UNIVERSITY OF AGDER

Data Mining
K-Clustering Problem

Elham Karoussi

Supervisor
Associate Professor Noureddine Bouhmala

This Master's Thesis is carried out as a part of the education at the University of Agder and is therefore approved as a part of this education.

University of Agder, 2012

Faculty of Engineering and Science

Department of ICT

“To the once I love”

Abstract

In statistic and data mining, k-means clustering is well known for its efficiency in clustering large data sets. The aim is to group data points into clusters such that similar items are lumped together in the same cluster. In general, given a set of objects together with their attributes, the goal is to divide the objects into k clusters such that objects lying in one cluster should be as close as possible to each other's (homogeneity) and objects lying in different clusters are further apart from each other.

However, there exist some flaws in classical K-means clustering algorithm. According to the method, first, the algorithm is sensitive to selecting initial Centroid and can be easily trapped at a local minimum regarding to the measurement (the sum of squared errors) used in the model. And on the other hand, the K-means problem in terms of finding a global minimal sum of the squared errors is NP-hard even when the number of the cluster is equal 2 or the number of attribute for data point is 2, so finding the optimal clustering is believed to be computationally intractable.

In this dissertation, to solving the k-means clustering problem, we provide designing a **Variant Types of K-means in a Multilevel Context**, which in this algorithm we consider the issue of how to derive an optimization model to the minimum sum of squared errors for a given data set. We introduce the variant type of k-means algorithm to guarantee the result of clustering is more accurate than clustering by basic k-means algorithms. We believe this is one type of k-means clustering algorithm that combines theoretical guarantees with positive experimental results.

Acknowledgement

This Master Thesis was submitted in partial fulfilment of the requirements for the degree Master of Science in Computer Science and Engineering. The project work was carried out at the University of Agder, Faculty of Engineering and Science, Grimstad. The task has been under the Supervision of Associate professor Nourddine Bouhmala at the University Of Agder.

I am very pleased to be able to acknowledge the contributions made by all those who have assisted and supported me in my research.

First and foremost, I should like to gratefully appreciate my supervisor Associate professor Nourddine Bouhmala for his valuable comments and enthusiastic support. Without his support, this work would not have been done. I also thank him for his very careful reading and insightful suggestions during the writing of this Thesis, and his thoughtful guidance during my graduate study. I am indebted to Associate professor Nourddine Bouhmala for the optimization knowledge he gave to me and helpful suggestions on, my thesis.

And also, I appreciate MR. Terje Gjøsæter (PHD student in UIA) and MR .Alireza Tadi who guide me in implementing code.

And also I appreciate my husband for his friendly help during my master studies, and for the pleasant and exciting working environment he built.

At last, my special thanks also go to my parents for everything they gave me, their love, understanding, encouragement, and support.

Grimstad

May 2012

Elham Karoussi

Contents

List of Figures

List of Tables

Chapter 1

1. Introduction

These days the role of data generation and collection, are producing data sets from variety of scientific disciplines. The ease with the possibility of whenever, wherever and whatever should be gathering data, has become an article of faith that collecting data will have value. For introducing the field of data mining and clustering data, which we try to cover in this report, by providing an overview of Data mining, clustering analysis and the prevailing clustering methods .

1.1 Background

Data mining is the process of automatically finding useful information in large data repositories [1].The purpose of deploying data mining techniques is discovering important patterns from datasets and also provide capabilities to predict the outcome of a future observation such as market basket analysis, means that by using "*Association Rules"* learning the supermarket can determined which products are frequently bought together or to predict if the new customer will spend more than 100 $ for shopping today at the store.

As for the Wikipedia definition, data mining involves six common tasks: Anomaly Detection, Association rule learning, Classification, Clustering and Regression. In this paper we discussed mostly on clustering class.

Clustering is the most important unsupervised-learning problem as every problem is of this type. The main purpose is finding a structure in a collection of unlabelled data. Totally, the clustering involves partitioning a given dataset into some groups of data whose members are similar in some way. The usability of cluster analysis has been used widely in data recovery,

text and web mining, pattern recognition, image segmentation and software reverse engineering.

K-means clustering as the most intuitive and popular clustering algorithm, iteratively is partitioning a dataset into K groups in the vicinity of its initialization such that an objective function defined in terms of the total within-group sum-of-squares is minimized. However, there exists several flaws; not only regarding the sensitivity of the algorithm to initialization, the algorithm's success in finding globally optimal partitions depends on its starting values. Several different initialization results have been proposed for the K-means algorithm and can be easily trapped at a local minimum regarding to the measurement (the sum of squared errors) used in the model. Also, on the other hand, it has been proved finding a minimal sum of squared errors, is NP-hard even when the number of cluster is equal 2. [2]

The scope of this thesis is: first we try to find the optimal function in terms of minimizing the within-group sum-of-squares errors, the best recovery of the true groupings and also replacing the optimal function with SSE function. The study also provides additional insight into the circumstances in which different methods perform better.

In the rest of this thesis: we try to implement one method to provide optimum clustering and attempting to solve the K-means problem. This method is one of the variant types of K-means, in Multilevel Context, which we first reduce the number of data points by selecting randomly each two data point and calculate their average. Second step is to cluster the data set in lowest layer of reduction, by applying the basic k-means algorithm. Then during each layer, by changing data points between clusters, we find out the minimum possible sum of squared errors.

1.2 Thesis Outline

This thesis organizes as follows: in this chapter we review the technique of data mining and its origins to solving the challenges of traditional data analysis technique and briefly, the four

core's of data mining and the measure of similarity and dissimilarity of data. It provides an essential foundation for data analysis that will be covered in the next chapter.

In chapter 2 we begin the technical discussion with an overview of definitions of clustering problems and cluster analysis (one of the data mining's core aspects). Cluster analysis is also discussed in this chapter which describes different types of clusters and an overview of clustering algorithms.

Chapter 3 and 4 cover basic K-means and K-means in Multilevel Context respectively. In chapter 3 we provide a foundation by discussing K-means and it's formula for measuring distance (SSE). We use this knowledge in comparison with K-means in multilevel technique in chapter 4 for different results, where we research and define one method to optimize K-means problem with reduction of the number of data points.

Finally in chapter 5, the thesis follows by validating the results of the two types of K-means and K-means in multilevel context.

1.3 What is Data mining?

The history of extraction of patterns from data is centuries old. The earlier method which has been used is Bayes' theorem (1700s) and regression analysis (1800s). [1] In the field of computer technology, using the ever growing power of computers, we develop an essential tool for working with data. Such as, it is being able to work with increasing size of the datasets and complexity. And also an urgent need to further refine the automatic data processing, which has been aided by other discoveries in computer science, means that our ability for data collection storage and manipulation of data has been increased. Among these discoveries of importance, according to Wikipedia, are the neural networks, cluster analysis, genetic algorithm (1950s), decision trees (1960s) and support vector machines (1990s).

Historically the field of finding useful patterns in data has a myriad of names including but not limited to; Data mining, Knowledge Extraction, Information discovery, data archaeology and

data pattern processing. Statisticians use the term of Data mining and also is very popular in the field of databases. The terms of knowledge discovery in databases was introduce at the first KDD workshop in 1989 (Piatetsky-Shapiro 1991) which has been used in the AI and machine-learning fields. [5]

As definition, Data mining or important part of Knowledge Discovery in Database (KDD), used to discover the most important information throughout the data, is a powerful new technology. Across a myriad variety of fields, data are being collected and of course, there is an urgent need to computational technology which is able to handle the challenges posed by these new types of data sets.

The field of Data mining grows up in order to extract useful information from the rapidly growing volumes of data. It scours information within the data that queries and reports can't effectively reveal.

As we mentioned earlier, the integral part of knowledge discovery in database (KDD) is data mining, which in our view, KDD refers to the overall process of discovering useful knowledge from data, and data mining refers to a particular step in this process. The KDD role is to convert raw data into suitable information as shown in figure 1.1:

This process contains a series of transformation steps, from data pre-processing to data mining results. [1]

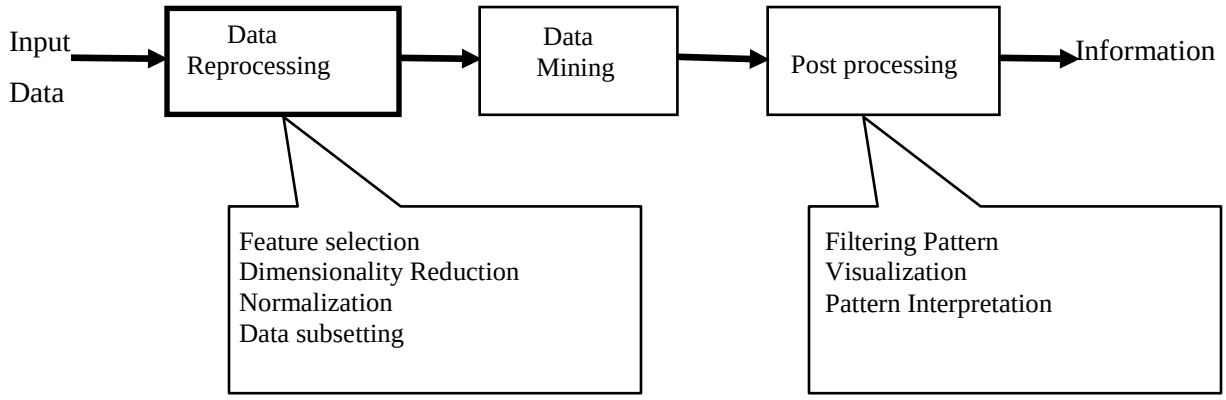


Figure 1.1 The overall steps of the process of Knowledge Discovery in Database (KDD)

The type of stored data as **Input Data** involves flat files, spread sheets or relational tables, and may be residing in a centralized Data Repository[1] or distributed across multiple sites.

In order to transform raw data into the appropriate format, **Pre-processing** Phase has been done to subsequent analysis. This step includes fusing data from multiple sources, removing nose and duplicate observations to cleaning purpose, select relevant features and record to data mining task. This step is the most time-consuming and laborious because of the many different types of data.

The phase **Post-processing** ensures that only valid and useful results are integrated and incorporated into decision support system. The example of this step is visualization, that allows the analysts to explore the data and the data mining results from a variety of viewpoints. Statistical and hypothesis methods can also be applied during this step to eliminate incorrect data mining results.

If the learned patterns do not meet the desired standards, then it is necessary to re-evaluate and change the pre-processing and data mining. If the learned patterns do meet the desired standards then the final step is to interpret the learned patterns and turn them into knowledge.

For more, as result validation in post-processing, in the final step of knowledge discovery from database data mining algorithm verify the patterns produced in the wide data set. Of course all patterns found by the data mining algorithms are not necessarily valid. The main purpose for the algorithms is to find patterns in the training set which are not present in the general data set (the definition of over fitting). To do this purpose use attest set of data for evaluation on which algorithm was not trained. Then compare desire output with the result of the learned patterns are applied to this test set. As an example, in the field of distinguishing spam from legitimate e-mails would be trained on a set of sample e-mails that first trained, the learned patterns would be applied to the test set which had not trained ,then the accuracy measure from the number of emails they correctly classify.

[1] Data Repository is a logical (and sometimes physical) partitioning of data where multiple databases which apply to specific applications or sets of applications reside. For example, several databases (revenues, expenses) which support financial applications (A/R, A/P) could reside in a single financial Data Repository [4]

In the terms of utilization of data mining in the society it can be notice that, although, the task of data mining is still in the first stage, there exist lots of companies in a wide range of industries in the fields of finance, health care, manufacturing transportation, and aerospace, which are already using data mining tools and techniques to take advantage of historical data. Data mining helps analysing relationship, trends, patterns, exception and unusual data that might be unnoticed by using pattern recognition technologies, statistical and mathematical techniques to sift through warehoused information.

In businesses, discovering patterns and relationships in the data by using data mining in order to make better business decision. It can help to predict customer loyalty or develop accurate marketing campaigns.

As example, specific uses of data mining include: [3]

- Market segmentation - Identify the common characteristics of customers who buy the same products from your company.
- Customer churn - Predict which customers are likely to leave your company and go to a competitor.
- Fraud detection - Identify which transactions are most likely to be fraudulent.
- Direct marketing - Identify which prospects should be included in a mailing list to obtain the highest response rate.
- Interactive marketing - Predict what each individual accessing a Web site is most likely interested in seeing.
- Market basket analysis - Understand what products or services are commonly purchased together; e.g., beer and diapers.
- Trend analysis - Reveal the difference between typical customers this month and last.

1.3.1 The Origins of Data Mining

As we mentioned before there are challenges in traditional data analysis techniques and always new types of datasets. In order to cope with these new challenges, researchers have been

developing more efficient and scalable tools that can more easily handle diverse types of data. In particular, data mining draws upon ideas such as:

1- Sampling ,estimating and hypothesis testing from statistic
2- Search algorithms, modelling techniques and learning theories from artificial intelligence, pattern recognition and machine learning.

And also data mining has been adopting from other areas, such as: optimization, evolutionary computing, information theory, signal processing, visualization and information retrieval. [6]

Another area, also supporting this field of research, is the use of databases in order to provide support for storage, index and query processing. Figure 1.2 represents relationship of data mining with other areas.

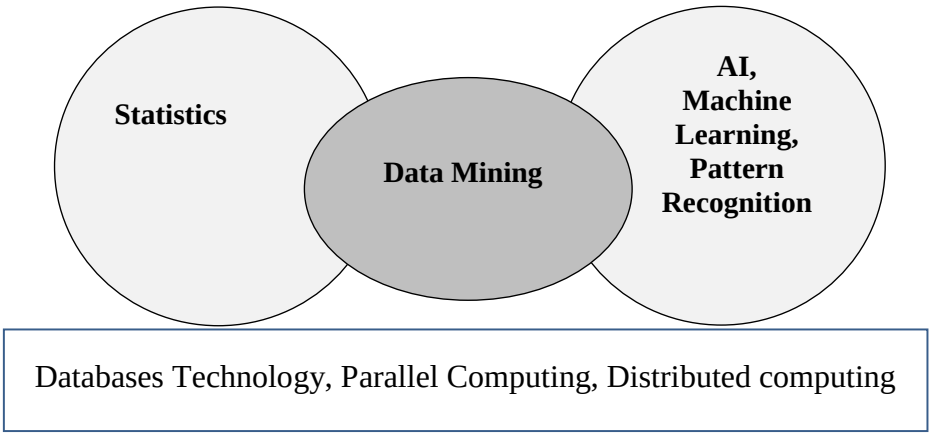


Figure 1.2 Data mining as a confluence of many disciplines

1.3.2 Data Mining Tasks

Data mining tasks are generally divided into two major categories: [7]

Predictive tasks:

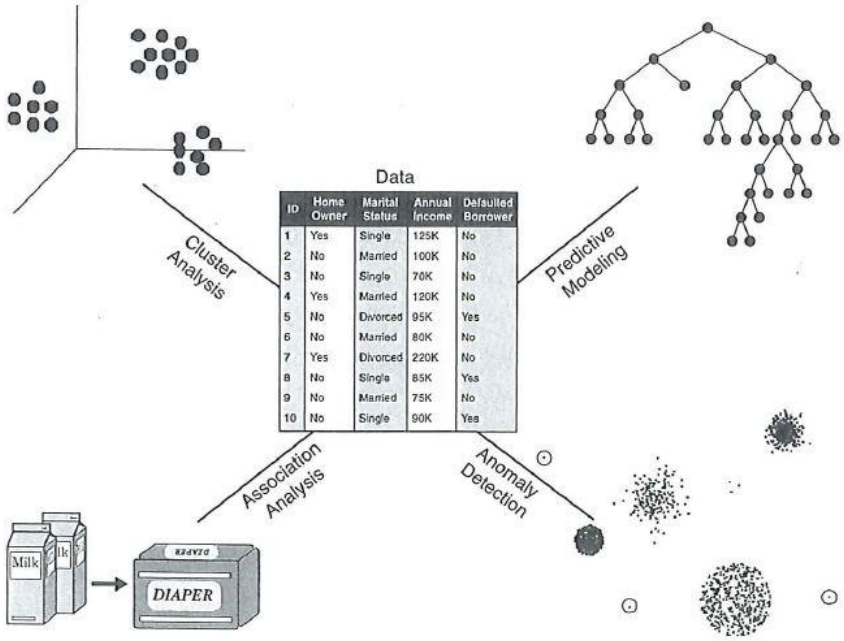
The goal of this task is to predict the value of one particular attribute, based on values of other attributes. The attributes used for making the prediction is named the *Explanatory* or

independent variable and the other value which is to be predicted is commonly known as the *Target* or *dependent value.*

Descriptive task:

The purpose of this task is surmise underlying relations in data .This task of data mining, values are independent in nature and frequently require post-processing to validate results.

