CHAPTER 2

2. CLUSTER ANALYSIS

2.1 INTRODUCTION

The process of grouping a set of objects into classes of similar objects is called clustering and a cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. A cluster of data objects can be treated collectively as one group and so may be considered as a form of data compression. Although classification is an effective means for distinguishing groups or classes of objects, it is often more desirable to proceed in the reverse direction: First, partition the set of data into groups based on data similarity (e.g., using clustering) and then assign labels to the relatively small number of groups. Additional advantage of such a clustering-based process is that it is adaptable to changes and helps to single out useful features that distinguish different groups.

Clustering is also called as data segmentation in some applications because clustering partitions large data sets into groups according to their similarity. As a data mining function, cluster analysis can be used as a stand-alone tool to gain insight into the distribution of data, to observe the characteristics of each cluster and to focus on a particular set of clusters for further analysis. Alternatively, it may serve as a pre-processing step for various algorithms, such as characterization, attribute subset selection and classification which would then operate on the detected clusters and the selected attributes or features.

Data clustering is under vigorous development and contributing areas of research include data mining, statistics, machine learning, spatial database technology and marketing. Owing to the huge amounts of data collected in databases, cluster analysis has recently become a highly active topic in data mining research. Clustering techniques are considered as efficient tools for partitioning data sets in order to get homogeneous clusters of objects and these techniques [A.K.Jain and R.C.Dubes, 1988] are among the well-known machine learning techniques. Clustering techniques are widely used in many domains such as medicine, banking, finance, marketing,
security, etc. They work under an unsupervised mode when the class label of each object in the training set is not known.

In machine learning, clustering is an example of unsupervised learning. Unlike classification, clustering and unsupervised learning do not rely on pre-defined classes and class-labelled training examples. For this reason, clustering is a form of learning by observation rather than learning by examples. In data mining, efforts have focused on finding methods for efficient and effective cluster analysis in large databases. Active themes of research focus on the scalability of clustering methods, the effectiveness of methods for clustering complex shapes and types of data, high-dimensional clustering techniques and methods for clustering mixed numerical and categorical data in large databases and method for clustering uncertain data sets.

2.2 CATEGORIZATION OF CLUSTERING METHODS

A large number of clustering algorithms exist in the literature and it is difficult to provide a crisp categorization of clustering methods since these categories may overlap so that a method may have features from several categories. In general, the major clustering methods can be broadly classified into three categories which are discussed in the following sections.

2.2.1 PARTITIONING METHODS

Given a database of \( n \) objects and \( K \), the number of clusters to form, a partitioning algorithm organizes the objects into \( K \) partitions \( (K \leq n) \), where each partition represents a cluster. The clusters are formed to optimize an objective partitioning criterion, often called a similarity function, such as distance, so that objects within a cluster are similar whereas objects of different clusters are dissimilar in terms of the database attributes.

The most well-known and commonly used partitioning methods are K-means method [J. MacQueen, 1967] where each cluster is represented by the mean value of the objects in the cluster, K-medoids method [L.Kaufman and P.J.Rousseeuw, 1990] where each cluster is represented by one of the objects located near the center of the cluster and K-modes method [Z.Huang, 1998] to handle categorical data.
2.2.2 HIERARCHICAL METHODS

A hierarchical clustering method works by grouping data into a tree of clusters. Hierarchical clustering methods can be further classified into agglomerative and divisive hierarchical clustering, depending on whether the hierarchical decomposition is formed in a bottom-up or top-down fashion.

Agglomerative hierarchical clustering

This bottom-up strategy starts by placing each object in its own cluster and then merges these atomic clusters into larger and larger clusters, until all of the objects are in a single cluster or until certain termination conditions are satisfied.

Divisive hierarchical clustering

This top-down strategy does the reverse of agglomerative hierarchical clustering by starting with all objects in one cluster. It subdivides the cluster into smaller and smaller pieces, until each object forms a cluster on its own or until it satisfied certain termination conditions such as a desired number of clusters is obtained or the distance between the two closest clusters is above a certain threshold distance. The methods introduced in this category are: density-based and Grid-based [T. Zhang et al., 1996; S. Guha et al., 1998]

Density-based methods

The general idea is to continue growing the given cluster as long as the density (number of objects or data) in the neighborhood exceeds some threshold; that is, for each data point within a given cluster, the neighborhood of a given radius has to contain at least a minimum number of points [M. Ester et al., 1998; M. Ankerst et al., 1999].

Grid-based methods

Grid-based methods divide the object space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed [W. Wang et al., 1997; G. Sheikholeslami et al., 1998; R. Agrawal et al., 1998]. Grid-based clustering methods quantize the space into a finite number of cells that
form a grid structure and then all of the clustering operations are performed on this grid structure. The computational complexity of all of the previously mentioned clustering methods is at least linearly proportional to the number of objects. The unique property of grid-based clustering approach is that its computational complexity is independent of the number of data objects but dependent only on the number of cells in each dimension in the quantized space.

STatistical Information Grid (STING) [Wang et al., 1997] is a typical grid-based clustering method which divides the spatial area into rectangular cells. The algorithm constructs several levels of such rectangular cells and these cells form a hierarchical structure; i.e. each cell is partitioned to form a number of cells at the next lower level and Figure 2.1 illustrates the idea.

![Diagram of Hierarchical structure for STING clustering](image)

**Figure 2.1 Hierarchical structure for STING clustering**

Statistical information such as means, maximum and minimum values of each grid cell are pre-computed and stored for later query processing. The clustering quality of STING highly depends on the granularity of the lowest level of the grid structure. If the granularity is too coarse then the accuracy of clustering solution will degrade. However, if the granularity is too fine, the processing time will increase drastically. Another limitation of STING is that it can only represent clusters in either
horizontal or vertical rectangular shape. Although this method is efficient, its limitations substantially lower than the accuracy of the clustering result.

2.2.3 MODEL-BASED METHODS

These methods attempt to optimize the fit between the given data and some mathematical model. Model-based methods follow two major approaches: a statistical approach and neural network approach. In the statistical approach, the attention is focused on statistical multisource analysis by means of a method based on Bayesian classification theory and this method is investigated and extended to take into account the relative reliabilities of the sources of data involved in the classification. This requires a way to characterize and quantify the reliability of a data source which becomes important when looking at the combination of information.

Recently, there has been a great resurgence of research in neural network approach and this approach can be successfully used to classify complex data. Neural network approaches have an advantage over the statistical methods that they are distribution free and no prior knowledge is needed about the statistical distributions of the classes in the data sources in order to apply these methods for classification. The neural network approaches also take care of determining how much weight each data source should have in the classification. A set of weights describes the neural network and these weights are computed in an iterative training procedure. On the other hand, neural network approaches can be very complex computationally need a lot of training samples to be applied successfully and their iterative training procedures usually slow to converge. Also, neural network approaches have more difficulty than do statistical methods in classifying patterns which are not identical to one or more of the training patterns. The performance of the neural network approaches in classification is therefore more dependent on having representative training samples, whereas the statistical approaches need to have an appropriate model of each class.

The choice of clustering algorithm depends both on the type of data available and on the particular purpose of the application. If cluster analysis is used as a descriptive or exploratory tool, it is possible to try several algorithms on the same data to see what the data may disclose.
2.3 ANALYSIS OF TEXT CLUSTERING ALGORITHMS

Large number of clustering algorithms described in the literature focuses on text clustering and there arises a need for analysing the algorithms which focuses on categorical text clustering. The following section provides a brief overview about the important categorical text clustering algorithms. The most common cluster algorithms such as K-mode [Zhexue Huang, 1998], RObust Clustering using linKs (ROCK) [Sudipto Guha et al., 1999] and Sieving Through Iterated Relational Reinforcement (STIRR) [David Gibson, Jon M et al., 1998] are discussed below. The difficulties in clustering algorithms are language variability which has same meaning but it is phrased on two ways. By making the sentence smaller it would be exact matching of their terms. By doing this, one can expect the cluster which can be closely matched to the concepts described based on the query terms. But most of the documents have irrelevant details of topics or themes and many sentences will be related to some degree. The calculation between the pair-wise similarities or dissimilarities can be performed with data points that should be done from attribute data based on similarities such as cosine similarity. This can be also applicable for relational clustering algorithms.

