A STUDY ON THE SCALABILITY OF CLASSICAL DATA CLUSTERING K-MEANS ALGORITHM

N.Revathy[1], T.Guhan[2], S.Selvarajan[3]
[1]Associate Professor, Department of Master of Computer Applications,
Hindusthan College of Arts and Science, Coimbatore, India
[2]Assistant Professor (Senior Grade)
Department of Computer Science Engineering,
Sri Ramakrishna Engineering College, Coimbatore, India
[3]Principal, Muthayammal College of Engineering
Rasipuram, Namakkal (Dist), India.

ABSTRACT

Clustering is a well-known problem in statistics and engineering, namely, how to arrange a set of vectors (measurements) into a number of groups (clusters). Clustering is an important area of application for a variety of fields including data mining, statistical data analysis and vector quantization. The problem has been formulated in various ways in the machine learning, pattern recognition optimization and statistics literature. The fundamental clustering problem is that of grouping together (clustering) data items that are similar to each other. The most general approach to clustering is to view it as a density estimation problem. Classification algorithms rely on human supervision to train itself to classify data into pre-defined categorical classes. The term "classification" is frequently used as an algorithm for all data mining tasks. Instead, it is best to use the term to refer to the category of supervised learning algorithms used to search interesting data patterns. While classification algorithms have become very popular and ubiquitous in DM research, it is just but one of the many types of algorithms available to solve a specific type of DM task.

KEYWORDS-Clustering, Scalability, K-Means clustering.

I. INTRODUCTION

The DM and KDD fields are relatively new; different authors appear to survey methods in different ways.

a) Fayyad's methods for data mining: Predictive Modeling; Clustering; Summarization; Dependency Modeling; Change and Deviation Detection
b) Goebel & Grunewald's methods for data mining: Statistical Models; Case-Based Reasoning; Neural Networks; Decision Trees; Rule Induction; Bayesian Belief Networks; Genetic algorithms; Fuzzy Sets; Rough Sets
c) Aggarwal & Yu's survey of techniques for data mining: Association rules, Clustering, Classification

In the course of gathering information for this thesis, Aggarwal and Yu's survey turned out to be very helpful in organizing descriptions of various methods related with data mining. So in this dissertation we classify and approach these techniques as per the Aggarwal and Yu's survey on this field.

II. PROBLEM DEFINITION

Clustering is a well-known problem in statistics and engineering namely, how to arrange a set of vectors (measurements) into a number of groups (clusters). Clustering is an important area of

application for a variety of fields including data mining, statistical data analysis and vector quantization. The problem has been formulated in various ways in the machine learning, pattern recognition optimization and statistics literature. The fundamental clustering problem is that of grouping together (clustering) data items that are similar to each other. The most general approach to clustering is to view it as a density estimation problem. Because of its wide application, several algorithms have been devised to solve the problem. Notable among these are the EM algorithm, neural nets, SVM and k-means. Clustering the data acts as a way to parameterize the data so that one does not have to deal with the entire data in later analysis, but only with these parameters that describe the data. Sometimes clustering is also used to reduce the dimensionality of the data so as to make the analysis of the data simpler.

In one of its forms, clustering problems can be defined as: given a dataset of N records, each having dimensionality d, to partition the data into subsets such that a specific criterion is optimized. The most widely used criterion for optimization is the distortion criterion. Each record is assigned to a single cluster and distortion is the average squared Euclidean distance between a record and the corresponding cluster center. Thus this criterion minimizes the sum of the squared distances of each record from its corresponding center.

Classification algorithms rely on human supervision to train itself to classify data into pre-defined categorical classes. For example, given classes of patients that corresponds to medical treatment responses; identify most responsive forms of treatment for the patient.

The following list shows some of the categories of classification algorithms generally used in data mining applications. In this dissertation those categories of algorithms are going to be addressed in detail. In this dissertation a detailed survey on existing algorithms will be made and the scalability of some of the existing classification algorithms will be examined.