Four of the common classes in data mining tasks illustrated in the figure 1.3. Cluster analysis is described in the next part:



ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Figure 1.3 the four core of Data mining task

The total six core data mining task includes: [7]

In predictive modelling task, there are two types, **Classification** that **is** used for discrete target variable and **Regression** which is used for continues target variable. The goal of both tasks is to build a model that produce minimum error between predict and real value. An example of classification is the classification of an email as legitimate or spam, or in an online book store

predict if a web-user will do a purchase or not. On the other hand, forecasting on the future of a price is a Regression task, because the price is a continues-value attribute.

Association Rule Learning (Dependency modelling) is a method that describes associated features in data, searching for relationships between variables. As an example, Web pages that are accessed together can be identified by association analysis.

Anomaly Detection (Outlier/change/deviation detection), this class identifies anomalies or outlier data records which cause errors, or might be of interest and requires further investigation. Another class is **Clustering,** which is the task to discover groups and structures in the data which in some aspect is "similar" or "dissimilar", without using known structures in the data (this task will be discussed more in detail in the next part of the report).And the last class, **Summarization**, attempts to provide a more compact representation of the data set, including visualization and report generation. [5]

Chapter 2

2. Clustering problem

Dividing objects in meaningful groups of objects or classes (cluster) based on common characteristic, play an important role in how people analyse and describe the world. For an example, even children can quickly label the object in a photograph, such as buildings, trees, people and so on.

In the field of understanding data we can say clusters are potential classes, and cluster analysis is a studying technique to find classes. [9] Before discussing about clustering technique we need to provide a necessary description as a background for understanding the topic. First we define cluster analysis and the reason behind its difficulties, and also explain its relationship to other techniques that group data. Then explain two subjects, different ways of grouping a set of objects into a set of clusters and cluster types.

2.1 What is Cluster Analysis?

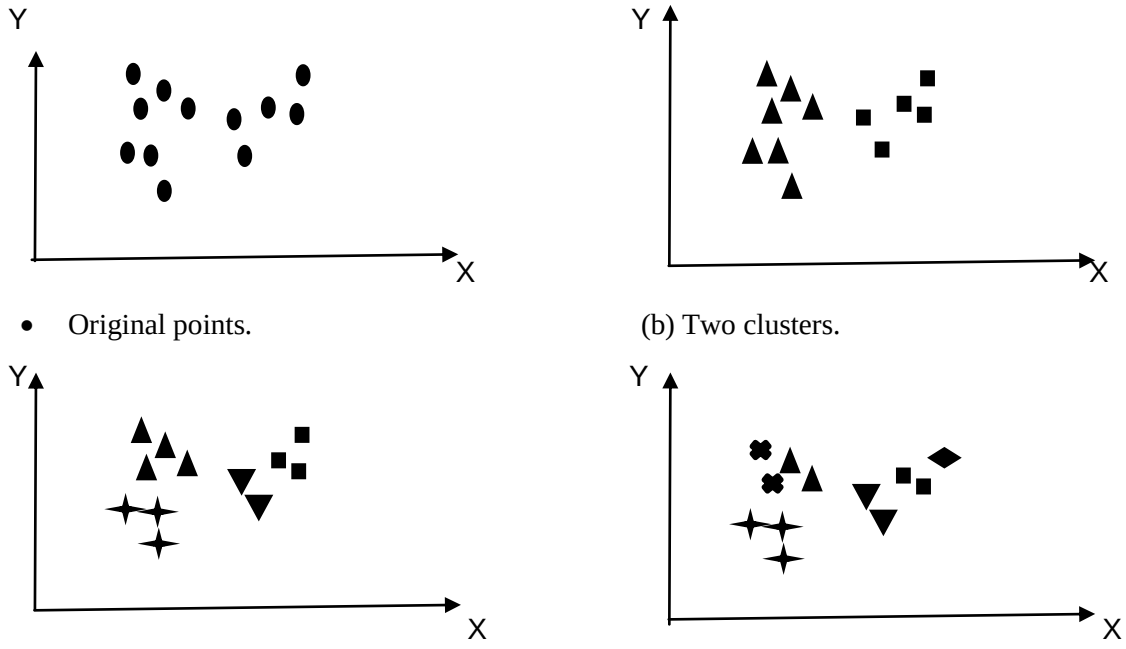
Cluster Analysis technique as a field grew very quickly with the goal of grouping data objects, based on information found in data and describing the relationships inside the data. The purpose is to separate the objects into groups, with the objects related (similar) together and unrelated with another group of objects. It is being applied in variety of science disciplines and has been studied in myriad of expert research communities such as machine learning, statistic, optimization and computational geometry. [8]

The following are some examples:

- **Biology.** Biologists when they a long time ago created a taxonomy (hierarchical classification) made a form of clustering according to genus, family, species and so on.

But also recently they have applied clustering to analyse the myriad amount of genetic information, such as a group of genes that has similar functions.

- **Information Retrieval.** The World Wide Web consists of billions of web pages that are accessed with the help search engine queries. Clustering can be used to create small clusters of search results.
- **Psychology and Medicine.** Clustering techniques are used to analyse frequent conditions of an illness and identifying different subcategories. For example, clustering is used to identify different types of depression, and cluster analysis is used to detect patterns in the distribution/spread of a disease.
- **Business.** In this field there exists a large amount of information on current and potential customers. Clustering helps to group customer activities, as previously mentioned in detail.



- Original points.

(b) Two clusters.

(c) Four clusters.

(d) Six clusters.

Figure 2.1 Different ways of clustering the same set of points

In a lot of research and in many applications, the cluster is not well defined. The figure 1.4 is for understanding this concept:

Assume we have twelve points and three different ways to dividing them into clusters. The figure represent that the definition of clustering is imprecise, grouping data depends on desired result and nature of data.

It is important to notice difference between discriminant analysis (supervised classification) and clustering (unsupervised classification). In clustering task, given a collection of unlabelled data, it should be grouped into more meaningful clusters. And also a label is assigned to each cluster. In contrast, in case of supervised classification, a collection of already labelled entities (called training set) are given.

When predefined labels are available for the data sets, new unlabelled patterns classify into one of the predefined groups associated with the labels. Typically, by using a training set it tries to find a classification scheme which can be used to label or predict new objects into the group. For this reason, cluster analysis refers to unsupervised classification. However the term of classification without any qualification within data mining refer to supervised classification.[9]

And also, the articles *Segmentation* and *Partitioning* are used as an synonym of clustering. The terms partitioning sometimes refer to graph divided to sub graph and segmentation are used for dividing data into group using simple technique, such as grouping people based on their income.

2.2 Clustering Algorithms

Cluster analysis is the general task to be solved which means that, it is not one specific algorithm. It is an result of various algorithms itself, in order to be efficient at clustering. It is distinguished by various type of clustering: **Hierarchical** (nested) versus partitioned (unnested), **Exclusive** versus overlapping versus fuzzy, complete versus partial.

Hierarchical versus partial will be discussed more among different clusters, whether the set of clusters is nested or unnested. In more traditional terminology, it have known as hierarchical

or partitional. A partitional is a division of one data set exactly in one subset. Each collection of clusters in figure 1.4 (b-d) individually is partitional clustering.

If the cluster has sub clusters it obtains the hierarchical clustering, which is a set of nested clusters that are organised as a tree. The main node (root) is a cluster and each node is an sub cluster, except leaves which sometimes are singleton cluster of individual data objects.

In figure 1.4 (a) is a main cluster and each division convert to sub cluster, part (b) has two subclasses, and also part (d) has three subclasses.

Exclusive versus overlapping versus fuzzy: All the clustering shown in figure 1.4 are exclusive, because each object is assigned to one cluster. The definition of overlapping or non-exclusive is that one or several data objects belongs to more than one class, such as one person at university that is can be both an enrolled student and an employee of the university.

In a fuzzy clustering, each object in each class has membership weight between 0 to 1 which means that it does not belong and belong respectively. With a different definition the cluster is treated as a fuzzy set[2]. Of course this type of clustering does not address a true multi-class situation as the previous example. This type always converts to exclusive clustering by assigning each object to a cluster in which its weight is highest.

Complete versus partial: The definition for partial is that some objects, even if part of a large cluster may not be of interest, so we only include objects of interest. In contrast with a complete cluster that assigns every object to a cluster, regardless of importance and interest. Many times some object in one cluster is uninteresting background and doesn't need to search such as when we wants to follow one series and the cluster of the story is full of common article that we need just desire subject the partial clustering is desired.[9]

Totally, the main idea of clustering analysis is to organize data as a grouping of individuals or as a hierarchy of groups by abstracting underlying structure. It follows naturally that clustering methods can be grouped into one of two main categories depending on the strategies used in clustering: partitioning methods or hierarchical methods.

[2] Mathematically fuzzy set is one which an object belongs to any set with a weight between 0 and 1in fuzzy clustering the sum of weights for each object must equal 1 that is imposed additionally constraint.

In general when we make an comparison between hierarchical algorithm and partitioning methods, the fact is that hierarchical algorithms cannot provide optimal partitions for their criterion. Which is in contrast with partitioning method.[1] However, partitional methods assume given the number of clusters to be found and then look for the optimal partition based on the objective function. As we mentioned earlier, the most important difference between hierarchical and partitional approach is that hierarchical methods produce a nested series of partitions while partitional methods produce only one.

Figure 2.2 from [14] gives a good taxonomy for the different methods for clustering analysis.

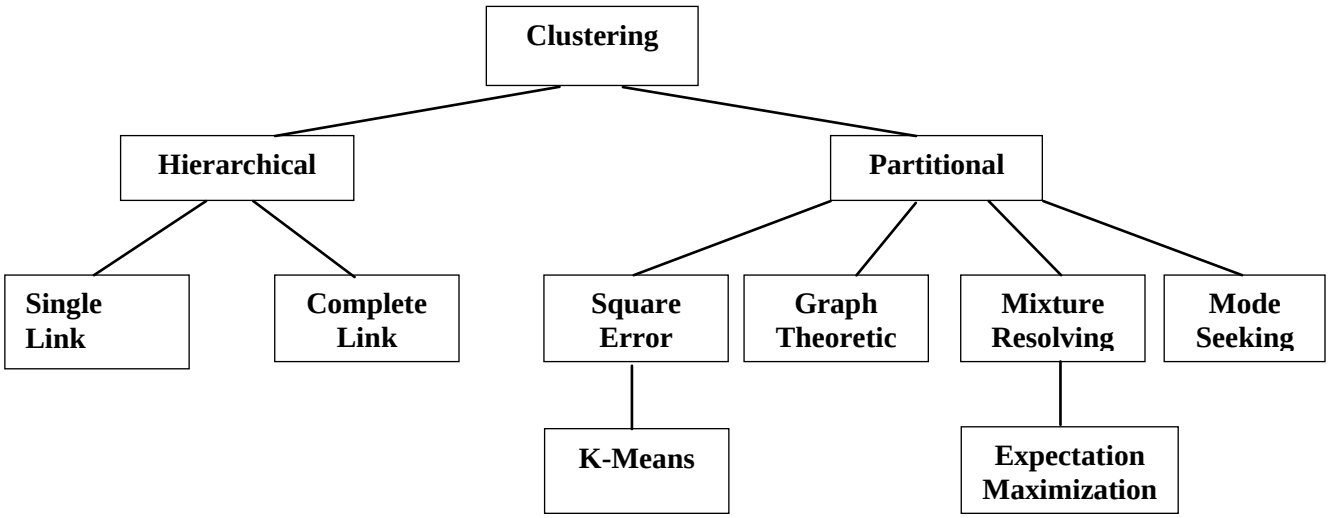


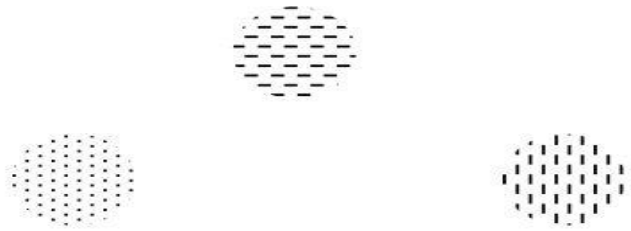
Figure 2.2 Taxonomy for different type of clustering analysis

Chapter 3 discuses in more detail with regards to K-means in partitioning method, and focuses on the existing approaches that are related to the work in this thesis.

2.3 Type of cluster

When usefulness of the data analysis is defined according to its goals, clustering is suggested to finding useful groups of objects. All these clusters are equally valid for each type of data. [10]

- **Well-separated:** Each point is closer to all of the points in its cluster than any point to another cluster. The distance between any two points in different cluster larger than any distance within the group.

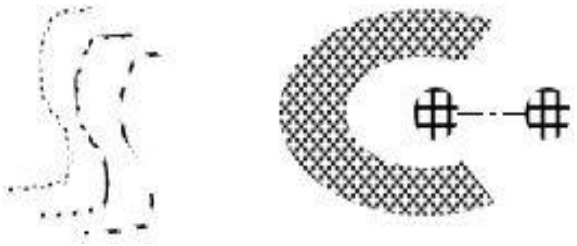


(a)Well-Separated

- **Prototyped-Based**: (canter-based/ centroid- based) In this type of cluster is a set of objects to similar or more closer to the prototype that defines the cluster than to prototype in another group. For continues data prototype is centroid (i.e. the mean of all points) and when the data has categorical attributes the prototype is medoid (i.e. the most representative point of cluster).

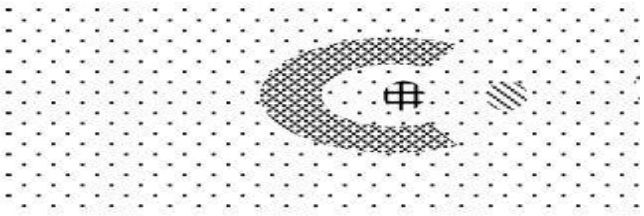
(b) Prototyped-Based

- **Graph-Based**: (contiguity-based) Each point is at least closer to one point in its cluster than to any point in another cluster.