2.3.1 OVERVIEW OF K-MEANS AND K-MODES METHODS

K-means [MacQueen, 1967] is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. Modification in k-means has been developed by focusing on background knowledge that can be expressed as a set of instance-level constraints on the clustering process.

In the context of text clustering algorithms, instance level constraints are a useful way to express a priori knowledge about which instances should or should not be grouped together. The most common instance level constraints are Must-link constraints which specify that two instances have to be in the same cluster and Cannot-link constraints which specify that two instances must not be placed in the same cluster. The must-link constraints define a transitive binary relation over the instances. Consequently, when making use of a set of constraints (of both kinds) cannot-link constraints take a transitive closure over the constraints [Kiri Wagsta et al., 2001].
K-means method [L.Kaufman and P.J.Rousseeuw, 1990] has been a very popular technique for partitioning large data sets with numerical attributes. However, data mining applications frequently involve many data sets that also consist of categorical attributes. One approach proposed by H.Ralambondainy [1995] uses the K-means algorithm to cluster categorical data by converting multiple categories attributes into binary attributes (using 0 and 1 to represent either a category absent or present) and treat the binary attributes as numeric. If this algorithm is used in data mining, it needs to handle a large number of binary attributes because categorical attributes in data sets often have hundreds or thousands of categories. This will increase both computational and space costs of the K-means algorithm. Another drawback is that the cluster means, given by real values between 0 and 1 does not really reflect the characteristics of the clusters.

Indeed, in applying K-means method to categorical data, two main problems are encountered, namely, the construction of clusters’ centers and the definition of dissimilarity between objects and clusters’ centers. The K-modes method [Z.Huang, 1998] is considered as one of the most popular of them and it was proposed to extend the K-means to tackle the problem of clustering large categorical data sets in data mining and this method uses the K-means paradigm to cluster data having categorical values.

K-modes method parameters

K-modes method is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The K-modes method extends the K-means [L.Kaufman and P.J.Rousseeuw, 1990] one by using a simple matching dissimilarity measure for categorical objects, modes instead of means for clusters and a frequency-based method to update modes in the clustering process to minimize the clustering cost function. The modifications of the K-means algorithm are discussed below.

Given a cluster \( \{X_1, \ldots, X_p\} \) of \( p \) categorical objects with \( X_i = (x_{i,1}, \ldots, x_{i,s}) \), \( 1 \leq i \leq p \), its mode \( Q = (q_1, \ldots, q_s) \) is defined by assigning \( q_j \), \( 1 \leq j \leq s \), where \( s \) is the number of attributes, the category most frequently encountered in \( \{x_{1,j}, \ldots, x_{p,j}\} \). However, the mode of cluster is not generally unique and this makes the algorithm unstable depending on mode selection during the clustering process.
The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume K clusters) fixed a priori and the main idea is to define K-modes, one for each cluster. These modes should be placed in a critical way because of different location causes different results. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest mode using the simple matching dissimilarity measure. When no point is pending, the first step is completed and an early groupage is done. At this point, it is necessary to recalculate K-new modes of the clusters resulting from the previous step based on frequency-based method. Then, the algorithm has these K-new modes, a new binding has to be done between the same data set points and the nearest new mode and a loop has been generated. As a result of this, loop may notice that the K-modes change their location step by step until no more changes are done and in other words modes do not move any more.

Although it is proved that the procedure will always terminate, the K-modes algorithm does not necessarily find the most optimal configuration, corresponding to the global objective function minimum. The algorithm is also significantly sensitive to the initial randomly selected cluster modes. The K-modes algorithm can be run multiple times to reduce this effect. This algorithm has an input, a pre-defined number of clusters i.e. the K from its name. K-modes algorithm is an iterative procedure in which a crucial concept is the one of mode. Mode is an artificial point in the space of records which represents all objects of the particular cluster. The coordinates of this point are the most frequent occurrences of attribute values that belong to the cluster.

The algorithm is composed of the following steps:

1. Select K initial modes, one for each cluster.
2. Allocate an object to the cluster whose mode is the nearest to it according to the simple matching dissimilarity measure and update the mode of the cluster after each allocation.
3. Once all objects have been allocated to clusters, retest the dissimilarity of objects against the current modes. If an object is found such that its nearest mode belongs to another cluster rather than its current one, reallocate the object to that cluster and update the modes of both clusters.
4. Repeat 3 until no object has changed clusters after a full cycle test of the whole data set.

Consider a classification problem of firm’s staff and suppose that a training set $T$ is defined by Table 2.1.

**Table 2.1 Training set $T$ relative to the standard K-modes method**

<table>
<thead>
<tr>
<th>Objects</th>
<th>Qualification</th>
<th>Income</th>
<th>Departments</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_1$</td>
<td>A</td>
<td>High</td>
<td>Finance</td>
</tr>
<tr>
<td>$X_2$</td>
<td>B</td>
<td>Low</td>
<td>Finance</td>
</tr>
<tr>
<td>$X_3$</td>
<td>C</td>
<td>Average</td>
<td>Marketing</td>
</tr>
<tr>
<td>$X_4$</td>
<td>C</td>
<td>Average</td>
<td>Accounts</td>
</tr>
<tr>
<td>$X_5$</td>
<td>B</td>
<td>Low</td>
<td>Marketing</td>
</tr>
<tr>
<td>$X_6$</td>
<td>A</td>
<td>High</td>
<td>Finance</td>
</tr>
<tr>
<td>$X_7$</td>
<td>B</td>
<td>Low</td>
<td>Accounts</td>
</tr>
</tbody>
</table>

Suppose that $K = 3$, 3-partition of $T$ is initialized randomly as follows:

$$C_1 = \{X_1\}, \ C_2 = \{X_2\}, \text{and} \ C_3 = \{X_3\}.$$

The three cluster modes, one for each cluster are defined by:

$Q_1 = (A, \text{High, Finance})$

$Q_2 = (B, \text{Low, Finance})$

$Q_3 = (C, \text{Average, Marketing})$

The k-modes algorithm is simple and understandable and the algorithm converges in a finite number of iterations. This standard version of the K-modes method has some of the weaknesses: ① The ways of initializing the modes was not specified. One popular way to start is to randomly choose the K of the samples. So, the produced results depend on the initial values for the modes. The standard solution is to try a number of different starting points. ② The results depend on the value of K. Unfortunately, there is no general theoretical solution to find the optimal number of
clusters for any given data set. A simple approach is to compare the results of multiple runs with different K classes and choose the best one according to a given criterion such that the clustering cost function. The mode of cluster is not generally unique and this makes the algorithm unstable depending on mode selection during the clustering process.

**K-modes method under uncertainty**

Standard versions of the K-modes method and its extensions give good results in a context in which everything is known with certainty. The reality is connected to uncertainty and imprecision by nature and such uncertainty may badly affect the classification performance. However, a good classifier must be able to predict the object’s class value even when information concerning the object is imperfect. So, the K-modes are inadequate and badly adapted to ensure its role of classification in an environment characterized by a lot of uncertainty and imprecision. Due to the above stated problems or reasons, researchers are interested in improving or extending this method. The idea of k-modes method with uncertainty is to combine theories managing uncertainty and imprecision with the K-modes method. The theories are probability theory, fuzzy set theory, belief function theory and possibilistic theory. Hence, this adaptation of K-modes method to an uncertain environment has led to a new approach and the fuzzy K-modes method [Z.Huang and M.K.Ng, 1999] was developed. In this approach, one object does not belong exclusively to a well-defined cluster. In fact, it may belong to several clusters with different membership degrees. This extension can be said as fuzzy k-modes method which is briefly presented below.