- *k*-Nearest Neighbor algorithm
- Decision Tree.
- DNF Rules
- Neural networks
- Genetic algorithms
- Bayesian networks
- Rough and Fuzzy Sets

The term "classification" is frequently used as an algorithm for *all* data mining tasks. Instead, it is best to use the term to refer to the category of supervised learning algorithms used to search interesting data patterns. While classification algorithms have become very popular and ubiquitous in DM research, it is just but one of the many types of algorithms available to solve a specific type of DM task.

Scalability refers to the ability of data mining algorithms to work under increasingly large databases. Because data mining deals with large databases, scalability is a desirable feature.

Literally, scalability means that as a system gets larger, its performance improves correspondingly. For data mining, scalability means that by taking advantage of parallel database management systems and additional CPUs, you can solve a wide range of problems without the need to change your underlying data mining environment. You can work with more data, build more models, and improve their accuracy by simply adding additional CPUs. Ideally, scalability should be linear or better. For example, if you double the number of CPUs in a parallel system, you can build twice as many models in the same amount of time, or the same number of models in half the time.

III. A REVIEW ON DATA MINING AND KNOWLEDGE DISCOVERY

Data mining has been the subject of many recent articles in business and software magazines. However, just a few short years ago, few people had not even heard of the term data mining. Though data mining is the evolution of a field with a long history, the term itself was only introduced relatively recently in the 1990s.

The roots of data mining can be traced back along three family lines. The longest of these three is classical statistics. Without statistics, there would be no data mining, as these statistics are the

foundation of most technologies on which data mining is founded upon. Classical statistics embrace concepts such as regression analysis, standard distribution, standard deviation, standard variance, cluster analysis, and confidence intervals, all of which are used primarily to study data and data relationships. These are the very building blocks on which more advanced statistical analyses are built upon. Even in today's data mining tools and knowledge discovery techniques, classical statistical analysis still plays a significant role.

The second longest family line for data mining is artificial intelligence, AI. This discipline, which is built upon heuristics as opposed to statistics, attempts to apply human thought-like processing to statistical problems. Because this approach requires vast amounts of computer processing power, it was not practical until the early 1980s, when computers began to offer useful power at reasonable prices. AI found relatively few applications at the very high-end scientific and government markets, and the required supercomputers of the era priced AI out of the reach of virtually everyone else. The notable exceptions were certain AI concepts that were adopted by some high-end commercial products, such as query optimization modules for Relational Database Management Systems. Over the time, this changed, as AI was used to create new ways in addressing and solving very complex and math-driven problems. At the Artificial Intelligence Laboratory at MIT, founded in the 1960s, there is extensive research in many aspects of intelligence. Their aim is two-fold: to understand human intelligence at all levels, including reasoning, perception, language, development, learning, and social levels; and to build useful artifacts based on intelligence.

The third family line of data mining is machine learning, which is more accurately described as the union of statistics and AI. While AI was not a commercial success, and is therefore primarily used as a research tool, its techniques were largely co-opted by machine learning. Machine learning, able to take advantage of the ever-improving price/performance ratios offered by computers of the 1980s and 1990s, found more applications because the entry price was lower than AI. Machine learning could be considered an evolution from AI because it blends AI heuristics with advanced statistical analysis. Machine learning attempts to let computer programs learn about the data they study, such that programs make different decisions based on the characteristics of the studied data, using statistics for fundamental concepts, and adding more advanced AI heuristics and algorithms to achieve its goals.

As such, data mining, in many ways is fundamentally the adaptation of machine learning techniques to business applications. Data mining is best described as the union of historical and recent developments in statistics, AI, and machine learning. These techniques are used together to study data and find previously hidden trends or patterns within. Data mining is finding increasing acceptance in science and business areas that need to analyze large amounts of data to discover trends that they could not otherwise find.

IV. KNOWLEDGE DISCOVERY PROCESS

Data mining is part of a larger iterative process called knowledge discovery. The following summarizes the steps of the knowledge discovery process.