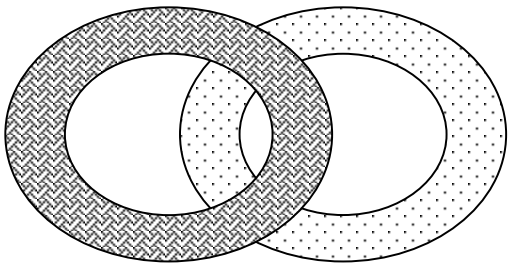
(c) Contiguity-Based

- **Density-based:** A cluster is a dense region of objects that is surrounded by a region of low density This definition is more often used when the clusters are irregular and when noise and outliers are present. The figure (d) shows that some density-based cluster created by adding noise to the data in figure (c).



(d) Density-Based

- **Conceptual (shared property):** A cluster is a set of objects that share some property.
  This definition cinctured all the definitions of previous clusters. Points in a cluster share some general property that derives from the entire set of points (point in the intersection in figure belong to both).



(e) Conceptual Cluster

2.4 Literature Review

Remaining part of this chapter gives a brief overview about the previous techniques regarding K-means, which is used in partition clustering methods and also variation of k-means.

2.4.1 Prior work

Historically as Wikipedia references, the terms of K-means clustering algorithm was first developed by J. MacQueen (1967) and then the idea was followed by J. A. Hartigan and M.A.Wong around 1975. The standard algorithm as a technique for pulse-code modulation was proposed the by Stuart Lloyd in 1957, though it wasn't published until 1982.

The consideration of K-means was demonstrated as early as 1956 by Steinhaus [15]. A simple local heuristic for problem was proposed in 1957 by Lloyds [16]. The method represents that first step choosing k arbitrarily point as facilities. In each stage, assign each point X into cluster with closest facility and then computes the centre of mass for each cluster. These centres of mass become the new facilities for the next phase, and the process repeats until the solution stabilizes. [17, 18]

In aspect of how the neighbours are computed for each centre there exists some different procedures for performing K-means: [22]

- Lloyd's: Repeatedly applies Lloyd's algorithm with randomly sampled starting points.

- Swap: A local search heuristic, which works by performing swaps between existing centres and a set of candidate centres
  This algorithm iteratively changes centres by performing swaps. Each run consists of a given number (max swaps) executions of the swap heuristic.

- Hybrid: A more complex hybrid of Lloyd's and Swap, which performs some number of swaps followed by some number of iterations of Lloyd's algorithm. To avoid getting trapped in local minima, an approach similar to simulated annealing is included as well.

- EZ Hybrid: A simple hybrid algorithm, which does one swap followed by some number of iterations of Lloyd's.

2.4.2 K-means Variation

There exists variation type of K-means methods in different dissertations: [23, 24, 30, 26, 27, 28, 32 and 33].

- Fuzzy C-Means Clustering [23,24]
  Is one type of K-means clustering, where each data points has a degree of belonging to each cluster in fuzzy logic. The algorithm similar to K-means but the centroid has different formula, because in c-means, the centroid of a cluster is the mean of all points, weighted by their degree of belonging to the cluster:

$$\mathbf{c_i} = \frac{\sum_x \mathbf{w_i}(x)x}{\sum_x w_i(x)} \qquad (2.1)$$

  The $\mathbf{w_i}$ is the set of coefficients giving the degree of being in the $\mathbf{i}^{th}$ cluster for each data point x.

- Gaussian mixture models trained with expectation-maximization algorithm (EM algorithm)
  In this method, instead of deterministic, maintain probabilistic assignment to the cluster and also, multivariate Gaussian distribution is used instead of means.

- K_means++[30]

  Between several methods have been proposed [25] for initializing value in k-means, this method is defined as a best algorithm until now. Sometimes because of poor initialization, k-means give poor result of clustering, to avoiding this problem k-means++ is represented.

- The algorithm uses kd-trees as filtering to speed up each k-means step.[26]

  Kd-trees is a binary tree where each node is k-dimensional point. In fact, as Wikipedia definition, is a space-partitioning[3] data structure for organizing points in a k-dimensional space. It's a very useful data structure for several applications, such as multidimensional searching (e.g. nearest neighbour search).

- Some methods attempt to speed up each k-means step using coresets[4] [27] or the triangle inequality.[28]

  Coresets has been recently represented as a powerful tool for approximating various measure of appoint in set P. It computes quickly a small subset Q of P named a \em coreset [27]. It approximates the original set P and solves the problem on subset Q by using relatively algorithm. The solution for Q is then translated to an approximate solution to the original point set P.

- Escape local optima by swapping points between clusters.[32]
- The Spherical k-means clustering algorithm is suitable for directional data. [33]

[3] Space partitioning divides a space into non-overlapping regions. Any point in the space can then be identified to lie in exactly one of the regions.

- The Minkowski metric weighted k-means deals with the problem of noise features by assigning weights to each feature per cluster. [34]

Partitional algorithms optimizing this criterion are also known as Squared Error Algorithm, the classical K-means algorithm is a typical example of the Squared Error Algorithm**,** which in the next chapter is discussed briefly.

Chapter 3

3. K-means Algorithm

The simple definition of k-means clustering, as mentioned earlier, is to classify data to groups of objects based on attributes/features into K number of groups. K is positive integer number. K-means is Prototype-based (center-based) clustering technique which is one of the algorithms that solve the well-known clustering problem. It creates a one-level partitioning of the data objects.

K-means (KM) define a prototype in terms of a centroid, which is the mean of a group of points and is applied to dimensional continuous space. Another technique as prominent as K-means is K-medoid, which defines a prototype that is the most representative point for a group and can be applied to a wide range of data since it needs a proximity measure for a pair of objects. The difference with centroid is the medoid correspond to an actual data point. [13]

As advantages [21] using K-means, there is some idea which find in one paper that referenced [19] to J. MacQueen:

> "*The process, which is called "k-means", appears to give partitions which are reasonably efficient in the sense of within-class variance, corroborated to some extend by mathematical analysis and practical experience. Also, the k-means procedure is easily programmed and is computationally economical, so that it is feasible to process very large samples on a digital computer.*"

And the other is Likewise idea [20] which summarized in introduction part of his work benefits of using K-means:

> "*K-means algorithm is one of first which a data analyst will use to investigate a new data set because it is algorithmically simple, relatively robust and gives "good enough" answers over a wide variety of data sets."*

Totally the analysis in aspect of optimality of K-means defines into two different components:

- Optimal Content for given (cluster membership optimal) :
  Each point is a member of the cluster to whose representative point, it is closest.
- Optimal Intent for given Content :
  Each cluster's representative point is the centroid of its member points, for more, the similarity define according to point that select. [29]

In the cluster memberships optimal, the concept of optimization is simple where any feature even out of original dataset is considered as a member of the cluster.

3.1 The Basic K-means Algorithm

The k-means clustering technique is one of the simplest algorithms; we begin with a description of the basic algorithm:

We assume we have some data point, D=(X1… Xn), first choose from this data points, K initial centroid, where k is user-parameter, the number of clusters desired. Each point is then assigned to nearest centroid. For many, we want to identify group of data points and assign each point to one group.

The idea is to choose random cluster centres, one for each cluster. The centroid of each cluster is then updated based on means of each group which assign as a new centroid.

We repeat assignment and updated centroid until no point changes, means no point don't navigate from each cluster to another or equivalently, each centroid remain the same.

K-means formally described by Algorithm3.1:

Algorithm 3.1: Basic K-Mean Clustering

1: Choose k points as initial centroid

2: **Repeat**

3: Assign each point to the closest cluster centre,

4: Recompute the cluster centres of each cluster,

5: **Until** convergence criterion is met.

The figure 3.1, shows the operation of K-means, which illustrate how starting with 3 centroid the final clusters are found in four iteration.

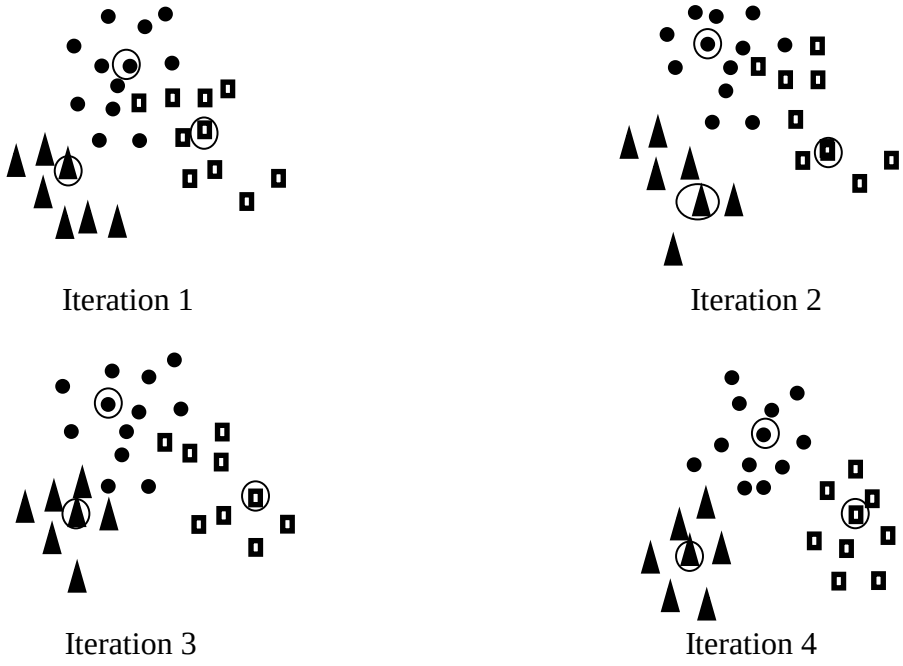


Figure 3.1: Using the K-means algorithm to find three clusters in sample data

The figure for each step shows the centroid at the beginning of the step and assignment of point to those centroids and in second step point assign to updated centroids. In step 2, 3, 4 which are shown in figure 3.1, the centroids move to the small groups of point at the bottom of the figure. In part 4, the K-means algorithm terminates, because no more change occur.

We consider each steps of basic K-means algorithm in more detail and then provide an analysis of the algorithm's space and time complexity.

- Assigning points to closest centroid: [13]

  To assign a point to closest centroid we need a proximity measure that quantifies the closest for the specific data under consideration. Euclidean (L2) distance is often used for data points. However, there may be several types of proximity measures that are appropriate for a given data. As example **Manhattan** (L1) distance can be used for Euclidean data while **Jaccard** measure is often used for documents.
  Sometimes the calculating similarity measure of each point is time consuming; in Euclidean space it is possible to avoiding thus speed up the K-means algorithm.

- Centroid and objective function

  Step 4 in algorithm is "Recompute the centroid of each cluster", since the centroid can be variable depending on the goal of clustering. For example, to measuring distance, minimized the squared distance of each point to closest centroid, is the goal of clustering that depends on the proximity of the point to another, which is expressed by an objective function.

- Data in Euclidean space

  Consider the proximity measure is Euclidean distance. For our objective function we use the **Sum of the Squared Error** (SSE) which is known as scatter. For more, we calculate the error of each data point.

SSE formula is formally defined as follow:

$$SSE = \sum_{i=1}^{K} \sum_{x \in c_i} dist(C_i, x)^2 \qquad (3.1)$$

Where *dist.* is the standard Euclidean (L2) distance between two objects in Euclidean space.

Table 3.1 describe each symbol of formula:

Table 3.1. Table of notation

Symbols	Description
x	An object
C_i	The i th cluster
c_i	The centroid of cluster i
c	The centroid of all points
m_i	The number of object in i th cluster
m	The number of object in data set
K	The number of cluster

Using the notation in table 3.1, the centroid (mean) of the i th cluster is defined by Equation 3.2.

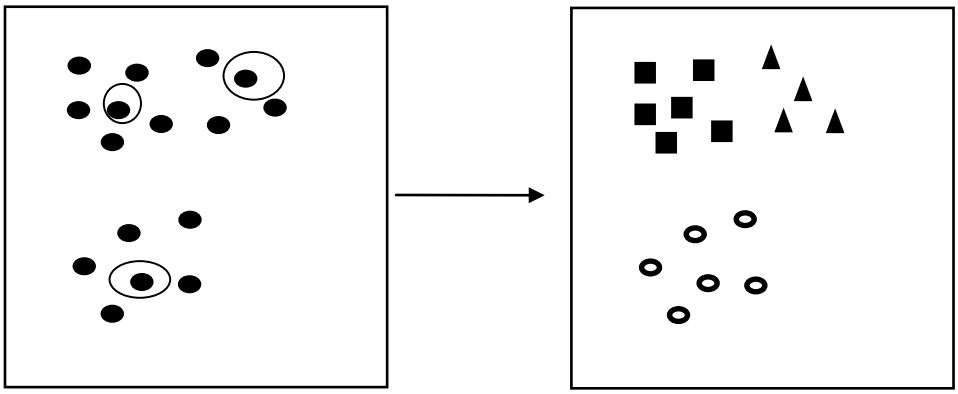
$$c_i = \frac{1}{m_i} \sum_{x \in c_i} x \qquad (3.2)$$

Step 3 and 4 in algorithm directly attempt to minimize the SSE. Step 3 forms group by assigning points to nearest centroid which minimized SSE for the given centroid. And Step 4 recomputes the centroid so as the further minimize the SSE.

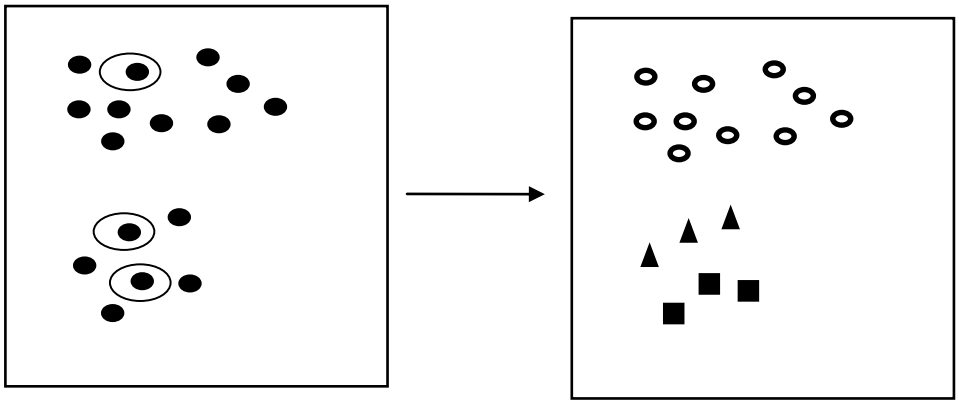
3.2 Choosing initial Centroid

When random initialization of centroid is used, different runs of K-means have different output in total SSEs. A common approach is to choose initial centroid randomly, but the choosing maybe is poor and that it caused poor results (with higher squared error).