**2.3.2 FUZZY K-MODES METHOD**

Fuzzy K-modes method deals with cognitive uncertainty and it can take into account imprecision and fuzziness in object class memberships by using fuzzy sets and membership degrees. Fuzzy K-modes approach is a method of clustering which allows one piece of data to belong to two or more clusters. It uses fuzzy partitioning such that a data point can belong to all groups with different membership grades between 0 and 1.
The algorithm is composed of the following steps:

1. Choose an initial point \( Q^{(1)} \). Determine a fuzzy partition matrix \( W^{(1)} \) such that 
   \[ P(W, Q^{(1)}) \] is minimized and set \( t = 1 \).

2. Determine \( Q^{(t+1)} \) such that \( P(W^{(t)}, Q^{(t+1)}) \) is minimized.
   If \( P(W^{(t)}, Q^{(t+1)}) = P(W^{(t)}, Q(t)) < e \) then STOP; otherwise return to step 3

3. Determine \( W^{(t+1)} \) such that \( P(W^{(t+1)}, Q^{(t+1)}) \) is minimized.
   If \( P(W^{(t+1)}, Q^{(t+1)}) = P(W^{(t)}, Q^{(t+1)}) < e \) then STOP;
   otherwise set \( t = t + 1 \) and go to step 2.

The fuzzy K-modes algorithm produces a fuzzy partition matrix \( W \). To obtain the cluster membership from \( W \), the record \( X_i \) was assigned to the \( l \)th cluster if 
\[ w_{i,l} = \max_{1 \leq h \leq k} \{ w_{i,h} \} . \] If the maximum is not unique, then \( X_i \) was assigned to the cluster of first achieving the maximum.

2.3.3 ROCK

ROCK is a hierarchical clustering algorithm that explores the concept of links (the number of common neighbors between two objects) for data with categorical attributes. Two distinct clusters may have a few points or outliers that are close; therefore, relying on the similarity between points to make clustering decisions could cause the two clusters to be merged. ROCK takes a more global approach to clustering by considering the neighborhoods of individual pairs of points. If two similar points also have similar neighborhoods, then the two points likely belong to the same cluster and so can be merged [Sudipto Guha et al., 1998].

![Figure 2.2 Overview of ROCK](image)

**Figure 2.2 Overview of ROCK**

The steps involved in clustering using ROCK are described in Figure 2.2. After drawing a random sample from the database, a hierarchical clustering algorithm that employs links is applied to the sampled points. Finally, the clusters involving only the sampled points are used to assign the remaining data.
points on disk to the appropriate clusters. In the following subsections describe the steps performed by ROCK in greater detail.

Overview of the algorithm

ROCK’s hierarchical clustering algorithm is presented in Figure 2.3 and it accepts as input the set $S$ of $n$ sampled points to be clustered (that are drawn randomly from the original data set) and the number of desired clusters $k$. The procedure begins by computing the number of links between pairs of points in Step 1. Initially, each point is a separate cluster. For each cluster $i$, build a local heap $q[i]$ and maintain the heap during the execution of the algorithm. $q[i]$ contains every cluster $j$ such that assume $[i, j]$ is non-zero. The clusters $j$ in $q[i]$ are ordered in the decreasing order of the goodness measure with respect to $i$, $g(i,j)$.

---

```plaintext
Procedure cluster(S, k)
begin
1. link:= compute_links(S)
2. for each $s \in S$ do
3.    $q[s] :=$ build_local_heap(link, $s$)
4.    $Q :=$ build_global_heap ($S, q$)
5. while size($Q$) > $k$ do {
6.    $u :=$ extract_max($Q$)
7.    $v :=$ max($q[u]$)
8.    delete($Q, v$)
9.    $w :=$ merge($u, v$)
10.   for each $x \in q[u] \cup q[v]$ do {
11.       link[$x, w] :=$ link[$x, u] + link[$x, v$]
12.       delete($q[x], u$); delete($q[x], v$)
13.       insert($q[x], w, g(x, w)$); insert($q[w], x, g(x, w)$)
14.       update($Q, x, q[x]$)
15.   }
16.   insert($Q, w, q[w]$)
17.   deallocate($q[u]$); deallocate($q[v]$)
18. }
End
```

Figure 2.3 Rock clustering algorithm
The while-loop in Step 5 iterates until only \( k \) clusters remain in the global heap \( Q \). In addition, it also stops clustering if the number of links between every pair of the remaining clusters becomes zero. In each step of the while-loop, the max cluster \( u \) is extracted from \( Q \) by \text{extract\_max} \ and \( q[u] \) is used to determine the best cluster \( v \) for it. Since clusters \( u \) and \( v \) will be merged, entries for \( u \) and \( v \) are no longer required and can be deleted from \( Q \). Clusters \( u \) and \( v \) are then merged in Step 9 to create a cluster \( w \) containing \(|u| + |v|\) points. There are two tasks that need to be carried out once clusters \( u \) and \( v \) are merged:

1. For every cluster that contains \( u \) or \( v \) in its local heap, the elements \( u \) and \( v \) need to be replaced with the new merged cluster \( w \) and the local heap needs to be updated.

2. New local heap for \( w \) needs to be created. Both these tasks are carried out in the for-loop of Step 10-15. A detailed description of how this for-loop works was given by Sudipto Guha et al., [1998].

```
procedure compute_links(S)
begin
1. Compute nbrlist[i) for every point i in S
2. Set link[i, j) to be zero for all i, j
3. for i := 1 to n do { 
4. N := nbrlist[i)
5. for j := 1 to 
6. for l := j + 1 to 
7. link[N(j), N(l)) := link[N(j), N(l)) + 1
8. }
End
```

**Figure 2.4 Algorithm for computing links**

The algorithm in Figure 2.4 provides a more efficient way of computing links. In that, on an average, the number of neighbors for each point will be small compared to the number of input points \( n \), causing the adjacency matrix \( A \) to be sparse.

For every point, after computing a list of its neighbors, the algorithm considers all pairs of its neighbors. For each pair, the point contributes one link. If
the process is repeated for every point and the link count is incremented for each pair of neighbors, then at the end, the link counts for all pairs of points will be obtained. If \( m_i \) is the size of the neighbor list for point \( i \), then for point \( i \) and the link count should be increased by one in \( m_i^2 \) entries. Thus, the complexity of the algorithm is \( \sum m_i^2 \) which is \( O(nm_am_a) \), where \( m_a \) and \( m_m \) are the average and maximum number of neighbors for a point, respectively. In the worst case, the value of \( m_m \) can be \( n \) in which case, the complexity of the algorithm becomes \( O(m_a n^2) \). In practice, \( m_m \) to be reasonably close to \( m_a \) and thus, for these cases, the complexity of the algorithm reduces to \( O(m^2 n) \) on average.

**Time and Space Complexity**

**Computation of Links:** It is possible to compute links among pairs of points in \( O(n^{2.37}) \) using standard matrix multiplication techniques, or alternatively in \( O(n^2 m_a) \) time for average number of neighbors \( m_a \). The space requirement for the link computation is at most \( n(n + 1)/2 \), when every pair of points are linked. However, in general, not every pair of points will have links between them and expect the storage requirements to be much smaller.

