-based method, called the filtering method [129].

Chapter 5 presents the proposed prototype based hybrid methods to speed-up the kernel k-means clustering method *viz.,* kernel-*lk*-means-CMFT, kernel-*lk*-means-CMVT and the single-pass kernel k-means method. The proposed single pass method gives a great reduction on total running time, but the deviation in the clustering result when compared to that obtained by the conventional kernel k-means clustering method is also high. On the other hand, both theoretically and experimentally, the proposed prototype based methods are shown to produce similar or same clustering result as the conventional kernel k-means, but in a reduced time. Finally, all the proposed methods are compared with the recent similar methods *viz.,* two-step kernel k-means [128] and the approximate kernel k-means clustering method [128].

Chapter 6 gives the conclusions of the thesis and the future scope of the present work. Finally, the Chapter presents some of the publications that resulted out of the present research.

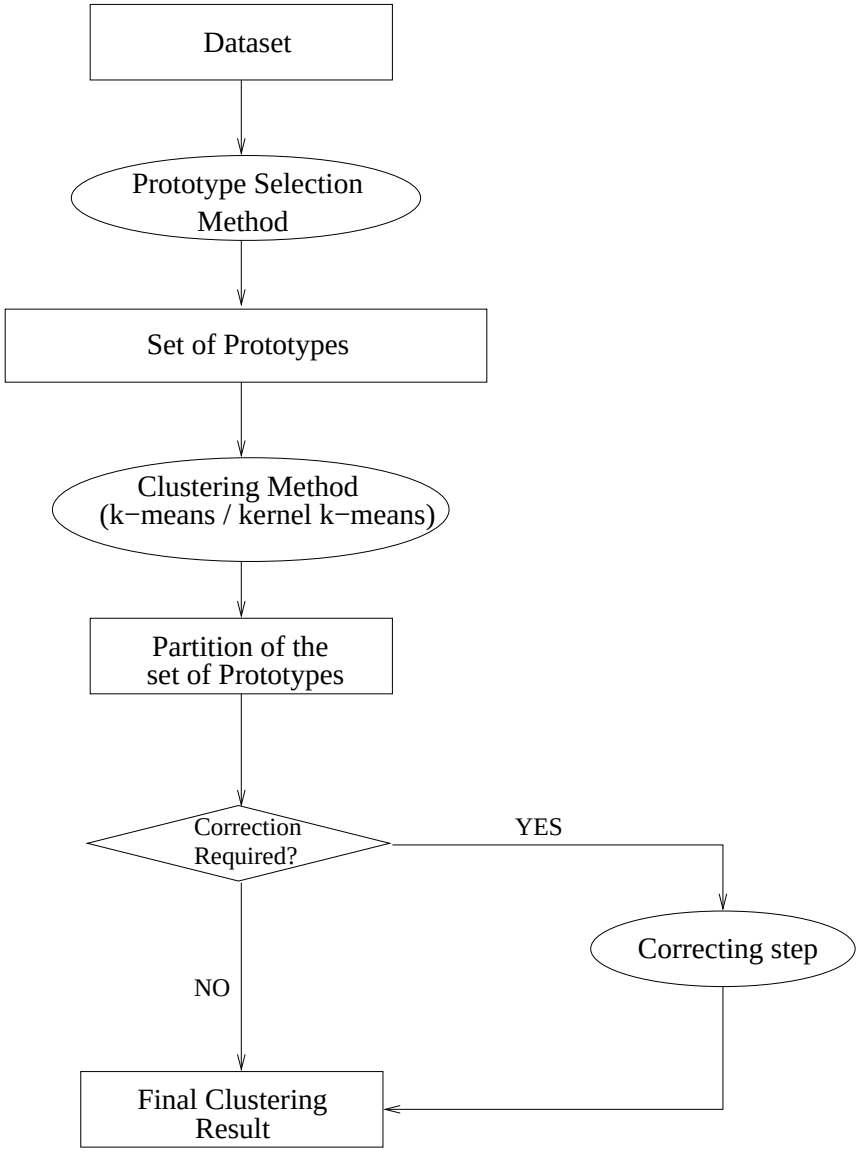


Figure 1.6: Schematic diagram of the proposed hybrid approaches

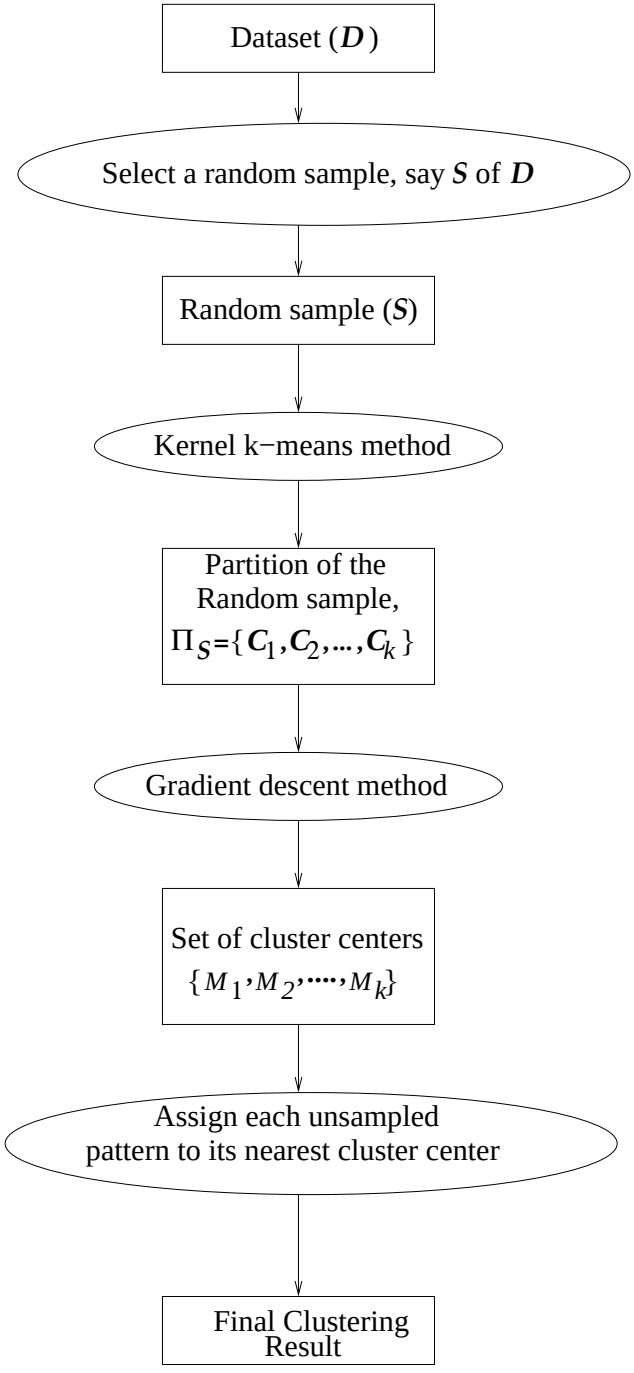


Figure 1.7: Schematic diagram of the proposed single-pass approach