Figure 3.2 illustrate a very simple example of three clusters and 16 points. Where part (a) Indicates the "natural" clustering due to the initial centroids is well distributed. In contrast part b is the results of selecting the initial centroid poorly which is a "less natural"



(a) Good starting centroids and a "natural" clustering



(b)Bad starting centroids and a "less natural" clustering

Figure 3.2 Initial Centroid

Another technique that is commonly used to address the problem of choosing initial centroid is to perform multiple runs, which each with randomly chosen initial centroids, and then select the set of cluster with minimum SSE.

As we mentioned earlier, a K-means clustering of a dataset is K-means optimal whenever each point is closer to its own cluster's centroid than to another available cluster's centroid. The closeness is measured in terms of Euclidean distance within the dataset's feature space.

3.3 The behaviour of the Algorithm

For validation of the basic K-means algorithm first we notice the behaviour of the algorithm by using our experimental results, which will be discussed briefly in next chapter.

The graph illustrates the behaviour of SSE in each time running the K-means algorithm on data set Iris (Chapter5, Table 5.1).

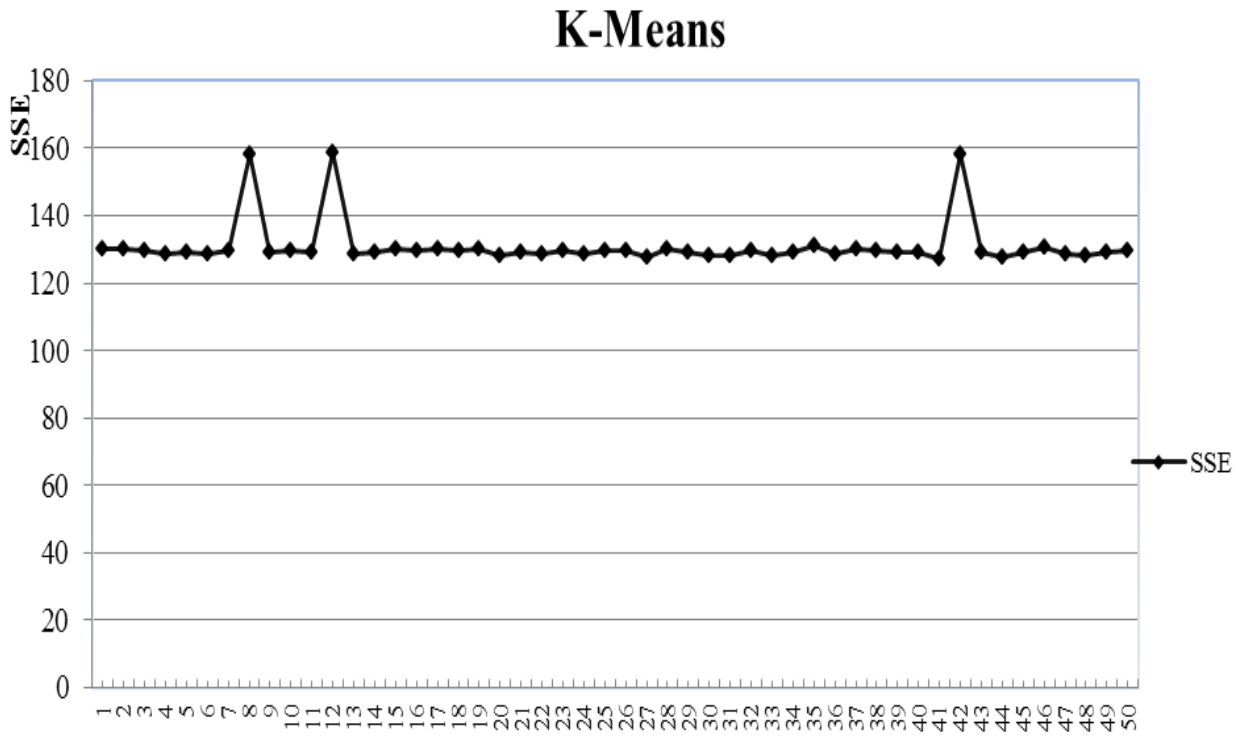


Figure 3.3 Behaviour of SSE for K-means algorithm

From formula 3.1, the SSE is the total summation of distance each data point from its centroid. This graph is for 50 times running K-means on data set Iris, the SSE is almost constant. But sometimes, whenever the poor clustering occurs (Table5.1, Rows: 47and 50) the SSE is maximum amount. (SSE is almost 158). In this time some cluster doesn't have any member because of poor initialize centroid randomly.

Chapter 4

4. Variant of k-means in Multilevel Context

In this chapter, we start discussing the second part of this thesis; implementing an improved algorithm based on K-means in multilevel context.

4.1 The concept of Algorithm

In the k-means algorithm we notice that K-means attempts to minimize the squared or absolute error of points with respect to their cluster centroids. However, this is sometimes a reasonable criterion and gets a simple algorithm; the K-means has still a number of limitations and problems. To getting accurate results, means that to achieve this goal, given a set of objects with their attributes and grouping these objects into K cluster which the objects lying in one cluster should be as close as possible to each other, we are implementing in this dissertation the algorithm 4.1:

Algorithm 4.1: k-means in multilevel context

---

1. Repeat /*loop 1*/
2. Select randomly each two point of data points and calculate average of them and generate one new point (reduction)
3. Until, last reduction has 10 % of whole data points in the first /*****End loop 1*/
4. Run algorithm basic k-means on this new data points
5. Repeat. /*loop 2*/
6. Repeat 10000 times **/***loop 3*/
7. Select two clusters randomly from between K clusters
8. Select from each cluster one data point and swap between two clusters
9. Calculate **SSE** for two clusters

9.1 If the SSE is better (minimum distance) then swap
Else don't change
10. Until 10000 times are done. **/***End loop 3*/
11. Expand each two point in each cluster and go to 6
12. Until back to total (original data points) **/*** End loop 2*/

In the first step of the algorithm, we start to reduce the number of data points. Generate new data points by selecting randomly, each two data point and calculating their mean, this reduction continues until the number of data points is equal or greater than ten percentage of the total number of data points. Of course, the information of each child (generated point) and its parent (the two points whose the new point is generated by) is keept for use later for expanding the group.

After reaching the minimum possible reduction of original data points, run the basic K-means algorithm and group data points to N cluster. Then, first step; on the smallest group of data points after clustering, running the loop (step 7 to 11) to optimizing K-means algorithm with this method in amount of ten thousand times. As we know, each points is the result of two data points (parent) in next step in each cluster we expand each point to its parents, which means , each cluster`s size become 2 times. Then again run the step 7 to 10 of algorithm for new data points. Then continue, until reaching the original number of data points in the first level.

4.2 The behaviour of the Algorithm

The graphs represent the behaviour of SSE after applying the algorithm 4.1; it shows the result after reduction and optimization of SSE, on data set Iris. (Appendix B, calculation 1and 2)

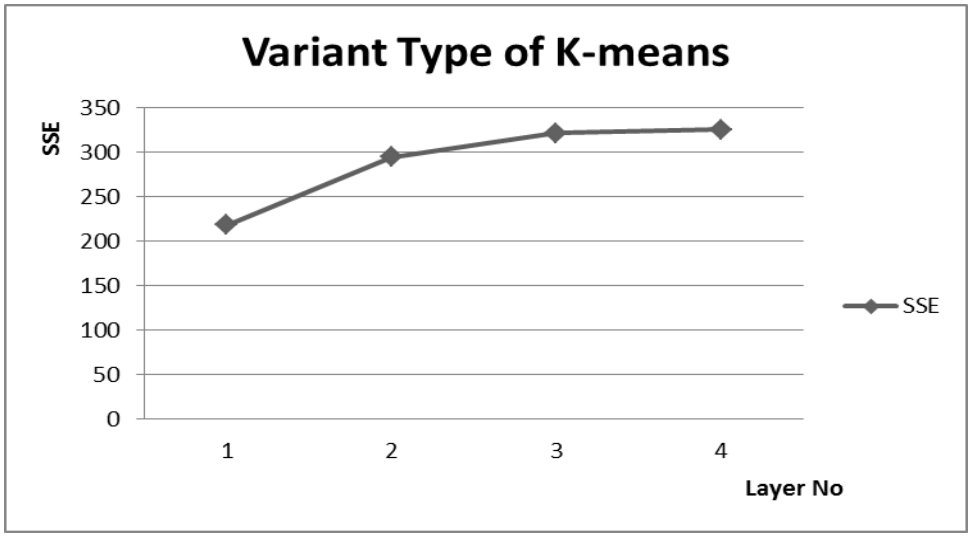


Figure 4.1 Behaviour of variant type of K-means in multilevel context

SSE is reached to minimum possible amount in each layer after running the algorithm. The layer 1 is the last layer of reduction, where the amount of data set is greater than 10% of dataset and because of the small number of data; the SSE is minimum compare with upper layer. Furthermore, because of optimization the SSE represent in each layer doesn't have different value and follow the normal range of data. We believe that this algorithm guarantees that the clustering has optimum results.

Chapter 5

5. Experimental Result

Some preliminary computation results for the previously discussed algorithms is presented in this chapter which is divided into two parts. The first section regards the K-means algorithm proposed in chapter 3 and the second section is about the results of K-means in multilevel context from chapter 4. The code is also represented in **Appendix A.**

5.1 Numerical Result of K-means Algorithm

To test the performance of the algorithm, we have done some preliminary numerical experiments on data sets Iris from the UCI Machine Learning Repository[5]. All experiments have done with Microsoft Visual Studio10, visual basic code on a personal computer. The table 5.1 represent the result of running algorithm 50 times. The program saves this result as CSV format after running , from the Form 1 in Appendix A.

Data set Iris flowers is a multilevel data set with 4 attributes and 3 types of flowers. The dataset contains from each flower species (Iris setosa, Iris virginica and Iris versicolor), 50 sample. The four attributes were measured the length and the width of the sepals and petals, in centimetres.

The optimal result in Iris dataset should be grouping each similar flower into one cluster. The table 5.1 is the result of Iris data set is clustered by K-means algorithm.

[5] http://archive.ics.uci.edu/ml/datasets.html

Table 5.1 Basic K-means Algorithm Result

	Cluster 1	Cluster 2	Cluster 3
Objects (a)	49	57	44
Percentage (b)	90%	26%	16%

(a): The number of object in cluster i^{th}

(b): The percentage of correct similarity in cluster i^{th} (The number of correct / 50 (in Iris data set))

The k-means algorithm group data set into 3 clusters which each cluster has 49,57 and 44 objects which the percentage of similarity with the centroid into each cluster are 90, 26 and 16, respectively. The result of 50 times running K-means algorithm shows in appendix B.

Because of randomly initial centroid in K-means algorithm the results in each running the program are different. Sometimes because of poor choice of initial centroid produce a poor result of clustering. For example, some groups don't have any object (Row 50 and 47, Appendix B).

The resulting behaviour of the code for another data set such as Wine data set from the UCI Machine Learning Repository is represented in Appendix B.

5.2 Numerical Result of K-mean in Multilevel Context Algorithm

The Table 5.2 illustrates the result of K-means in multilevel context in data set Iris; we provide the code for 10 times running the algorithm 4.1 **"**k-means in multilevel context" in Appendix B**.**

Table 5.2 k-means in multilevel context

Layer No(a)	SSE (b)
1	217.9688
2	294.7311
3	321.581
4	325.4503

Cluster1 (c)	Percentage1 (d)	Cluster2 (c)	Percentage2 (d)	Cluster3 (c)	Percentage3 (d)
54	42.59259	64	35.9375	32	28.125

(a): The number of layer : This part depend on the number of reduction is variable

(b): SSE: This column represents the SSE of each layer

(c): The number of object in cluster i^{th}

(d): The percentage of correct similarity in cluster i^{th} (The number of correct / 50 (in Iris data set))

In the experiments for the Iris data set, each cluster should be has at least 50 members that if we notice in Table5.1, the optimal results are not always achieved by k-means basic algorithm. Whereas, in by applying this algorithm the optimum result are gotten in each cluster after 10000 times iteration to find minimum SSE. After applying the algorithm 4.1 on data set Iris, first reduction generate 18 objects in minimum layer (almost more than 10% of 150 objects). Then the basic K-means algorithm run on this layer and produces 3 clusters.

Note: we know that each point itself is the result of an average from two other points, which in turn was two points selected randomly in upper layer.

As an Example:

We assume 18 point in last layer of reduction (data set Iris) :

Layer 1:

Cluster1: X1, X2, X3, X4, X5, X6, X7

Cluster2: X8, X9, X10, X11

Cluster3: X12 , X13, X14, X15, X16, X17, X18

SSE: S1

In each cluster Xi generate by Yn, Ym, which these points maybe relate to different group of data that average of them produce one data point similar to its cluster in last layer.

In this layer we expand each data to parents. And now we have 37 points instead.

Cluster1: X1(Y2, Y5), X2(Y3, Y21), X3(Y4, Y11), X4(Y1, Y6), X5(Y12, Y7), X6(Y13, Y8), X7(Y10, Y18)

Cluster2: X8(Y9, Y14), X9(Y20, Y15), X10(Y17, Y16), X11(Y19, Y23)

Cluster3: X12(Y30, Y24), X13(Y25, Y35), X14(Y26, Y31), X15(Y27, Y28), X16(Y29, Y32), 17(Y33, Y36), X18(Y34, Y22), Y37

Note: Y37 is generated by dividing odd number of data points and in next layer reminds without change from previous layer.

In this time 10000 time select 2 clusters randomly then from each cluster selecting 1 point (Yi) by random also:

Cluster 1: Y30

⇕ (Swap group)

Cluster 3: Y5

We calculate SSE (S1), if the S1 compare with previous value become minimum update the new SSE (S2) and the swap occurs, otherwise the points go back to previous group and SSE stay S1. We have now 3 clusters with 37 points and SSE = Sn, where Sn is minimum possible for this clustering.

We have done the same for 37, 75 and 150 data points and we have reached, layer 2,3 and 4 respectively, and ultimately their optimal SSE, to catch the optimum results where represent in table 5.2.

Chapter 6

6. Conclusion and Future work

In this dissertation, the problem was to solve the K-clustering problem by introducing a clustering technique – Multilevel context of K-means. The problem in clustering as we notice in result part of chapter 5, is because of the nature of K-means method which in the first step the centroids are initialized randomly. Sometimes we have a poor clustering (some clusters don't have any member). The goal is clustering in the best behaviour, which should be to group similar data points as much as possible. But with basic K-means clustering this rarly is the case.

To optimize K-means, we propose an algorithm. Which in this method first take 2 data points of N data points randomly were selected, after calculate average of each this 2 data points, generate one new data point. Then we reduce the size of data point to N/2 and repeat this reduction until the number of data points in last reduction, are equal or greater than 10 % of N. Then we were running K-means algorithm of each layer and also in each layer, 10000 times by exchanging points between clusters and get the minimum SSE, we try to reach to optimal clustering.

There are several different ways to extend our results. First; the current model can deal with only a simple case of basic K-means clustering. The issue of how to deal with general constrained K-means clustering still remains open. It is worth mentioning that in this algorithm instead of choosing randomly each two point, another method for reduction can be used instead. It could be considered a research study in itself to find a method of choosing these two points.

Bibliography

[1] M. S. V. K. Pang-NingTan, "Data mining," in *Introduction to data mining*, Pearson International Edition , 2006, pp. 2-7.

[2] J. Peng and Y. Wei, "Approximating k-means-type clustering via semi definite programming," *SIAM Journal on Optimization*, vol. 18, 2007.