The algorithm can show this to be \( O(min(nm_m m_a, n^2)) \) where \( mm \) is the maximum number of neighbors for a point and this is because a point \( i \) can have links to at most \( min(n, m_mm_i) \) other points. The time to build each local heap initially is \( O(n) \) (a heap for a set of \( n \) input clusters can be built in time that is linear in the number of clusters [Thomas H et al., 1990]). The global heap also has at most \( n \) clusters initially and can be constructed in \( O(n) \) time. Next, the complexities of the steps in the while-loop which is executed \( O(n) \) times. The inner for-loop dominates the complexity of the while-loop. Since the size of each local queue can be \( n \) in the worst case and the new merged cluster \( w \) may need to be inserted in \( O(n) \) local queues, the time complexity of the for-loop becomes \( O(n \log n) \) and that of the while-loop is \( O(n^2 \log n) \) in the worst case. Due to the above analysis, ROCK’s clustering algorithm, along with computation of neighbor lists and links, has a worst-case time complexity of \( O(n^2 + nm_mm_a + n^2 \log n) \).
The space complexity of the algorithm depends on the initial size of the local heaps. The reason for this is that when two clusters are merged, their local heaps are deleted and the size of the new cluster’s local heap can be no more than the sum of the sizes of the local heaps of the merged clusters. Since each local heap only contains those clusters to which it has non-zero links, the space complexity of ROCK’s clustering algorithm is the same as that of link computation, that is, $O(\min \{n^2, nm_m\})$.

**Random Sampling**

In case the database is large, random sampling enables ROCK to reduce the number of points to be considered and ensures that the input data set fits in main memory. Consequently, significant improvements in execution times for ROCK can be realized. With an appropriate sample size, the quality of the clustering is not sacrificed. On the contrary, random sampling can aid clustering by filtering outliers. Efficient algorithms for selecting random samples from a database are found vastly in the literature [Jeff Vitter, 1985]. Also, an analysis of the appropriate sample size for good quality clustering can also be found [Sudipto Guha et al., 1998] and it is noted that the salient feature of ROCK is not sampling but the clustering algorithm that utilizes links instead of distances.

**Handling Outliers**

In ROCK, outliers can be handled fairly effectively. The first pruning occurs when choose a value for $\theta$ and by definition outliers are relatively isolated from the rest of the points. This immediately allows one to discard the points with very few or no neighbors because they will never participate in the clustering and this is the most significant part where outliers are eliminated. However, in some situations, outliers may be present as small groups of points that are loosely connected to the rest of the dataset. This immediately suggests to the researchers that these clusters will persist as small clusters for the most part of clustering. These will only participate in clustering when the number of clusters remaining is actually close to the number of clusters in the data. So, the algorithm should be stopped at a point such that the number of remaining clusters is a small multiple of the expected number of clusters.
In the final labelling phase, ROCK assigns the remaining data points residing on disk to the clusters generated by using the sampled points. This is performed as follows. First, a fraction of points from each cluster \( i \) is obtained; Let \( L_i \) denote this set of points from cluster \( i \) used for labelling. Then, the original data set is read from disk and each point \( p \) is assigned to the cluster \( i \) such that \( p \) has the maximum neighbors in \( L_i \) (after normalization). Sudipto Guha et al., [1999] proposed a new concept of links to measure the similarity/proximity between a pair of data points with categorical attributes and this algorithm employs links and not distances for merging clusters.

### 2.3.4 STIRR ALGORITHM

A novel approach for clustering collections of sets has been described [Gibson et al, 1998] and its application to the analysis and mining of categorical data. The collection of sets referred as the relational table with each tuple visualized as a set. The STIRR algorithm works on categorical data-fields in tables whose attributes cannot be ordered as numerical values can [Zhang et al, 2000]. As categorical data usually do not have inherent geometric properties, the clustering of categorical data seems more complicated than that of numerical data. Mining of association rules is effective only for items that appear in the same tuple. The STIRR algorithm not only takes into consideration items that appear together in a tuple, but also identifies relationships amongst items occurring in different tuples.

#### Overview of the algorithm

- **Iterative method** – The STIRR algorithm is an iterative method and the number of iterations depends upon the dataset in consideration. The algorithm keeps on performing the same steps a number of times until a result is obtained which does not change on further iterations.

- **Assigning and propagating weights on categorical values** - A relational table is taken as input to the algorithm and this relational table has fields (attributes) that can take values in a particular domain. The STIRR algorithm takes each distinct value in the table and performs a series of steps to assign numerical values (weights) to it.
• *Similarity measure obtained from co-occurrence of values in dataset* - Each distinct value in the database is assigned a weight. In the first iteration of the STIRR algorithm, the weight of each distinct value is calculated depending on with what values this distinct value appears in the database. e.g. for the distinct group value “defence”, for every tuple in which the group “defence” appears and takes the sum of the target and weapon weights that occur in those tuples and assign this total weight to the “defence” attribute value. This is repeated for all distinct values in the database. Then, in the subsequent iterations, the same procedure is repeated and hence this weight is propagated further – to other targets and weapons and even to other military groups. Thus, the algorithm achieves a two-fold objective – items highly related to “defence” acquire weight even without occurring in the same tuple; and since the weight diffuses as it propagates through the database, a limited form of transitivity is formed.

• *Based on mathematical model of non-linear dynamical systems* - Gibson et al, [1998] suggested that each tuple in the database can be viewed as a set of items and the entire collection of tuples as an abstract set system or hypergraph. According to Zhang et al, [2000] a hypergraph is an extension of a graph such that each hyperedge may be identified by more than two nodes. The node set corresponds to distinct items in the dataset. The STIRR algorithm can be seen as a generalization of the spectral graph partitioning method to the problem of clustering collection of sets (hypergraphs). This generalization involves changes in the algorithms, in particular, the eigenvectors concept is replaced by certain types of non-linear dynamical systems.

• The authors argued that this introduction of dynamical systems is perhaps the most natural way to extend the power of spectral methods to the problem of clustering collections of sets; and authors showed that this approach suggests a framework for analyzing co-occurrence in categorical datasets in a way that avoids many of the pitfalls encountered with intractable combinatorial formulations of the problem [Gibson et al, 1998].
Algorithm and analysis

Step 1

The Table 2.2 views mine as a relational table with three fields Groupname, Target and Weapon, each of which can assume one of many possible values. To represent each distinct value as a node, the dataset is viewed as a set T of tuples and each tuple consists of one node from each field. The dataset can be viewed as shown in Table 2.3.

Step 2

To maintain a configuration of distinct nodes as a vector, a weight $w_v$ is assigned to each node $v$. The configuration is denoted by the letter $w$.

For the above example

$w' = [\text{Node1} \ \text{Node2} \ \text{Node3} \ \text{Node4} \ \text{Node5} \ \text{Node6} \ \text{Node7} \ \text{Node8} \ \text{Node9}]$

where $w'$ is the transpose of $w$.

Table 2.2 Example relational table with three fields

<table>
<thead>
<tr>
<th>Groupname</th>
<th>Target</th>
<th>Weapon</th>
</tr>
</thead>
<tbody>
<tr>
<td>17N</td>
<td>Unknown</td>
<td>Explosives</td>
</tr>
<tr>
<td>AKSh</td>
<td>Government</td>
<td>Firearms</td>
</tr>
<tr>
<td>…..</td>
<td>…..</td>
<td>…..</td>
</tr>
<tr>
<td>…..</td>
<td>…..</td>
<td>…..</td>
</tr>
<tr>
<td>ASG</td>
<td>Private Citizens &amp; Property</td>
<td>Firearms</td>
</tr>
<tr>
<td>ASG</td>
<td>Private Citizens &amp; Property</td>
<td>Explosives</td>
</tr>
</tbody>
</table>

Table 2.3 Dataset view of Table 2.2

<table>
<thead>
<tr>
<th>Node 1</th>
<th>Node 4</th>
<th>Node 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 2</td>
<td>Node 5</td>
<td>Node 8</td>
</tr>
<tr>
<td>…..</td>
<td>…..</td>
<td>…..</td>
</tr>
<tr>
<td>…..</td>
<td>…..</td>
<td>…..</td>
</tr>
<tr>
<td>Node 3</td>
<td>Node 6</td>
<td>Node 8</td>
</tr>
<tr>
<td>Node 3</td>
<td>Node 6</td>
<td>Node 9</td>
</tr>
</tbody>
</table>
Step 3

Initialize the configuration with initial weights. If all distinct values are to be treated equally, then initial configuration weights are assigned as 1. If some nodes are to be favored or diminished, then, that specific nodes can be initialized accordingly.