[3] D.Alexander,"DataMining,"[Online].Available: http://www.laits.utexas.edu/~norman/BUS.FOR/course.mat/Alex/.

[4] "What is Data Repository," GeekInterview, 4 June 2008. [Online]. Available: http://www.learn.geekinterview.com/data-warehouse/dw-basics/what-is-data-repository.html.

[5] Fayyad, Usama; Gregory Piatetsky-Shapiro, and Padhraic Smyth (1996) ,*from Data mining to knowledge discovery in data base*

[6]M. S. V. K. Pang-NingTan, "Data mining," in *Introduction to data mining*, Pearson International Edition , 2006 pp. 8.

[7]M. S. V. K. Pang-NingTan, "Data mining," in *Introduction to data mining*, Pearson International Edition , 2006, pp. 7-11.

[8] Han, Jiawei, Kamber, Micheline. (2000) Data Mining: Concepts and Techniques. Morgan Kaufmann

[9]M. S. V. K. Pang-NingTan, "Data mining," in *Introduction to data mining*, Pearson International Edition , 2006, pp. 487-496.

[10] "An Introduction to Cluster Analysis for Data Mining," 2000. [Online]. Available: http://www.cs.umn.edu/~han/dmclass/cluster_survey_10_02_00.

[11] Joaquín Pérez Ortega, Ma. Del Rocío Boone Rojas, María J. Somodevilla García Research issues on, *K-means Algorithm: An Experimental Trial Using Matlab*

[12] J. MacQueen, "Some Methods For Classification And Analysis Of Multivariate Observations," In proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, 1967, pp. 281-297

[13] M. S. V. K. Pang-NingTan, "Data mining," in *Introduction to data mining*, Pearson International Edition , 2006, pp. 496-508

[14] Jain, A.K., Murty, N.M. and Flynn, P.J. (1999) Data Clustering: A Review.ACM Computing Surveys, Vol.31 No.3, pp. 264-323.

[15] V. Braverman, A. Meyerson, R. Ostrovsky, A. Roytman, M. Shindler and B. Tagiku, "Streaming k-means on Well-Clusterable Data," pp. 26-40, 2011.

[16] Stuart Lloyd. "Least Squares Quantization in PCM". In Special issue on quantization, IEEE Transactions on Information Theory, volume 28, PP. 129,137, 1982.

[17] Robert Choate Tryon, Daniel Edgar Bailey. "Cluster Analysis". McGraw-Hill . PP. 147-150 , 1970.

[18] Adam Meyerson and Alex Wong, *Fast and Accurate k-means for Large Datasets*, NIPS, 2011

[19] MacQueen, J.: "Some Methods for Classification and Analysis of Multivariate Observations". Fifth Berkeley Symposium Mathematics Statistics and Probability. Vol.1. Berkeley, CA (1967) 281-297.

[20] Wesan, Barbakh and Colin Fyfe. "Local vs. global interactions in clustering algorithms: Advances over K-means". International Journal of knowledge-based and Intelilligent Engineering Systems 12 (2008).

[21]Joaquín Pérez Ortega1, Ma. Del Rocío Boone Rojas, 1,2, María J. Somodevilla García2, "Research issues on K-means Algorithm: An Experimental Trial Using Matlab"

[22] David M. Mount (2005), *KMlocal: A Testbed for k-means Clustering Algorithms,*
University of Maryland

[23] Nock, R. and Nielsen, F. (2006) "On Weighting Clustering", IEEE Trans. on Pattern Analysis and Machine Intelligence, 28 (8), 1–13

[24] J.-C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, 1981.

[25] Anna D. Peterson, Arka P. Ghosh and Ranjan Maitra,A systematic evaluation of different methods for initializing the K-means clustering algorithm 2010.
http://www.public.iastate.edu/~apghosh/files/IEEEclust2.pdf

[26] Kanungo, T.; Mount, D. M.; Netanyahu, N. S.; Piatko, C. D.; Silverman, R.; Wu, A. Y. (2002). "An efficient k-means clustering algorithm: Analysis and implementation". IEEE Trans. Pattern Analysis and Machine Intelligence 24: 881–892

[27] Frahling, G.; Sohler, C. (2006). "A fast k-means implementation using coresets". Proceedings of the twenty-second annual symposium on Computational geometry (SoCG)

[28] Elkan, C. (2003). "Using the triangle inequality to accelerate k-means". Twentieth International Conference on Machine Learning (ICML).

[29] R. W. Stanforth, "Extending K-Means Clustering for Analysis of Quantitative Structure Activity Relationships (QSAR)," 2008.

[30] D. Arthur, "Analysing and improving local search: k-means and ICP," *Stanford University*, 2009.

[31] C. M. W. Guojun Gan, Data Clustering, Philadelphia, Pennsylvania Alexandria, Virginia: SIAM, American Statistical Association, 2007.

[32] Hartigan, J. A.; Wong, M. A. (1979). "Algorithm AS 136: A K-Means Clustering Algorithm". Journal of the Royal Statistical Society, Series C (Applied Statistics) 28 (1): 100–108. JSTOR 2346830

[33] A. Banerjee, I. S. Dhillon, J. Ghosh and S. Sra, "Clustering on the Unit Hypersphere using," *Journal of Machine Learning Research 6*, p. 1345–1382, (2005).

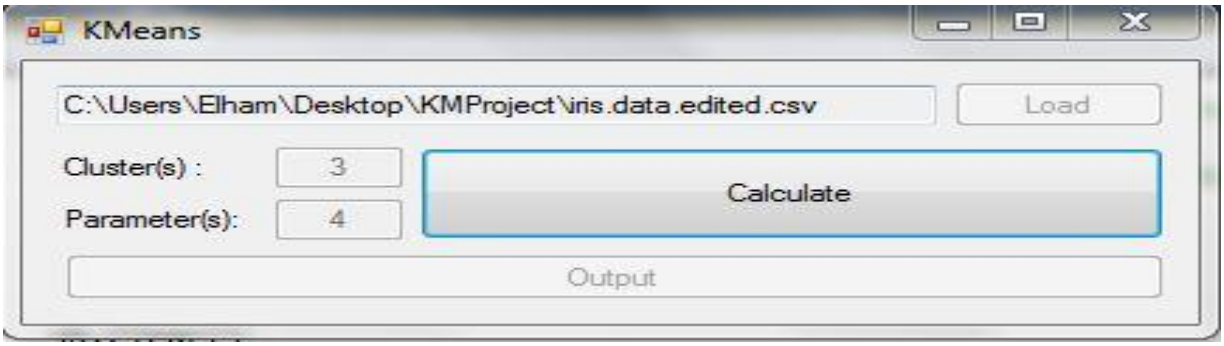
[34] R. C. d. Amorim, "Learning feature weights for K-Means clustering," Department of Computer Science and Information Systems, London, 2011.

Appendices

Appendix A

1. Code listing:

Form 1:



In this form after running the program first press bottom load, require an address of CSV file contains the data set. After loading correctly data set the number of cluster and Parameters (number of attributes) automatically get from data set and fill in text boxes, cluster(s) and parameter(s) respectively.
After pressing calculate bottom the k-means algorithm run and when calculate is done the bottom output active and after click on it the output as CSV file save in the root: "`C:\KMeans_out.CSV`".

Note: The form 1, is just for calculating K-means and in next part (variant type of Kmeans)
We have another form in separate code. This code again repeats in next form.

1.1 K-means source code in VB

```
Public Class Main
 Dim myNode As New ArrayList ()
 Dim RowCount As UInteger
 Dim FieldCount As UInteger
 Dim Distances As New ArrayList ()
 Dim myGroups As New ArrayList ()
 Dim BResult As CResults
 Dim ResArray As New ArrayList ()
 Dim threshold As Integer = 2 ' Inputed threshold for stopping the KMeans algorithm

-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------


Private Sub LoadBtn_Click (ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles LoadBtn.Click

  If OpenFileDialog1.ShowDialog () = vbOK Then' open a dialog for choosing a CSV file
      TextBox2.Text = OpenFileDialog1.FileName 'Set textbox value (path) with path of
                                               Choosed file
          LoadData () ' Run LoadData function
      End If
 End Sub
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

Public Function CalcDistance (ByVal XPoint As KNode, ByVal CPoint As KNode) As Double
      Dim pCount As UInteger
      Dim pPowers As Double = 0

      For pCount = 0 To FieldCount - 2
          pPowers = pPowers + Math.Pow ((XPoint.P (pCount) - CPoint.P (pCount)), 2)
      Next
      'Calculates distance of 2 points
      CalcDistance = Math.Sqrt(pPowers)
      Exit Function
 End Function
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------

Public Sub LoadData ()
      Dim pCount As UInteger
      Dim CountCluster As New ArrayList
      Dim cCounter1, cCounter2 As UInteger

'Extend an object from FileIO using TextBox2.text (path of file) to load a CSV file

  Using MyReader As New Microsoft.VisualBasic.FileIO.TextFieldParser (TextBox2.Text)

          'Set file type to a text delimited type
          MyReader.TextFieldType = Microsoft.VisualBasic.FileIO.FieldType.Delimited
```

```
'Set the delimiter character to Camma
MyReader.Delimiters = New String () {","}

Dim i As Integer = -1 ' Line counter
Dim CurrentLine () As String ' to read all 4 coordinates at one time
Dim tempNode As KNode

'Loop through all of the fields in the file.
'If any lines are corrupt, report an error and continue parsing.
While Not MyReader.EndOfData ' Continue till reaching end of the file
    i = i + 1
    Try
        CurrentLine = MyReader.ReadFields ()
        If Not Is Numeric (CurrentLine (CurrentLine.Length - 1)) Then
                MsgBox ("Invalid cluster type. You should use UInteger as
                cluster umbers. Skipping")
            Exit While
        End If
        tempNode = New KNode
        FieldCount = CurrentLine.Length - 1
        TextBox3.Text = FieldCount

        With tempNode
            .Index = i
            .C = CurrentLine (CurrentLine.Length - 1)
            .Group = -1
            .Selected = False
        End With

        For pCount = 0 To FieldCount - 1
            tempNode.P.Add (Convert.ToDouble (CurrentLine (pCount)))
        Next pCount

        myNode.Add (tempNode)
    Catch ex As Microsoft.VisualBasic.FileIO.MalformedLineException
        'If any line of file was unreadable then show an error message
        MsgBox ("Line " + ex.Message + " is invalid.  Skipping")
    End Try
End While
RowCount = i

For cCounter1 = 0 To RowCount
    If CountCluster.Count > 0 Then
        For cCounter2 = 0 To CountCluster.Count - 1
            If CType(CountCluster(cCounter2), UInteger) =
                  CType(myNode(cCounter1), KNode).C Then
                Exit For
            End If
        Next cCounter2
        If cCounter2 = CountCluster.Count Then

            CountCluster.Add (CType (myNode (cCounter1), KNode).C)
    Else
        CountCluster.Add (CType (myNode(cCounter1), KNode).C)
```

```
            End If
        Next cCounter1
        TextBox1.Text = CountCluster.Count

        If Not MyReader.EndOfData Then ' if you have not reached the end of file then

                                       set textbox path to null
            TextBox2.Text = ""
        Else
            TextBox2.Text = OpenFileDialog1.FileName ' else set the textbox value
                                                with filename which is choosed by user
            If Len (TextBox1.Text) > 0 Then ' If the file is correctly choosed enable
                                                both buttons
                CalcBtn.Enabled = True
                LoadBtn.Enabled = False
            Else
                'Else disable both buttons
                CalcBtn.Enabled = False
                LoadBtn.Enabled = True
            End If
        End If
    End Using
   End Sub
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
     Handles Button1.Click
     Dim i As Integer
     Dim sb As New StringBuilder () ' Extend an string for output
     Dim pCount As UInteger

          Using outfile As New StreamWriter ("C:\KMeans_out.CSV") ' Create a file on

        sb.Clear ()
        sb.AppendLine ("Groups specification")
        For i = 1 To myGroups.Count
            sb.Append ("Cluster" + i.ToString () + "," + "CorrectPercentage" +
                        i.ToString () + ",")
        Next i
        sb.AppendLine ("Calculation time (ms)")
        outfile.Write (sb.ToString ())
        sb.Clear ()

        For i = 0 To ResArray.Count - 1
            'Append cluster members and Elapsed time
            For pCount = 0 To CType (ResArray (i), CResults).ClusterCounts.Count - 1
                sb.Append ((CType (ResArray (i),
                        CResults).ClusterCounts (pCount)).ToString () + ",")
                sb.Append ((CType (ResArray (i), CResults).CorrectCounts (pCount) *
                            100).ToString () + ",")
            Next pCount
            sb.AppendLine ((CType (ResArray (i),
               CResults).CalculationTime).TotalMilliseconds.ToString ())
```

```
                outfile.Write (sb.ToString ())
                sb.Clear ()
            Next i

            sb.Clear ()
            sb.AppendLine ()
            sb.AppendLine ("Points specification")
            sb.Append ("Index,")
            For i = 1 To CType (myNode (0), KNode).P.Count
                sb.Append ("Point" + i.ToString () + ",")
            Next i
            sb.AppendLine ("Initial Cluster,Final Group no,Is Center Point")
            outfile.Write (sb.ToString ())
            sb.Clear ()

            For i = 0 To RowCount
                'Append cluster members and Elapsed time
                sb.Append ((CType (myNode (i), KNode).Index).ToString ())

                For pCount = 0 To FieldCount - 1
                    sb.Append ("," + CType (myNode (i), KNode).P (pCount).ToString ())
                Next pCount
                sb.AppendLine ("," + CType (myNode (i), KNode).C.ToString () + "," +
                  CType (myNode (i), KNode).Group.ToString () + "," + CType (myNode(i),
                   KNode).Selected.ToString ())

                outfile.Write (sb.ToString ())
                sb.Clear ()
            Next

        End Using
    End Sub
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

     Public Sub Kmeans (ByVal Rows As ULong)
        Dim RunAgain As Boolean = True ' For controlling circulations and repeating
                                       Structures
        Dim i, j, g, c, ccnt As UInteger
        Dim XD As XDistance
        Dim myNormal As New ArrayList
        Dim tempnode As KNode
        Dim tempCenter As KNode
        Dim tempIndex As ULong = 0
        Dim mGroup As Groups
        Dim Rand As Double
        Dim pCount As UInteger
        Dim pCondition As Boolean
        Dim StartTime As DateTime
        Dim EndTime As DateTime

        ResArray.Clear ()
        For ccnt = 1 To Math.Abs ((RowCount + 1) / TextBox1.Text)
```

```
StartTime = Now
Distances.Clear ()
myGroups.Clear ()
For i = 0 To myNode.Count - 1
    CType (myNode (i), KNode).Group = -1
    CType (myNode (i), KNode).Selected = False
Next
RunAgain = True

While RunAgain
    pCondition = False
    mGroup = New Groups
    tempCenter = New KNode
    tempnode = New KNode
    XD = New XDistance
    i = 0
    j = 0
    g = 0
    c = 0

    j = Convert.ToUInt16 (TextBox1.Text) - 1
    For i = 0 To j
        'Choosing random points
        VBMath.Randomize () ' pushing seed into Rnd function
        Rand = VBMath.Rnd () * Rows
        CType(myNode(Math.Round(Rand, 0)), KNode).Selected = True ' Choosing
                a random number between 0 and Rows (point 1)
    Next i

    g = 0
    For i = 0 To Rows
        If CType (myNode (i), KNode).Selected = True Then
            If myGroups.Count > 0 Then
                For j = 0 To myGroups.Count - 1
                    If CType (myGroups(j), Groups).Groupno = CType(myNode(i),
                            KNode).Index Then Exit For
                Next j
                'If CType (myGroups (j), Groups).Groupno <> CType (myNode(i),
                    KNode).Index Then
                'End If
            End If
            mGroup = New Groups
            With mGroup
                .GroupCount = 1
                .CorrGroupCount = 0
                .Groupno = CType (myNode (i), KNode).Index
                .Index = g
            End With
            For pCount = 0 To FieldCount - 1
                mGroup.GroupTotal.Add (CType (myNode (i), KNode).P(pCount))
            Next pCount
            myGroups.Add (mGroup)
            g = g + 1
        End If
```

```
Next i

' Distance calculation for each point
'j = 0
For c = 0 To Rows
    If CType (myNode(c), KNode).Selected = False Then
        tempnode = myNode(c)
        tempIndex = 0

        For i = 0 To myGroups.Count - 1
            XD = New XDistance
            XD.CenterPoint = CType (myGroups (i), Groups).Groupno
            XD.DDistance = CalcDistance (tempnode,
                  myNode (CType (myGroups(i), Groups).Groupno))
            Distances.Add (XD)
        Next i

        If Distances.Count > 0 Then
            j = 0
            For i = 0 To Distances.Count - 1
                If CType (Distances (j), XDistance).DDistance >
                  CType (Distances (i), XDistance).DDistance Then
                    j = i
                End If
            Next i
            tempIndex = CType (Distances (j), XDistance).CenterPoint
        End If
        For j = 0 To myGroups.Count - 1
            If CType (myGroups (j), Groups).Groupno = tempIndex Then
                Exit For
            End If
        Next j
        CType (myNode(c), KNode).Group = CType (myGroups (j),
                                                Groups).Index
        'CType (myNode (tempCenter.Index), KNode).Group =
                CType (myGroups (j), Groups).Index

        CType (myGroups (j), Groups).GroupCount = CType (myGroups (j),
           Groups).GroupCount + 1
        If CType (myNode(c), KNode).C = CType (myNode(c), KNode).Group+ 1
        Then
            CType (myGroups (j), Groups).CorrGroupCount =
             CType (myGroups (j), Groups).CorrGroupCount + 1
        End If
        For pCount = 0 To FieldCount - 1
            CType (myGroups (j), Groups).GroupTotal (pCount) =
              CType (myGroups (j), Groups).GroupTotal (pCount) +
                   CType (myNode (c), KNode).P (pCount)
        Next pCount

        Distances.Clear ()
    End If
Next c
```

```
For i = 0 To myGroups.Count – 1

    CType (myNode (CType (myGroups (i), Groups).Groupno), KNode).Group =
         CType (myGroups (i), Groups).Index
Next i

i = 0
While Convert.ToUInt16 (TextBox1.Text) > myGroups.Count + i
    mGroup = New Groups
    With mGroup
        .GroupCount = 0
        .CorrGroupCount = 0
        .Groupno = 0
        .Index = myGroups.Count + i
    End With
    For pCount = 0 To FieldCount - 1
        mGroup.GroupTotal.Add (0)
    Next pCount
    myGroups.Add (mGroup)
    i = i + 1
End While

'If we have got a new normal point and it is not so near the last one
       (according the threshold parameter)
If myGroups.Count > 0 Then
    For i = 0 To myGroups.Count - 1
        tempnode = New KNode

        tempnode.Index = i
        tempnode.Group = -1
        tempnode.Selected = False

        For pCount = 0 To FieldCount - 1
          tempnode.P.Add (CType (myGroups (i),
         Groups).GroupTotal (pCount) / CType (myGroups (i),
                                          Groups).GroupCount)
        Next pCount

        myNormal.Add (tempnode)

        For pCount = 0 To FieldCount - 1
           If Math.Round (CType (myNormal (i), KNode).P (pCount),
            threshold)<> Math.Round (CType (myNode (CType (myGroups (i),
                Groups).Groupno), KNode).P (pCount), threshold) Then
                pCondition = True
                Exit For
            End If
        Next pCount

        If pCondition Then
            'When we are in first layer
            For pCount = 0 To FieldCount - 1
                If CType (myNormal (i), KNode).P (pCount) > 0 Then
```

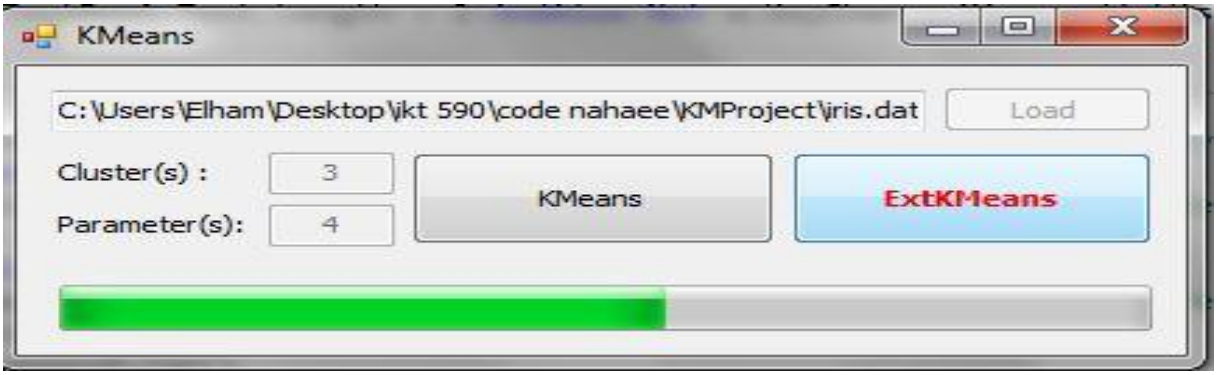
```
                                CType (myNode (CType (myGroups (i), Groups).Groupno),
                                      KNode).P (pCount) = Math.Round (CType
                                      (myNormal (i), KNode).P (pCount), threshold)
                            Next pCount

                            RunAgain = True ' Calculate normal point again
                        Else
                            RunAgain = False ' We have reached the threshold
                        End If
                    Next i
                End If

                If RunAgain Then
                    Distances.Clear ()
                    myGroups.Clear ()
                    For i = 0 To Rows
                        CType (myNode (i), KNode).Selected = False
                        CType (myNode (i), KNode).Group = -1
                    Next i

                    'Run calculation of normal point again
                    'Kmeans (myNormal, Rows)
                End If
            End While
            BResult = New CResults

            EndTime = Now
            BResult.CalculationTime = EndTime.Subtract (StartTime)
            For i = 0 To myGroups.Count - 1
                BResult.ClusterCounts.Add (CType (myGroups (i), Groups).GroupCount)
                BResult.CorrectCounts.Add ((CType (myGroups (i), Groups).CorrGroupCount /
                            Math.Abs ((RowCount + 1) / TextBox1.Text)))
            Next i

            ResArray.Add (BResult)
        Next ccnt
        Button1.Enabled = True
    End Sub
    ----------------------------------------------------------------------------------
    ----------------------------------------------------------------------------------


    Private Sub CalcBtn_Click (ByVal sender As System.Object, ByVal e As System.EventArgs)
     Handles CalcBtn.Click
        Kmeans (RowCount)
     End Sub
    ----------------------------------------------------------------------------------
    ----------------------------------------------------------------------------------


    Private Sub TextBox3_KeyPress (ByVal sender As System.Object, ByVal e As
      System.Windows.Forms.KeyPressEventArgs) Handles TextBox3.KeyPress
        If (Not Char.IsNumber (e.KeyChar) AndAlso Not ".,-".Contains(e.KeyChar) AndAlso
             Not e.KeyChar = Microsoft.VisualBasic.Chr (Keys.Back)) Then
```

```
            e.Handled = True
        Else
            If Not TextBox3.Text.Length < 2 AndAlso Not e.KeyChar =
                  Microsoft.VisualBasic.Chr (Keys.Back) Then
              e.Handled = True
        End If
End Sub
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

Private Sub TextBox1_KeyPress (ByVal sender As System.Object, ByVal e As Sys-
tem.Windows.Forms.KeyPressEventArgs) Handles TextBox1.KeyPress
           If (Not Char.IsNumber (e.KeyChar) AndAlso Not ".,-".Contains (e.KeyChar)
            AndAlso Not e.KeyChar = Microsoft.VisualBasic.Chr (Keys.Back)) Then
            e.Handled = True
        Else
              If Not TextBox1.Text.Length < 2 AndAlso Not e.KeyChar =
    Microsoft.VisualBasic.Chr (Keys.Back) Then e.Handled = True
         End If
End Sub
-------------------------------------------------------------------------------------
End Class
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

Public Class LNode
   Public CNode As New KNode
   Public TopIndexLeft, TopIndexRight As Long
End Class
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

 Public Class KNode
    Public Index As UInteger
    Public P As New ArrayList ()
    Public C As UInteger
    Public Group As Integer
    Public Selected As Boolean
 End Class
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

Public Class XDistance
    Public DDistance As Double
    Public CenterPoint As UInteger
End Class
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

Public Class Groups
    Public Index As UInteger
    Public Groupno As UInteger
    Public GroupCount As ULong
    Public CorrGroupCount As ULong
```

```
    Public GroupTotal As New ArrayList ()
End Class
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

Public Class CResults
    Public ClusterCounts As New ArrayList ()
    Public CorrectCounts As New ArrayList ()
    Public CalculationTime As TimeSpan
End Class
```

1.2 K-means in Multilevel Context source code in VB

Form 2:



In this form after running the program first press bottom load, require an address of CSV file contains the data set. After loading correctly data set the number of cluster and Parameters (number of attributes) automatically get from data set and fill in text boxes, cluster(s) and parameter(s) respectively.