Step 4

Update the weight \( w_v \) of each node \( v \).

1. For each node \( v \), obtain tuples where \( v \) occurs.
2. For each tuple \( t = \{v, u_1, u_2, \ldots, u_{k-1}\} \) containing \( v \) do

\[
x = u_1 + u_2 + \ldots + u_{k-1}
\]

\[
w_v = \sum x
\]

e.g. For nodes in example below (initial weights are in parentheses).

<table>
<thead>
<tr>
<th>Groupname</th>
<th>Target</th>
<th>Weapon</th>
</tr>
</thead>
<tbody>
<tr>
<td>17N (1)</td>
<td>Unknown (1)</td>
<td>Explosives (1)</td>
</tr>
<tr>
<td>ASG (1)</td>
<td>Private Citizens &amp; Property (1)</td>
<td>Firearms (1)</td>
</tr>
<tr>
<td>ASG (1)</td>
<td>Private Citizens &amp; Property (1)</td>
<td>Explosives (1)</td>
</tr>
</tbody>
</table>

After 1\(^{st}\) iteration of STIRR algorithm the values Table 2.4 are modified as shown in Table 2.5

<table>
<thead>
<tr>
<th>Groupname</th>
<th>Target</th>
<th>Weapon</th>
</tr>
</thead>
<tbody>
<tr>
<td>17N (2)</td>
<td>Unknown (2)</td>
<td>Explosives (4)</td>
</tr>
<tr>
<td>ASG (4)</td>
<td>Private Citizens &amp; Property (4)</td>
<td>Firearms (2)</td>
</tr>
<tr>
<td>ASG (4)</td>
<td>Private Citizens &amp; Property (4)</td>
<td>Explosives (4)</td>
</tr>
</tbody>
</table>
Step 5

Normalize weights obtained of each field separately to eliminate influence of highly occurring values.

Table 2.6 Result of normalized weights

<table>
<thead>
<tr>
<th>Groupname</th>
<th>Target</th>
<th>Weapon</th>
</tr>
</thead>
<tbody>
<tr>
<td>17N (2)</td>
<td>Unknown (2)</td>
<td>Explosives (4)</td>
</tr>
<tr>
<td>ASG (4)</td>
<td>Private Citizens &amp; Property (4)</td>
<td>Firearms (2)</td>
</tr>
<tr>
<td>ASG (4)</td>
<td>Private Citizens &amp; Property (4)</td>
<td>Explosives (4)</td>
</tr>
</tbody>
</table>

Table 2.7 is obtained after normalization from Table 2.6

Table 2.7 Result of normalization of Table 2.6

<table>
<thead>
<tr>
<th>Groupname</th>
<th>Target</th>
<th>Weapon</th>
</tr>
</thead>
<tbody>
<tr>
<td>17N (0.33)</td>
<td>Unknown (0.33)</td>
<td>Explosives (0.67)</td>
</tr>
<tr>
<td>ASG (0.67)</td>
<td>Private Citizens &amp; Property (0.67)</td>
<td>Firearms (0.33)</td>
</tr>
<tr>
<td>ASG (0.67)</td>
<td>Private Citizens &amp; Property (0.67)</td>
<td>Explosives (0.67)</td>
</tr>
</tbody>
</table>

Step 6

Replace the weights in the old configuration with these normalized weights. A new configuration can be received as follows.

\[ w' = [\text{Node1}(0.33) \quad \text{Node2}(0.67) \quad \text{Node3}(0.33) \quad \text{Node4}(0.67) \quad \text{Node5}(0.67) \quad \text{Node6}(0.33)] \]

Step 7

Keep iterating Step 1 to Step 6 until the new configuration and the old configuration do not differ (norm of the residual must reach a threshold) and the number of iterations depends on the dataset.
Step 8

The final configuration has weights corresponding to each distinct value in the database (node). This determines which cluster the node belongs to.

Final configuration is \( w = [0.004 \ 0.048 \ 0.034 \ 0.045 \ 0.003 \ 0.043] \)

In the example, the first two weights 0.004 and 0.048 correspond to two groups (17N and ASG). If the weights are similar, then they belong to the same cluster. Similarity can be gauged based upon all the resulting weights that obtain and then determining ranges of weights falling in each cluster and weights in the same cluster are deemed to be similar.

The reasons behind choosing the STIRR algorithm for the study are listed below:

- **No apriori quantization**: The format of input dataset is not converted into a numerical format as which was converted in the previous numerical algorithms. This preserves the inherent structure of data and this also avoids problems with sparsely distributed data.

- **Differing from association rules**: Similarity should also be propagated among items in the database that don’t occur in the same tuple. Similarity in STIRR is measured purely on co-occurrence. This promotes relationships even among data items that do not occur in the same tuple. STIRR also promotes transitive similarity. If A is related to B and B is related to C, then A is related to C.

- **New Hypergraph approach**: STIRR generalizes the spectral graph partitioning technique into a system based on non-linear dynamical systems. This avoids NP-complete combinatorial formulations that earlier techniques followed.

Apart from many advantages of STIRR, it has the following limitations:

- **Does not converge for some input** – For some input, the STIRR algorithm fails to converge and it goes on looping and does not reach an end point.
The algorithm normalizes weights belonging to each attribute separately. This favours attributes that have few distinct values in their domain. These weights tend to obtain higher weights than those attributes that have more distinct values in their domains.

These disadvantages prompted researchers to come up with a new technique called “modified” STIRR or “revised” STIRR.

To cluster the categorical data, the STIRR based algorithms introduced various approaches in order to tackle the problems. Their performance gives various solutions in respect to the time, power and memory which also improves the quality of the clustering. The comparison Table 2.8 shows that k-modes and its prototypes are scalable on discussing with the datasets. The hierarchical algorithm ROCK is based on the attribute values and its occurrence is examined with number of other attribute values with which it exists. The results are also scalable on comparing with other sampling techniques but the efficiency is less than the k-modes. The STIRR algorithm results have acquired either a positive or negative weight on two clusters. In resulting stage, it requires costly post-processing steps. The different partitions of their data set which leads to the conclusion but those clustering must be in meaningful order. The best thing in STIRR algorithm is it converges quickly and identifies clusters in the presence of irrelevant values.

The K-modes algorithm requires more memory operation and also it is applicable for large inputs. For a large dataset, the STIRR needs one pass over the data set and a linear number of in-memory operations. ROCK is not suitable for large datasets and does not contain common quality measure. The above mentioned three algorithms face some disadvantages like text clustering, similarity analyses in clustering tuples and are not able to produce more than two clusters of attribute values. Moreover, the comparison of three algorithms based on single dataset is too difficult. Hence, based on the above observation, fuzzy based algorithm has been proposed in which the result belongs to a single cluster [Jagatheesan S.M and Thiagarasu V, 2014].
Table 2.8 Comparison of clustering methods

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Input Parameters</th>
<th>Optimized For</th>
<th>Outlier Handling</th>
<th>Computational Complexity (number of in-memory operations)</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-modes</td>
<td>Number of cluster</td>
<td>Data Sets with Well-separated Clusters</td>
<td>No</td>
<td>$\varphi(n)$</td>
</tr>
<tr>
<td>ROCK</td>
<td>Number of Cluster, Similarity Threshold</td>
<td>Small Data Sets with Noise</td>
<td>Yes</td>
<td>$\varphi(n^2 + nm + n^2\log n)$</td>
</tr>
<tr>
<td>STIRR</td>
<td>Initial Configuration, Combining Operator, Stopping Criteria</td>
<td>Large Data Sets with Well separated Clusters</td>
<td>No</td>
<td>$\varphi(n)$</td>
</tr>
</tbody>
</table>

2.4 FUZZY LOGIC

In 1965, Professor L.A. Zadeh of the University of California, Berkely outlined fuzzy theory and introduced fuzzy set theory and operation, fuzzy logic based controller, etc., In 1970, fuzzy logic theory was applied in many systems in Japan, China and Europe. In 1987, sixteen station subway railway systems were built and it worked with a fuzzy logic-based automatic train operation control system in Sendai and Japan. It was proved that by applying fuzzy logic based automation, the train ride was smooth. Fuzzy logic is a powerful problem-solving methodology and it has myriad of applications in embedded information processing and control. Fuzzy provides remarkably simple and definite conclusions. Conclusions are made from vague, ambiguous and imprecise information. Fuzzy logic resembles human decision making and it has ability to work from approximate data and also it finds precise solutions. Classical logic requires a deep understanding of a system,
exact equations and precise numeric values. Fuzzy logic provides an alternative way of thinking and fuzzy logic allows modeling complex systems while using a higher level of abstraction that originates from knowledge and experience. Fuzzy logic expresses knowledge with subjective concepts like bright red, very hot, long time, very quick etc. are mapped into exact numeric ranges.