The next step press Kmeans bottom to run the algorithm K-means. Then after some times, result as a CSV format saves on the drive in this Address "`C:\KMeans_out.CSV`". The same occurs while select ExtKmeans bottom in the address "`C:\ExtKMeans_out.CSV`".

```
-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------
Public Sub Create_Levels ()
      Dim Rnd1, Rnd2 As UInteger ' To find two not repeated points
      Dim LRow As ULong = RowCount + 1
      Dim accept1, accept2, fin As Boolean ' if the selected point is accepted or not
      Dim K, Level As UInteger
```

```
Dim c As ULong
Dim temNode As LNode
Dim ttnode As KNode
Dim Progress As ULong = 0

Level = 0
Do
    Dim ttempnode As New ArrayList ()
    For K = 1 To Math.Round (CType(myNodes(Level), ArrayList).Count / 2)
        temNode = New LNode
        ttnode = New KNode
        With temNode
            .CNode = ttnode
            .TopIndexLeft = -1
            .TopIndexRight = -1
        End With

        ttempnode.Add (temNode)
    Next K

    myNodes.Add (ttempnode)

    Level = Level + 1
    LRow = Math.Round (LRow / 2)
Loop While (Math.Round (LRow / 2) > Math.Abs ((RowCount + 1) / 10))

Progress = 0
For i = 1 To myNodes.Count - 1
    Progress = Progress + CType (myNodes (i), ArrayList).Count
Next i
ProgressBar1.Maximum = Progress - 1

Level = 0
LRow = RowCount + 1
Do
    LRow = Math.Round (LRow / 2)

    ' Notice that this function is used from Layer 2 to 4
    ' Start calculating level 2 layer of points (75 members)
    c = 0
    fin = False
    Do
        Dim sWatchMain As System.Diagnostics.Stopwatch = New Sys-
        tem.Diagnostics.Stopwatch sWatchMain.Start ()

        Do
            'Choosing first random point
            VBMath.Randomize ()
            Rnd1 = VBMath.Rnd () * (CType (myNodes (Level), ArrayList).Count - 1)
            accept1 = True ' Initially accept the point

            If c > 0 Then ' If it is not the first point of array
                For K = 0 To CType (myNodes (Level + 1), ArrayList).Count - 1
                    'Search the other points in this layer
```

```
                    If (CType (myNodes (Level + 1)(K), LNode).TopIndexLeft = Rnd1)
                    Or (CType (myNodes(Level + 1)(K), LNode).TopIndexRight = Rnd1)
                    Then ' Is the point recently choosed ?
'If yes then do not accept the point and go for next choose
                        accept1 = False
                    End If
                Next K
                If K = CType (myNodes(Level + 1), ArrayList).Count Then Exit Do
            Else
                Exit Do ' Accept the first point and exit do
            End If
        Loop Until accept1


        If accept1 = True Then
            sWatchMain.Stop ()
            With CType (myNodes(Level + 1)(c), LNode)
                With .CNode
                    .Index = c
                    .C = 0
                    .Group = -1
                    .Selected = False
                    .CalcTime = New TimeSpan (sWatchMain.ElapsedTicks)
                End With
                .TopIndexLeft = Rnd1
            End With

            c = c + 1
            fin = True


        Else
            With CType (myNodes (Level + 1) (c), LNode)
                With .CNode
                    .Index = c
                    .C = 0
                    .Group = -1
                    .Selected = False
                End With
                .TopIndexLeft = -1
            End With
            fin = False
        End If
        For K = 0 To CType (myNodes (Level + 1), ArrayList).Count - 1
             If CType (myNodes (Level + 1)(K), LNode).TopIndexLeft = -1 Then fin =
             False
        Next

        If c >= Math.Round (CType (myNodes (Level), ArrayList).Count / 2, 0) Then
            fin = True
        End If
    Loop Until fin

    fin = False
```

```
        c = 0
        Do
            Dim sWatchMain As System.Diagnostics.Stopwatch = New Sys-
            tem.Diagnostics.Stopwatch
            sWatchMain.Start ()

            Do
                'Choosing first random point
                VBMath.Randomize ()
                Rnd2 = VBMath.Rnd () * (CType (myNodes (Level), ArrayList).Count - 1)
                accept2 = True ' Initially accept the point

'If c > 0 Or Level <> myNodes.Count - 1 Then ' If it is not the first point of array
   For K = 0 To CType (myNodes(Level + 1), ArrayList).Count - 1  ' Search the other
   points in this layer
            If (CType (myNodes (Level + 1)(K), LNode).TopIndexRight = Rnd2) Or (CType
            (myNodes(Level + 1) (K), LNode).TopIndexLeft = Rnd2) Then ' Is the point re-
            cently choosed ?
                 ' If yes then do not accept the point and go for next choose
                          accept2 = False
                          'Exit For
                    End If
                Next K
                If K = CType (myNodes (Level + 1), ArrayList).Count Then Exit Do
                'Else
                'Exit Do ' Accept the first point and exit do
                'End If
            Loop Until accept2

            If accept2 = True Then
                sWatchMain.Stop ()
                With CType(myNodes(Level + 1)(c), LNode)
                    With .CNode
                        .Index = c
                        .C = 0
                        .Group = -1
                        .Selected = False
                         .CalcTime = .CalcTime + New TimeSpan(sWatchMain.ElapsedTicks)
                    End With
                    .TopIndexRight = Rnd2
                End With

                c = c + 1
                ProgressBar1.Value = ProgressBar1.Value + 1


            Else
                With CType(myNodes(Level + 1)(c), LNode)
                    With .CNode
                        .Index = c
                        .C = 0
                        .Group = -1
                        .Selected = False
                    End With
```

```
                        .TopIndexRight = -1
                    End With
                End If
                fin = True
                For K = 0 To CType (myNodes (Level + 1), ArrayList).Count - 1
                    If CType (myNodes (Level + 1)(K), LNode).TopIndexRight = -1 Then fin =
                    False
                Next

                If CType (myNodes (Level), ArrayList).Count Mod 2 <> 0 And c = Math.Floor
                (CType (myNodes (Level), ArrayList).Count / 2) Then
                    fin = True
                End If

                If c >= Math.Round (CType (myNodes (Level), ArrayList).Count / 2) Then
                    fin = True
                End If

            Loop Until fin

            For i = 0 To CType (myNodes (Level + 1), ArrayList).Count - 1
                For nz = 0 To Convert.ToUInt16 (TextBox3.Text) - 1
                    Dim nu As Double
                    If CType (myNodes (Level + 1) (i), LNode).TopIndexLeft <> -1 Then
                     nu = CType (myNodes (Level) (CType (myNodes (Level + 1) (i),
                     LNode).TopIndexLeft), LNode).CNode.P(nz)
                    End If
                    If CType (myNodes (Level + 1) (i), LNode).TopIndexRight <> -1 Then
                     nu = nu + CType(myNodes(Level) (CType (myNodes (Level + 1)(i),
                     LNode).TopIndexRight), LNode).CNode.P(nz)
                    End If
                    nu = nu / 2

                    CType (myNodes (Level + 1)(i), LNode).CNode.P.Add (nu)
                Next (nz)
            Next i
            Level = Level + 1
        Loop While (Math.Round (LRow / 2) > Math.Abs ((RowCount + 1) / 10))
    End Sub
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------

Public Sub ExtKMeans_Out (ByVal InputLevel As ULong)
        Dim i As Integer
        Dim sb As New StringBuilder () ' Extend an string for output
        'Dim pCount As UInteger
        'Dim level As ULong
        Dim iSSE As Double
        Dim iCalcTime As TimeSpan

        Using outfile As New StreamWriter ("C:\ExtKMeans_out.CSV", True) ' Create a file


            sb.Clear ()
```

```
sb.AppendLine ("Layer no, SSE, Calculation Time (ms),")
outfile.Write (sb.ToString ())
sb.Clear ()

iSSE = 0
iCalcTime = New TimeSpan (0)
For i = 0 To myNodes.Count - 1
    For j = 0 To CType (myNodes (i), ArrayList).Count - 1
        iSSE = iSSE + CType (myNodes (i) (j), LNode).CNode.Distance
        iCalcTime = iCalcTime + CType (myNodes (i) (j), LNode).CNode.CalcTime
    Next j
      sb.AppendLine ((i + 1).ToString () + "," + iSSE.ToString () + "," +
      (iCalcTime.Ticks / 10000).ToString() + ",")
Next i
sb.Remove (sb.Length - 1, 1)
sb.AppendLine ()
outfile.Write (sb.ToString ())
sb.Clear ()

For i = 1 To Convert.ToUInt16 (TextBox1.Text)
     sb.Append ("Cluster" + i.ToString () + "," + "CorrectPercentage" +
     i.ToString () + ",")
Next i
sb.AppendLine ("SSE, Calculated Time (ms)")
outfile.Write (sb.ToString ())
sb.Clear ()

For i = 0 To myGroups.Count - 1
    'Append cluster members and Elapsed time
    sb.Append ((CType (myGroups (i), Groups).GroupCount).ToString () + ",")
     sb.Append ((CType (myGroups (i), Groups).CorrGroupCount / CType (myGroups
    (i), Groups).GroupCount * 100).ToString() + ",")
    outfile.Write (sb.ToString ())
    sb.Clear ()
Next i

iSSE = 0
For i = 0 To myGroups.Count - 1
    iSSE = iSSE + CType (myGroups(i), Groups).SSE
    CType (myGroups(i), Groups).CalcTime = New TimeSpan(0)
Next i
For j = 0 To myNodes.Count - 1
    For z = 0 To CType(myNodes(j), ArrayList).Count - 1
       CType(myGroups(CType(myNodes(j)(z), LNode).CNode.Group - 1),
       Groups).CalcTime = CType(myGroups(CType(myNodes(j)(z),
       LNode).CNode.Group - 1), Groups).CalcTime + CType(myNodes(j)(z),
       LNode).CNode.CalcTime
    Next z
Next j
For i = 0 To myGroups.Count - 1
    iCalcTime = iCalcTime + CType(myGroups(i), Groups).CalcTime
Next i
sb.AppendLine (iSSE.ToString () + ","+ (iCalcTime.Ticks / 10000).ToString ())
outfile.Write (sb.ToString ())
```

```
            sb.Clear ()
        End Using
    End Sub
    Public Sub Extractor ()
        For gg = 0 To myGroups.Count - 1
            CType (myGroups (gg), Groups).GroupCount = 0
            CType (myGroups (gg), Groups).CorrGroupCount = 0
        Next
        For k = (myNodes.Count - 1) To 0 Step -1
            For j = 0 To CType (myNodes (k), ArrayList).Count - 1
                Dim sWatchMain As System.Diagnostics.Stopwatch = New Sys-
                tem.Diagnostics.Stopwatch
                sWatchMain.Start ()

                If k > 0 Then
                    If CType (myNodes (k) (j), LNode).TopIndexLeft <> -1 Then CType
                    (myNodes (k - 1) (CType (myNodes(k)(j), LNode).TopIndexLeft),
                    LNode).CNode.Group = CType(myNodes(k)(j), LNode).CNode.Group

                    If CType(myNodes(k)(j), LNode).TopIndexRight <> -1 Then CType
                    (myNodes(k - 1) (CType (myNodes(k)(j), LNode).TopIndexRight),
                    LNode).CNode.Group = CType(myNodes(k)(j), LNode).CNode.Group
                End If
                If k = 0 Then
                    For gg = 0 To myGroups.Count - 1
                        If CType(myNodes(k)(j), LNode).CNode.Group = CType (myGroups(gg),
                        Groups).Index + 1 Then
                         CType (myGroups (gg), Groups).GroupCount = CType (myGroups (gg),
                         Groups).GroupCount + 1
                        If CType (myNodes (k) (j), LNode).CNode.C = CType(myNodes(k)(j),
                        LNode).CNode.Group Then
                            CType(myGroups(gg), Groups).CorrGroupCount =
                            CType(myGroups(gg), Groups).CorrGroupCount + 1
                           End If
                        End If
                    Next gg
                End If
                sWatchMain.Stop ()
               CType (myNodes(k)(j), LNode).CNode.CalcTime = CType(myNodes(k)(j),
               LNode).CNode.CalcTime +New TimeSpan(sWatchMain.ElapsedTicks)
            Next j

        Next k
    End Sub
    Public Sub Iteration (ByVal InputLevel As ULong)
        Dim Rand1, Rand2, ChPoint As Double
        Dim Rows As ULong
        Dim XD As XDistance
        Dim j As Integer
        Dim tempIndex As Integer = -1
        Dim tempSSE As Double = 0

        Rows = CType(myNodes(InputLevel), ArrayList).Count - 1
```

```
For i = 0 To 9999
    ' Choosing random points
    VBMath.Randomize () ' pushing seed into Rnd function
    Rand1 = VBMath.Rnd () * (myGroups.Count - 1)
    Do
        VBMath.Randomize () ' pushing seed into Rnd function
        Rand2 = VBMath.Rnd () * (myGroups.Count - 1)
    Loop Until Math.Round (Rand1) <> Math.Round(Rand2)

    VBMath.Randomize () ' pushing seed into Rnd function
    ChPoint = VBMath.Rnd () * Rows

    Rand1 = Math.Round (Rand1)
    Rand2 = Math.Round (Rand2)
    ChPoint = Math.Round (ChPoint)

    Distances.Clear ()

    XD = New XDistance

    XD.CenterPoint = CType (myGroups (Rand1), Groups).Groupno
    XD.DDistance = CalcDistance (CType (myNodes (InputLevel) (ChPoint),
    LNode).CNode, CType(myNodes(myNodes.Count - 1) (CType(myGroups(Rand1),
    Groups).Groupno), LNode).CNode)

    Distances.Add (XD)

    XD = New XDistance

    XD.CenterPoint = CType (myGroups (Rand2), Groups).Groupno
    XD.DDistance = CalcDistance (CType(myNodes(InputLevel)(ChPoint), LNode).CNode,
    CType(myNodes(myNodes.Count - 1)(CType(myGroups(Rand2), Groups).Groupno),
    LNode).CNode)

    Distances.Add (XD)
    tempSSE = 0
    tempIndex = -1

    If Distances.Count > 0 Then
        j = 0
        For di = 0 To Distances.Count - 1
            If CType (Distances(j), XDistance).DDistance > CType(Distances(di),
            XDistance).DDistance Then
                j = di
            End If
        Next di

        tempIndex = CType(Distances(j), XDistance).CenterPoint
        tempSSE = CType(Distances(j), XDistance).DDistance
    End If

    For j = 0 To myGroups.Count - 1
        If CType(myGroups(j), Groups).Groupno = tempIndex Then
            Exit For
```

```
            End If
        Next j


        CType(myNodes(InputLevel)(ChPoint), LNode).CNode.Group = CType(myGroups(j),
        Groups).Index + 1
        CType(myNodes(InputLevel)(ChPoint), LNode).CNode.Distance = tempSSE

    Next i
  End Sub
```

```
Public Sub ExtKmeans (ByVal InputLevel As ULong)
    Dim RunAgain As Boolean = True ' For controlling circulations and repeating
    structures
    Dim i, j, g, c, ccnt As UInteger
    Dim XD As XDistance
    Dim myNormal As New ArrayList
    Dim tempnode As KNode
    Dim tempIndex As ULong = 0
    Dim tempSSE As Double = 0
    Dim mGroup As Groups
    Dim Rand As Double
    Dim pCount As UInteger
    Dim pCondition As Boolean
    Dim once As ULong = 0
    Dim Rows As ULong

    Rows = CType (myNodes (InputLevel), ArrayList).Count - 1
    ResArray.Clear ()
    myNormal.Clear ()

    For ccnt = 1 To Math.Abs ((RowCount + 1) / TextBox1.Text)
        Dim sWatchMain As System.Diagnostics.Stopwatch = New Sys-
        tem.Diagnostics.Stopwatch
        sWatchMain.Start ()

        Distances.Clear ()
        myGroups.Clear ()
        myNormal.Clear ()
        once = 0

        For i = 0 To CType (myNodes (InputLevel), ArrayList).Count - 1
            CType (myNodes (InputLevel) (i), LNode).CNode.Group = -1
            CType (myNodes (InputLevel) (i), LNode).CNode.Selected = False
        Next
        RunAgain = True

        While RunAgain
            pCondition = False
```

```
mGroup = New Groups
tempnode = New KNode
XD = New XDistance
i = 0
j = 0
g = 0
c = 0

If once = 0 Then
    j = Convert.ToUInt16 (TextBox1.Text) - 1
    Do
        For i = 0 To j
            'Choosing random points
            VBMath.Randomize () ' pushing seed into Rnd function
            Rand = VBMath.Rnd () * Rows
             CType (myNodes (InputLevel) (Math.Round (Rand, 0)),
             LNode).CNode.Selected = True ' Choosing a random number be-
             tween 0 and Rows (point 1)
        Next i
        g = 0
        For i = 0 To CType (myNodes (InputLevel), ArrayList).Count - 1
              If CType (myNodes (InputLevel)(i), LNode).CNode.Selected =
              True Then g = g + 1
        Next
        If g <> j + 1 Then
        For i = 0 To CType (myNodes (InputLevel), ArrayList).Count - 1
            CType (myNodes (InputLevel)(i), LNode).CNode.Selected = False
           Next
        End If
    Loop Until g = j + 1
End If

g = 0
For i = 0 To Rows
    If CType (myNodes (InputLevel) (i), LNode).CNode.Selected = True Then
        If myGroups.Count > 0 Then
            For j = 0 To myGroups.Count - 1
              If CType (myGroups (j), Groups).Groupno = CType (myNodes
              (InputLevel) (i), LNode).CNode.Index Then Exit For
            Next j
        End If
        mGroup = New Groups
        With mGroup
            .GroupCount = 1
            .CorrGroupCount = 0
             .Groupno = CType (myNodes (InputLevel) (i), LNode).CNode.Index
            .Index = g
        End With
        For pCount = 0 To FieldCount - 1
             mGroup.GroupTotal.Add (CType (myNodes (InputLevel) (i),
             LNode).CNode.P (pCount))
        Next pCount
        myGroups.Add (mGroup)
        g = g + 1
```

```
    End If
Next i

'Distance calculation for each point
'j = 0
For c = 0 To Rows

      Dim sWatch As System.Diagnostics.Stopwatch = New Sys-
      tem.Diagnostics.Stopwatch
       sWatch.Start ()

        If CType (myNodes (InputLevel) (c), LNode).CNode.Selected = False
        Then
        tempnode = New KNode
        tempnode = CType (myNodes (InputLevel) (c), LNode).CNode
        tempIndex = 0

        For i = 0 To myGroups.Count - 1
            XD = New XDistance
            XD.CenterPoint = CType (myGroups (i), Groups).Groupno
           XD.DDistance = CalcDistance (tempnode, CType (myNodes (In-
           putLevel) (CType (myGroups (i), Groups).Groupno),
           LNode).CNode)
            Distances.Add (XD)
        Next i

        If Distances.Count > 0 Then
            j = 0
            For i = 0 To Distances.Count - 1
               If CType (Distances (j), XDistance).DDistance > CType (Dis-
               tances (i), XDistance).DDistance Then
                    j = i
                End If
            Next i
            tempIndex = CType (Distances (j), XDistance).CenterPoint
            tempSSE = CType (Distances (j), XDistance).DDistance
        End If
        For j = 0 To myGroups.Count - 1
            If CType (myGroups (j), Groups).Groupno = tempIndex Then
                Exit For
            End If
        Next j
         CType (myNodes (InputLevel) (c), LNode).CNode.Group = CType
         (myGroups (j), Groups).Index + 1
        CType (myNodes (InputLevel) (c), LNode).CNode.Distance = tempSSE

        CType (myGroups (j), Groups).GroupCount = CType (myGroups (j),
        Groups).GroupCount + 1
        CType (myGroups (j), Groups).SSE = CType (myGroups (j),
         Groups).SSE + tempSSE
        If CType (myNodes (InputLevel) (c), LNode).CNode.C = CType
        (myNodes (InputLevel) (c), LNode).CNode.Group + 1 Then
        CType (myGroups (j), Groups).CorrGroupCount = CType (myGroups (j),
        Groups).CorrGroupCount + 1
```

```
                    End If
                    For pCount = 0 To FieldCount - 1
                       CType (myGroups (j), Groups).GroupTotal (pCount) = CType
                       (myGroups (j), Groups).GroupTotal (pCount) + CType (myNodes
                       (InputLevel) (c), LNode).CNode.P (pCount)
                    Next pCount

                    Distances.Clear ()
                End If

                sWatch.Stop ()

               CType (myNodes (InputLevel) (c), LNode).CNode.CalcTime = CType (myNodes
               (InputLevel) (c), LNode).CNode.CalcTime + New TimeSpan
               (sWatch.ElapsedTicks)
            Next c

            For i = 0 To myGroups.Count - 1
                CType(myNodes(InputLevel)(CType(myGroups(i), Groups).Groupno),
                LNode).CNode.Group = CType(myGroups(i), Groups).Index + 1
            Next i

            i = 0
            While Convert.ToUInt16 (TextBox1.Text) > myGroups.Count + i
                mGroup = New Groups
                With mGroup
                    .GroupCount = 0
                    .CorrGroupCount = 0
                    .Groupno = 1
                    .SSE = 0
                    .Index = myGroups.Count + i
                End With
                For pCount = 0 To FieldCount - 1
                    mGroup.GroupTotal.Add (0)
                Next pCount
                myGroups.Add (mGroup)
                i = i + 1
            End While

'If we have got a new normal point and it is not so near the last one (according the
threshold parameter)
            If myGroups.Count > 0 Then
                For i = 0 To myGroups.Count - 1
                    tempnode = New KNode

                    tempnode.Index = i
                    tempnode.Group = -1
                    'tempnode.Selected = False

                    For pCount = 0 To FieldCount - 1
                        tempnode.P.Add (CType (myGroups (i), Groups).GroupTotal
                        (pCount) / CType (myGroups (i), Groups).GroupCount)
                    Next pCount
```

```
                myNormal.Add (tempnode)

                For pCount = 0 To FieldCount - 1
                    pCondition = False
                         If Math.Round(CType(myNormal(i), KNode).P(pCount),
                         threshold) <>
                         Math.Round(CType(myNodes(InputLevel)(CType(myGroups(i),
                         Groups).Groupno), LNode).CNode.P(pCount), threshold) Then
                        pCondition = True
                        Exit For
                    End If
                Next pCount

                If pCondition Then
                    'When we are in first layer
                    For pCount = 0 To FieldCount - 1
                      If CType (myNormal (i), KNode).P (pCount) > 0
                      Then
                      CType (myNodes (InputLevel) (CType (myGroups (i),
                      Groups).Groupno), LNode).CNode.P (pCount) = Math.Round
                      (CType (myNormal(i), KNode).P(pCount), threshold)
                    Next pCount


                    RunAgain = True ' Calculate normal point again
                    once = once + 1
                Else
                    RunAgain = False ' We have reached the threshold
                    once = 0
                    Exit For
                End If
            Next i
        End If

        If RunAgain Then
            Distances.Clear ()
            myGroups.Clear ()
            myNormal.Clear ()
            'once = 0
            For i = 0 To Rows
                'CType (myNode (i), KNode).Selected = False
                CType (myNodes (InputLevel) (i), LNode).CNode.Group = -1
                CType (myNodes (InputLevel) (i), LNode).CNode.Distance = 0.0
            Next i
        End If
    End While

    BResult = New CResults

    sWatchMain.Stop ()
     BResult.CalculationTime = BResult.CalculationTime + New TimeSpan (sWatch-
     Main.ElapsedTicks)
    For i = 0 To myGroups.Count - 1
        BResult.ClusterCounts.Add (CType (myGroups (i), Groups).GroupCount)
```

```
                BResult.CorrectCounts.Add ((CType (myGroups (i), Groups).CorrGroupCount /
                Math.Abs ((RowCount + 1) / TextBox1.Text)))
            Next i
            For i = 0 To CType (myNodes (InputLevel), ArrayList).Count - 1
                BResult.SSE = BResult.SSE + CType (myNodes (InputLevel)(i),
                LNode).CNode.Distance
            Next i
            ResArray.Add (BResult)
        Next ccnt
    End Sub
```

```
Private Sub Button2_Click (ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        Dim sb As New StringBuilder ()
        Using outfile As New StreamWriter ("C:\ExtKMeans_out.CSV")
      'Create a file on root of drive C for writing reasons
        End Using
        For k = 0 To 9
            Using outfile As New StreamWriter ("C:\ExtKMeans_out.CSV", True)
        'Create a file on root of drive C for writing reasons
                sb.Clear ()
                sb.AppendLine ()
                sb.AppendLine ("Calculation: " + (k + 1).ToString ())
                outfile.Write (sb.ToString ())
            End Using
            ProgressBar1.Value = 0
            myNodes.Clear ()
            myNode.Clear ()
            LoadData ()
            Create_Levels ()
            ExtKmeans (myNodes.Count - 1)
            Extractor ()
            For i = myNodes.Count - 1 To 0 Step -1
                Iteration (i)
            Next i
            ExtKMeans_Out (0)
        Next k
    End Sub
```

```
Public Class LNode
    Public CNode As New KNode
    Public TopIndexLeft, TopIndexRight As Long
End Class
```