Fuzzy Logic is an extension of Boolean logic and it incorporates partial values of truth. Instead of sentences being "Completely True" or "Completely False". In fuzzy logic, they are assigned a value which represents their degree of truthiness. In fuzzy systems, values are indicated by a number called as truth value and it lies in the range from 0 to 1.0. 0 represents absolute falseness and 1.0 represents absolute truth. Fuzzification is generalization of theory from discrete to continuous. Fuzzy logic allows computers to answer 'to a certain degree' unlike Boolean logic (one extreme or the other). Computers are allowed to think more 'human-like'. However, it is true only to a certain degree. In fuzzy logic, machines think in degrees and it can solve problems in the cases where there is no simple mathematical model and it solves highly non-linear processes. Fuzzy logic uses expert knowledge to make decisions and a block diagram of fuzzy logic system is shown in Figure 2.5.

![Figure 2.5 Block diagram of fuzzy logic system](image)

Fuzzy logic was first invented as a representation scheme and it acts as calculus for uncertain or vague notions and also it allows more human-like interpretations. Fuzzy logic has put reasoning in machines by resolving intermediate categories between notations like true/false, hot/cold etc. Fuzzy logic is a problem-
solving control system methodology. It lends itself to implementation in systems ranging from small, simple, embedded micro-controllers to large, multi-channel, networked PC or workstation-based data acquisition control systems etc. It can be implemented in software, hardware or a combination of both. Fuzzy logic provides a simple way to arrive at a definite conclusion. Conclusion is based upon ambiguous or vague, noisy, imprecise or missing input information. Fuzzy logic's approach to control problems simply mimics how a person will make efficient decisions much faster.

**Principles of Fuzzy logic**

The step for designing a simple fuzzy logic control system is as follows:

- Identify the variables (Input, states and output) of the plant.
- Partition the universe of discourse or the interval spanned by each variable into a number of fuzzy subsets, assigning each a linguistic labels(subset includes all the elements in the universe).
- Assign or determine a membership function for each fuzzy subset.
- Assign the fuzzy relationship between the “inputs” or the “states” fuzzy subsets on the one hand and the “output” fuzzy subsets on the other hand, thus forming the rule base.
- Choose appropriate scaling factor for the input and variables in order to normalize the variable to the [0,1] or [-1,1] interval.
- Fuzzy inputs to the controller.
- Use fuzzy appropriate reasoning to input the output contributed from each rule.
- Aggregate the fuzzy outputs recommended by each rule.
- Apply defuzzification to form a crisp output.

It is quite important to define the control objectives and control criteria by answering the following questions.

- What is to be controlled?
- What has to be done to control the system?
- What kind of response should be there?
- What are the possible failure modes in the systems?
It is necessary to determine the input and output relationships. A minimum number of variables are chosen for input to the fuzzy logic inference engine typically error and rate-of-change-of-error. Using the rule-based structure of fuzzy logic, the control problem is broken down into a series of IF X AND Y THEN Z rules. Rules must define the desired system output response for the given system input conditions. Number and complexity of rules depends on the number of input parameters be processed. It also depends upon the number fuzzy variables associated with each parameter and it is preferred to use at least one variable and its time derivative. A single instantaneous error parameter should be used along with its rate of change. Fuzzy logic membership functions need to be created which define the meaning (values) of input / output terms that are used in the rules. System need to be tested, evaluated for results. Tune the rules and membership functions, until satisfactory results are obtained, retest the system.

2.5 FUZZY INFERENCE SYSTEM

Fuzzy Inference Systems (FIS) are rule-based systems and it is based on fuzzy set theory and fuzzy logic. FIS are mappings from an input space to an output space and it allows constructing structures which are used to generate responses (outputs) for certain stimulations (inputs). Response of FIS is based on stored knowledge (relationships between responses and stimulations). Knowledge is stored in the form of a rule base and rule base is a set of rules. Rule base expresses relations between inputs of system and its’ expected outputs and knowledge is obtained by eliciting information from specialists. These systems are usually known as fuzzy expert systems. Another common denomination for FIS is fuzzy knowledge-based systems and it is also called as data-driven fuzzy systems.

FIS are usually divided into two categories viz. Multiple Input and Multiple Output (MIMO) systems and Multiple Input and Single Output (MISO) systems. The MIMO systems return several outputs based on the inputs which it receives. MISO systems are those where only one output is returned from multiple inputs. MIMO systems are decomposed into a set of MISO systems which work in parallel. In terms of inference process, there are two main classes of FIS.
Comparison of Sugeno Type FIS and Mamdani-type FIS

Both Sugeno and Mamdani FIS can be used to perform the similar tasks. Rule base and fuzzification remain the same for the variables. There are various defuzzifiers that can be chosen for a Mamdani FIS and these defuzzifiers also originate similar results in a Sugeno FIS and there is a certain overlap between both types of systems. Mamdani FIS is more widely used and it is used for decision support applications because of its intuitive and interpretable nature. Consequents of the rules in a Sugeno FIS do not have a direct semantic mean. This means that they are not linguistic terms. Also, this interpretability is partially lost. Sugeno FIS rules consequents can have many parameters per rule as per input values.

2.6 FUZZY CLUSTERING TECHNIQUES

The process-data modeling technique uses fuzzy logic and statistical clustering. Patterns of system calls must be represented as a model of normal behavior. The fuzzy model should extract the essence of correctness or normalness of a process. Clustering methods group data by centers and clustering techniques partition data into several clusters in such a way that similar objects belong to the same cluster. The cluster centers represent the most normal of sequences and deviations from the centers indicate behavior that is more abnormal.

Fuzzy clustering involves fuzzy logic and fuzzy logic defines what degree of normality a classifier should give to new process-data compared against the database. Fuzzy logic can better help represent the uncertainty that lies in the data. Hence, look at various fuzzy clustering techniques that can determine the true nature of the underlying data to help predict whether new sequences are abnormal or not and to what degree.

In fuzzy clustering, each data element belongs to several partitions to certain degrees. Non-fuzzy clustering techniques generate different partitions to which data elements belong and these partitions are disjoint. In fuzzy clustering, the partitions
are not disjoint. Several previous attempts have tried to create a good fuzzy clustering algorithm. A general high level partitioning fuzzy clustering algorithm called Fuzzy Clustering Algorithm (FCA) is presented by Jain, et al., [1999]. Additionally, a generalization of the Fuzzy C-Means (FCM) algorithm was presented by Bezdek[1981], while an adaptive variant for detecting circular and elliptical boundaries was presented by Dave [1992]. These algorithms have failed when trying to work with large data sets and in addition to these algorithms, focus will be on quantitative data.

In order to handle categorical or qualitative data, Ralambondrainy [1995] represented multiple categorical attributes by using binary attributes to indicate the presence or absence of a category. The binary values were then used in the well-known c-means algorithm. The number of binary values becomes very large when each attribute has many categories. The complexity of the binary feature vector technique was reduced in the k-modes algorithm, introduced by Zhexue Huang [1998]. It reduces the complexity by using a simple matching dissimilarity measure and the algorithm is very sensitive to initialization.

The fuzzy version of the k-modes algorithm was first proposed by Zhexue Huang and Michael K. Ng [1999] as an extension for the fuzzy c-means algorithm. The fuzzy c-means algorithm is the most prominent fuzzy clustering algorithm [Bezdek, J.C. et al., 1984]. The fuzzy k-modes algorithm was developed to cluster large categorical data sets in data mining. They added the element of fuzzy logic to represent better the uncertainty found in the data set.

Unfortunately, the fuzzy k-modes algorithm gets stuck in local optima. Several heuristics have been developed to find the global optima. One heuristic uses fuzzy centroids instead of the hard-type centroids [Kim, D. W. et al., 2004]. Another variation represents categorical data clusters with k-populations [Kim, D. W. et al., 2005]. Still, another way to find the global optimum is presented by Ng et al., [2002]. The authors used a Tabu search technique and these methods include a random restart method, variable neighbor search, tabu search and candidate list search.
Fuzzy Clustering Validation Techniques

The fuzzy k-modes algorithm has a drawback and one must specify the number of clusters beforehand. To combat this drawback by running the fuzzy k-modes for several cluster sizes and picking the best one according to some criterion. This criterion is known as a validity index. Validity indexes measure the fitness of a partition scheme for a given data set and a validity index shows how closely a certain pre-defined number of clusters fit the underlying data set. Good validity scores indicate that the clusters closely model the underlying truth of nature of the data patterns. To use the validity indexes to determine how many clusters should be used in the process-data model.

Halkidi, et al., [2001] identified three approaches to investigate cluster validity. *External criteria* use a pre-defined structure for the data set which reflects the intuitive understanding of the data. The results of the clustering algorithm are evaluated against this model. *Internal criteria* use the quantities of the vectors of the data set themselves. *Relative criteria* compare the structures of several clustering schemes together, usually using the same algorithm but with different parameters. The authors later identified two evaluation criteria for the selection of optimal clustering schemes: *compactness* which involves how close the members of a certain cluster are to each other; and *separation* which shows how far apart the clusters are from each other. Fuzzy clustering validity indices seek clustering schemes where the dataset exhibits a high degree of membership in one cluster. Fuzzy validity indices divide into two general categories. One category uses only the membership values of the data sets to all clusters and the second uses both the membership values and the underlying structure of the data set.

Bezdek et al., [1984] first introduced the partition coefficient and the partition entropy coefficient. Although effective, these indices have several drawbacks. The indices tend to decrease or increase monotonically as the number of clusters increase. In these cases, one should take the point of maximum curvature on the graph. On the other hand, one can take the global minima in certain cases where the number of clusters is low, around the square root of the number of unique strings. The other category of fuzzy clustering validation extracts the knowledge of the underlying structure of the data. The indices use distance as well as membership values. Various
indices of this type include the Xie-Beni index [1991], the Fukuyama-Sugeno index and other indices proposed by Gath and Geva [1989], which are based on hyper-volume and density. All these indices use some sort of distance measure to extract the underlying structure of the data.

A newer validity index for fuzzy qualitative data was found. This index does not contain the problems of converting a known quantitative index to a qualitative index. The index is presented by Tsekouras et al., [2004]. The authors tackle the two problems that the fuzzy k-modes algorithm presents: Φ its sensitivity to initialization and Φ the a priori knowledge of the number of clusters. They design a new three step categorical data-clustering algorithm that can determine the proper number of clusters for the fuzzy k-modes algorithm based on an entropy fuzzy clustering method, the fuzzy k-modes algorithm and a new validity index.

Fuzzy logic is a form of many-valued logic; it deals with reasoning that is approximate rather than fixed and exact. Compared to traditional binary sets (where variables may take on true or false values) fuzzy logic variables may have a truth value that ranges in degree between 0 and 1. Fuzzy logic has been extended to handle the concept of partial truth where the truth value may range between completely true and completely false [Pivnickova L et al., 2013]. Furthermore, when linguistic variables are used, these degrees may be managed by specific functions. Irrationality can be described in terms of what is known as the fuzzjective [N Ahlawat. et al., 2014]

Fuzzy clustering relaxes the requirement that data points have to be assigned to one and only one cluster. In these algorithms data points can belong to more than one cluster and even with different levels of membership. These non-exclusive cluster assignments can represent the database structure in a more natural way, especially when clusters do not have a perfect boundary or what is the same, when clusters overlap. At these overlapping boundaries, the fuzzy membership can indicate the ambiguity of the cluster assignment. There are two major approaches to this gradual cluster assignment: the first one is probabilistic methods where one can find the Fuzzy C-Means algorithm among others. The second one is the possibilistic methods where the Possibilistic fuzzy C-Means (PCM) is placed.
FCM: Fuzzy C-Means

This algorithm allows gradual membership which will be measured as degrees in [0,1] and this makes the data model much more detailed and allows the total model to express how ambiguous or definite the database is. Given that now memberships are fuzzy, they cannot be expressed with only one value or label. Now, they have to be a vector for each point \( x_i \in X \) the length of which is \( c \), the number of clusters:

\[
u_j = (u_{ij}, ..., u_{cj})^T\]

where \( u_{ij} \) represents the membership to each cluster.

In this algorithm, all data are equally included and receives the same weight as all other data, although the distribution of this weight among the clusters differs from one object to another. As consequence, no cluster can contain all data and the membership values have to be normalized for each object. Obviously, the closer a data point lies to the center of a cluster, the higher its degree of membership should be to this cluster. The problem of finding the best partition of the data set not only relies in the optimization of a cost index where computed the sum of all distances from the point to their cluster center is computed, but it is also desired to maximize the degrees of a membership.

PCM: Possibilistic fuzzy C-Means

It is desirable to have the property of the probabilistic membership degrees, although sometimes it can be misleading. High values for the membership of a datum in more than one cluster mean that the point is at the same distance to those clusters. If there is another point with similar characteristics this can suggest that both points are close, but this might not be true. Both points are equally distanced to the two clusters, but they are not close.

The normalization of membership values can lead further to undesired effects in the presence of noise and outliers. The membership values affect the clustering results, since data point weights influence on cluster prototypes. A more intuitive assignment of degrees of membership can be achieved by dropping the normalization.
constraint, avoiding undesirable normalization effects. This last point can be highly desirable if the clusters are considered completely independent from one another.

The main difference between FCM and PCM is that probabilistic algorithms are forced to partition the data exhaustively while the corresponding possibilistic methods are not compelled to do so. Another difference is that probabilistic methods distribute the total membership of the data points while the possibilistic methods are required to determine the data point weights themselves. Probabilistic algorithms attempt to cover all data points with clusters which can be an advantage when the real clusters have this property. In the possibilistic case, there is no interaction between clusters. Given that the initialization of the PCM is much more complex than the initialization of FCM but it performs better, sometimes FCM is used to initialize $\eta$ and after that the possibilistic algorithm is applied. However, this has a high computational cost and some applications make it impractical.

As the distance measure used is the Euclidean distance only spherical shapes can be detected. Some algorithms have been designed to overcome this problem, by modifying the distance computed. All of them can work with both probabilistic or possibilistic methods [J. Abonyi et al., 2007].

**GK: Gustafson-Kessel** [Abonyi. J et al., 2007]

This algorithm replaces the Euclidean distance used in FCM and PCM by the Mahalanobis distance, in order to adopt various shapes and sizes of the clusters. In FCM and PCM, as the distance function used is the Euclidean distance, the shapes of the clusters to be identified are only hyper-spherical. GK models each cluster $D_i$ by both its center, $c_i$ and its covariance matrix $\Sigma_i$. Both parameters have to be learned and the eigenvalues of the matrix $\Sigma_i$ represent the shape of the cluster $D_i$. Specific constraints can be set depending on the requirements of the application. For instance, restricting to axis-parallel cluster shapes by considering only diagonal matrices. This case is preferred when clustering is used for the generation of fuzzy rule systems and the distance function is now defined as:

$$d^2_{ij} = (x_j - c_j)^T \Sigma_i^{-1} (x_j - c_j)$$

Apart from that, the cost function and the update equation for the cluster centers and the update equation for the membership degrees are identical to the FCM
or PCM, depending which approach is being used. GK extracts more information about the data than FCM and PCM, but it is more sensitive to its initialization. Then, a good recommendation could be the use of FCM or PCM for its initialization and afterwards, applying GK.