Appendix B

1. Experimental Results Data

K-means

Wine data set from the UCI Machine Learning Repository[6]
This data set has 13 attribute , and 178 instance
1) Alcohol
2) Malic acid
3) Ash
4) Alcalinity of ash
5) Magnesium
6) Total phenols
7) Flavanoids
8) Nonflavanoid phenols
9) Proanthocyanins
10) Color intensity
11) Hue
12) OD280/OD315 of diluted wines
13) Proline

	cluster1 (a)	Percentage 1 (b)	cluster2 (a)	Percentage 2 (b)	Cluster 3 (a)	Percentage 3 (b)	Time (c)
1	57	67.41573	59	55.61798	62	60.67416	392.0225
2	58	64.04494	56	48.8764	64	60.67416	51.0029
3	58	65.73034	120	92.69663	0	0	24.0014
4	59	69.10112	54	52.24719	65	60.67416	95.0032
5	58	67.41573	55	53.93258	65	60.67416	9.0005
6	62	70.78652	52	50.5618	64	60.67416	123.007
7	61	70.78652	54	53.93258	63	60.67416	32.0018
8	65	70.78652	52	48.8764	61	58.98876	34.002
9	52	64.04494	59	55.61798	67	62.35955	97.0055
10	55	64.04494	63	60.67416	60	62.35955	31.0018
11	55	64.04494	63	60.67416	60	60.67416	4.0002

[6] http://archive.ics.uci.edu/ml/datasets/Wine

	cluster1 (a)	Percentage 1 (b)	cluster2 (a)	Percentage 2 (b)	Cluster 3 (a)	Percentage 3 (b)	Time (c)
12	56	64.04494	62	58.98876	60	60.67416	7.0004
13	57	65.73034	61	57.30337	60	62.35955	8.0005
14	57	67.41573	60	58.98876	61	62.35955	7.0004
15	57	67.41573	62	60.67416	59	58.98876	42.0024
16	54	67.41573	62	62.35955	62	60.67416	31.0018
17	53	65.73034	62	62.35955	63	60.67416	12.0007
18	55	65.73034	62	60.67416	61	60.67416	25.0014
19	56	67.41573	63	64.04494	59	60.67416	28.0016
20	56	67.41573	66	69.10112	56	58.98876	36.002
21	57	67.41573	65	65.73034	56	58.98876	7.0004
22	54	64.04494	62	62.35955	62	65.73034	91.0053
23	54	64.04494	60	62.35955	64	65.73034	10.0005
24	53	62.35955	63	62.35955	62	62.35955	15.0009
25	53	62.35955	63	62.35955	62	62.35955	4.0002
26	53	64.04494	63	64.04494	62	64.04494	44.0025
27	51	67.41573	65	67.41573	62	64.04494	30.0017
28	52	60.67416	63	55.61798	63	64.04494	65.0038
29	53	58.98876	66	58.98876	59	58.98876	18.001
30	56	65.73034	60	57.30337	62	60.67416	42.0024
31	56	65.73034	60	55.61798	62	60.67416	4.0002
32	55	64.04494	61	57.30337	62	58.98876	10.0006
33	54	69.10112	60	53.93258	64	57.30337	63.0036
34	55	69.10112	60	57.30337	63	58.98876	14.0008
35	52	67.41573	60	57.30337	66	62.35955	26.0009
36	50	69.10112	61	65.73034	67	65.73034	50.0012
37	52	69.10112	60	62.35955	66	65.73034	11.0006
38	54	70.78652	61	62.35955	63	64.04494	25.0015
39	54	70.78652	61	62.35955	63	64.04494	4.0002
40	53	69.10112	61	60.67416	64	64.04494	7.0004
41	50	65.73034	63	62.35955	65	65.73034	36.0021
42	49	64.04494	63	62.35955	66	65.73034	11.0006
43	50	64.04494	62	62.35955	66	67.41573	6.0003
44	51	65.73034	61	62.35955	66	67.41573	8.0005
45	49	62.35955	63	60.67416	66	65.73034	18.001
46	52	65.73034	61	60.67416	65	64.04494	19.0011
47	52	69.10112	62	64.04494	64	65.73034	24.0013
48	55	69.10112	62	60.67416	61	62.35955	26.0015
49	55	70.78652	62	58.98876	61	58.98876	17.001
50	55	67.41573	62	55.61798	61	57.30337	15.0009

Iris data set

	cluster1 (a)	Percentage 1 (b)	cluster2 (a)	Percentage 2 (b)	Cluster 3 (a)	Percentage 3 (b)	SSE	Time (c)
1	52	96	57	26	41	12	129.7023	17.2067
2	50	92	56	24	44	16	129.2683	2.2374
3	50	92	56	26	44	16	128.689	1.9433
4	51	94	56	24	43	14	129.3069	1.8524
5	50	94	56	26	44	16	129.1282	1.9563
6	50	92	56	26	44	16	128.0786	1.9202
7	51	94	57	26	42	16	128.7349	1.8502
8	52	96	56	24	42	14	130.308	1.8943
9	52	96	56	26	42	16	130.5844	2.0709
10	51	94	55	24	44	16	129.077	1.9724
11	51	94	56	24	43	16	128.1977	2.1602
12	52	96	55	26	43	16	130.4117	1.8194
13	50	92	56	26	44	16	128.9271	1.8567
14	51	94	56	24	43	16	129.402	1.9147
15	50	92	56	26	44	16	129.2284	2.017
16	51	94	57	26	42	16	129.9607	1.7977
17	49	90	57	26	44	16	127.6264	2.032
18	60	94	90	74	0	0	158.0216	1.8657
19	50	92	56	24	44	16	128.817	1.8957
20	51	94	57	26	42	16	129.3237	1.8776
21	51	94	56	26	43	16	128.5336	1.8752
22	51	94	57	26	42	14	129.1604	1.8952
23	52	96	56	26	42	16	129.6465	1.8641
24	51	94	55	24	44	16	127.9965	1.9189
25	51	94	55	26	44	16	129.5532	1.883
26	50	92	56	26	44	16	129.2589	1.8667
27	51	94	55	24	44	16	129.1031	1.8728
28	49	90	57	26	44	16	127.9345	1.905
29	50	92	56	26	44	16	128.4695	2.1306
30	52	96	55	26	43	16	129.4349	1.8291
31	51	94	56	26	43	16	129.6649	1.8719
32	51	94	56	24	43	16	128.7177	1.9012
33	51	94	56	26	43	16	130.0647	1.9433
34	51	94	55	24	44	16	129.75	1.9034

	cluster1 (a)	Percentage 1 (b)	cluster2 (a)	Percentage 2 (b)	Cluster 3 (a)	Percentage 3 (b)	SSE	Time (c)
35	51	94	55	26	44	16	127.4524	1.982
36	50	94	56	26	44	16	128.2718	1.9294
37	50	92	56	26	44	16	129.5011	1.9245
38	52	96	56	26	42	14	129.7968	1.8931
39	51	94	57	26	42	14	129.1652	2.0677
40	51	94	55	26	44	16	129.4159	1.9313
41	51	94	55	26	44	16	129.9618	1.8782
42	51	94	55	26	44	16	129.9886	1.8922
43	52	96	55	26	43	16	130.965	1.9399
44	51	94	55	26	44	16	129.7833	1.9517
45	52	96	55	26	43	16	129.8184	1.8999
46	52	96	56	24	42	16	130.2999	1.9055
47	60	92	90	74	0	0	158.5622	1.8735
48	51	94	57	26	42	16	129.4795	1.9012
49	50	92	56	26	44	16	128.3527	1.8946
50	60	92	90	74	0	0	158.4514	2.016

(a): The number of object in cluster i^{th}

(b): The percentage of correct similarity in cluster i^{th} (e.g. The number of correct / 50 (in Iris data set))

(c): The total time takes to run K-means clustering in millisecond

Variant type of K-means

Iris data set

Calculation : 1							
Layer no	SSE	Time					
1	217.9688	5.5183					
2	294.7311	183.4119					
3	321.581	229.2639					
4	325.4503	354.3113					
Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
54	42.59259	64	35.9375	32	28.125	1.561815	708.6226

Calculation : 2							
Layer no	SSE	Time					
	251.3778	5.5944					
2	328.7124	202.8683					
3	359.1202	249.8959					
4	364.4597	380.2715					
Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
86	33.72093	56	28.57143	8	50	6.320031	760.543

Calculation : 3							
Layer no	SSE	Time					
1	201.6191	6.1034					
2	276.1809	184.0671					
3	306.5602	231.8094					
4	316.5864	388.0524					
Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
54	42.59259	64	37.5	32	43.75	1.612899	776.1048

Calculation : 4		
Layer no	SSE	Calculation Time (ms)
1	217.9475	5.6467
2	282.1524	188.6218
3	307.2358	236.1759
4	313.0625	394.084

Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
80	40	64	31.25	6	16.66667	2.130927	788.168

Calculation : 5		
Layer no	SSE	Calculation Time (ms)
1	234.4526	5.6971
2	315.3763	182.3964
3	348.1499	227.7536
4	352.8541	362.9929

Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
64	26.5625	70	27.14286	16	18.75	1.375163	725.9858

Calculation : 6		
Layer no	SSE	Calculation Time (ms)
1	204.6664	5.7028
2	273.7712	182.7993
3	301.9031	227.9791
4	307.9687	363.3678

Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
72	25	70	28.57143	8	37.5	7.013511	726.7356

Calculation : 7							
Layer no	SSE	Calculation Time (ms)					
1	224.6536	5.8461					
2	305.3532	188.2822					
3	337.362	237.9586					
4	343.1193	364.7035					
Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
30	53.33333	96	37.5	24	41.66667	12.91086	729.407

Calculation : 8							
Layer no	SSE	Calculation Time (ms)					
1	219.3554	5.588					
2	289.1834	184.9003					
3	318.4569	231.1628					
4	325.1562	367.6882					
Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
32	21.875	72	34.72222	46	26.08696	7.466078	735.3764

Calculation : 9							
Layer no	SSE	Calculation Time (ms)					
1	247.2931	5.6332					
2	323.2556	190.854					
3	350.4102	237.0972					
4	356.1533	361.4234					
Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
72	31.94444	8	25	70	34.28571	1.934009	722.8468

Calculation : 10		
Layer no	SSE	Calculation Time (ms)
1	224.8794	5.7813
2	304.8083	182.154
3	341.9277	227.9214
4	348.5273	367.8831

Cluster 1	Percentage 1	Cluster 2	Percentage 2	Cluster 3	Percentage 3	SSE	Time (ms)
56	21.42857	46	26.08696	48	29.16667	2.148743	735.7662