**FSC: Fuzzy Shell Clustering** [Abonyi. J et al., 2007]

All the algorithms described so far in this section detect shapes like solid objects and thus they are called solid algorithms. Variants of FCM and PCM have been proposed to detect some other shapes like lines, circles or ellipses called as shell algorithms. They extract prototypes that have a different nature from the data points and for that they need to modify the definition of the distance function and some of these algorithms are:

**FCV: Fuzzy C-Varieties** [Abonyi. J et al., 2007]

It detects lines, planes or hyperplanes and each cluster is a subspace defined by a point and a set of orthogonal unit vectors \( D_i = (c_i, e_{i1}, \ldots, e_{iq}) \) where \( q \) is the dimension of the subspace. The distance function is now defined as:

\[
d(x_j, D_i) = \| x_j - c_i \|^2 - \sum_{t=1}^{q} (x_j - c_i)^T e_{it}
\]

This algorithm can also be used for construction of locally linear models of data with underlying interrelations.

**FCQS: Fuzzy C-Quadratic Shells** [Abonyi. J et al., 2007]

This algorithm is able to recognize ellipses, hyperbolas, parabolas or linear clusters. Sometimes, the projections of the circle-shaped clusters form an ellipse where this algorithm is very useful.

Other fuzzy shell algorithms are AFCE to recognize ellipses, FCRS to recognize non-smooth shapes, such as rectangles, FC2RS to recognize rectangles and other non-smooth polygonal shapes.
KFC: Kernel-based Fuzzy Clustering [Abonyi. J et al., 2007]

Kernel learning methods constitute a set of machine learning algorithms that make it possible to extend classic linear algorithms. The kernel variants of fuzzy clustering algorithms further modify the distance function to handle non-vectorial data, such as sequences, trees or graphs, without needing to modify the algorithms themselves. The aim is two-fold: first, they make possible to treat tasks that require a more complex algorithm than a linear one and second, they make it possible to apply algorithms to data that are not described in a vectorial form. More generally, kernel methods can be applied independently of the data nature. Kernel methods are based on an implicit data representation transformation with which the normal space is transformed to feature space. The second principle of kernel methods is that data are not handled directly in the feature space. But they are only handled through their scalar products that are computed by using the initial representation. The algorithms are written only in terms of scalar products between data. Then, the data representation improvement comes from using scalar products based on an implicit transformation of the data.

Applying this approach to clustering, it aims at extracting prototypes that have a different nature from the data points and thus it modifies the concept of distance between points. In the kernel approach, the similarity is computed between pairs of data points and does not involve cluster centers. On the other hand, kernel methods do not have an explicit representative of the cluster and cannot be seen as prototype based clustering methods. The application of kernel methods needs to select the kernel and its parameters and this may be difficult but they are able to cluster a non-vectorial defined object which is one of their advantages.

FCRM: Fuzzy C-Regression Models [Hathaway. RJ and Bezdek. JC, 1993]

Various algorithms are designed, instead to form different groups, to construct a model based on the inputs in order to predict the output from a new input. These algorithms are also called model-based clustering algorithms. FCRM is one of these types which will be better described in below equation. Its cost function basically tries to find a set of fuzzy models to represent the output with a linear combination of the inputs.
Given a data set where each independent input observation \( x_k \) has a correspondent output observation \( y_k \), it is assumed that several lineal models \( c \) describe the relation between the input and the output: 
\[
y^* = f_i(x; \beta_i) + \epsilon_i, \quad 1 \leq i \leq c
\]
and this is known as a switching regression model.

If each object is described by a combination of all or some models, then the problem to solve can be divided into two: finding good estimations of parameters \( \beta_i \) which define \( y^* \) (predicted output) and finding the membership of each object to each one of those models.

Basically, the steps to find the solution are:

1. Initialization of the variables: \( m, U^{(0)}, \epsilon, \) etc. Choice of the error measure \( E_{ik} \).
2. Calculate the values for all \( \beta_i \). This will be done with least squares.
3. Update the membership matrix.
4. Compare the new matrix with the last one, if the sum of changes between the two matrices is smaller than \( \epsilon \) then iteration will stop. Otherwise, go back to 2.

This algorithm is tested to work very well when the correct number of clusters, \( c \), is chosen and when data is distributed following lineal models.

**AFCR: Adaptive Fuzzy Clustering and Fuzzy Prediction Models** [Ryoke. M et al., 1995]

This algorithm is based on the FCRM which could also be classified as a model-based algorithm. It addresses the problem of the shapes of the clusters and they are changed dynamically and adaptively in the clustering process.

The steps to find the solution are very similar to the case of the FCRM:

1. Initialization of the variables: \( m, U^{(0)}, \epsilon, \) etc. Choice of the error measure \( E_{ik} \) and distance measure \( D_{ik} \).
2. Calculate the values for all \( \beta_i \). This will be done applying least squares using the membership matrix.
3. Update the membership matrix and compute \( \alpha_i \).
4. Compare the new matrix with the last one, if the sum of changes between the two matrices is smaller than \( \epsilon \) then iteration will stop. Otherwise, go back to 2.
2.7 DISCUSSION

Various type of clustering techniques such as Hierarchical, Partitional and Model-based algorithms have been presented in this chapter. Overview and steps involved in unsupervised learning algorithms are discussed. The necessity of fuzzy logic concepts was elaborately discussed. Principles, techniques, interference system and validity techniques in fuzzy logic are described. Some of the fuzzy clustering algorithms such as fuzzy c-means, Possibilistic Fuzzy C-Means, Gustafson-Kessel, Fuzzy Shell Clustering, Fuzzy C-Regression Models and Adaptive Fuzzy Clustering and Fuzzy Prediction Models were explained.

The functionalities and steps involved in the k-means, k-modes, STIRR and ROCK algorithms were discussed. Due to the limitations of the k-means algorithm, k-modes algorithm has been developed because k-means algorithm does not support categorical data. It measures the dissimilarities between the two nodes and also minimizes the cost functions [Topchy et al, 2003]. The ROCK is a robust hierarchical clustering algorithm which employs links and not distances when merging clusters. This method extends non-metric similarity measures and cluster with the categorical attributes which produce quality cluster rather than the traditional approaches [Anil K.Jain and Richard C.Dubes 1988]. STIRR is an effective algorithm that clusters the attribute values and it is efficient in processing intra-attribute value clustering [Gibson et al., 1998].

The k-mode algorithm is not applicable for the processing wide range of inputs which is a major limitation and also it requires in-memory operations. When considering the STIRR algorithm, only one data set can pass over and it also requires a linear number of in-memory for processing large dataset operations. The ROCK algorithm is still in lack of efficiency in determining the quality of the similarity measures. In addition, the limitations such as inefficiency on text clustering, similarity analyses and inability to produce more than two clusters of attribute values were explored. This research concerns with analyzing the problem of text clustering like language variability where the same meaning can be phrased in various ways. While matching the similarity, the shorter sentences are less effective. Generally, the text clustering mainly focuses on reducing dimensionality, removing irrelevant data, increasing learning accuracy and improving result comprehensibility.
The solution produced by the existing algorithms like STIRR, ROCK and k-modes are not effective on above mentioned problems. In the high dimensional vector space, every aspect corresponds to a unique keyword. Along with this, the problem of extracting representative sentences from text is also not effective. Towards this end, it is necessary to develop a novel fuzzy based clustering algorithm for categorical clustering.