- Define the Problem. This initial step involves understanding the problem and figuring out what the goals and expectations are of the project.
- Collect, clean, and prepare the data. This requires figuring out what data are needed, which data are most important and integrating the information. This step requires considerable effort, as much as 70% of the total data mining effort.
- Data mining. This model-building step involves selecting data mining tools, transforming the data if the tool requires it, generating samples for training and testing the model, and finally using the tools to build and select a model.
- Validate the models. Test the model to ensure that it is producing accurate and adequate results.
- Monitor the model. Monitoring a model is necessary as with passing time, it will be necessary to revalidate the model to ensure that it is still meeting requirements. A model that works today may not work tomorrow and it is therefore necessary to monitor the behavior of the model to ensure it is meeting performance standards.

V. THE DATA MINING PROCESS

The goal of identifying and utilizing information hidden in data has three requirements :

- The captured data must be integrated into organization-wide views instead of specific views.
- The information contained in the integrated data must be extracted.
- The obtained information must be organized in ways that enable decision-making.

The data mining process can be classified as going through a series of four steps. This consists of transforming the already summarized data found in a data warehouse into information than can produce useful results. These four steps can be summarized into:

- Data selection
- Data transformation
- Mining the Data
- Interpretation of Results

Data selection consists of gathering the data for analysis. Data transformation will then convert appropriate data to a particular format. Data mining will then extract the desired type of information yielding in results to be interpreted. In Figure 1, the data-mining tool will extract the relevant information from the data warehouse environment. In order for the data-mining tool to work, the sub-processes of data selection and transformation must take place prior to data mining. The results are then passed to a decision-oriented databases or data mart, where the user can make a recommendation based on the results and put the recommendations into action. Of course this assumes that all of the four steps will be successfully completed, which is not always the case.

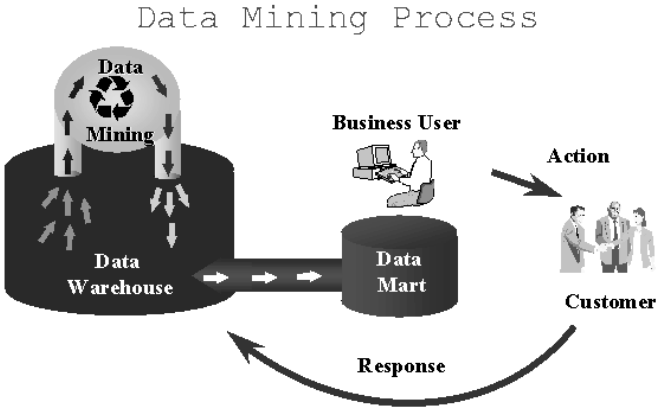


Figure 1: Data Mining Process

Data selection can be the most important step in the process. This is due to the complexity in finding and constructing pre-selection criteria before the extraction of data actually transpires. The variables selected and the range of each of them should be determined in this step. For example, a marketing executive wishing to improve sales figures will pre-select those customers that have been most active in making purchases and observe their behavior. An executive can mine all the data, but this can turn out to be a very costly operation because the data-mining tool will have to search through all this data and moreover if results are generated, they have more risk in predicting an optimal recommendation. Carefully choosing the data is therefore a very important step.

Once the data to be mined has been chosen the next step in the data mining process usually consists of transforming the data into the particular formats needed by the data-mining tool. Data are further synthesized by computing certain rations and applying algorithms to convert the data to a particular type suitable for future applied tools.

Once the data have been selected and required transformations done, a data-mining tool can now be applied. Specific predictions about futuristic events based on previous collected data can yield in significant hidden findings through the use of well-designed algorithms, a topic of discussion in later sections. Using a data warehouse alongside with a mining tool is usually recommended as this allows for a more efficient organization of the collected data in ways that can facilitate and optimize analysis. Furthermore, the mining tool can also interface with a DSS for further interpretation of the data.

However, a data mining system need not always interact with a data warehouse, and in fact, data mining can still extract pertinent information if given raw data from a database. The main advantage of using a data warehouse is that most of the data are already integrated in a suitable format of choice making it easier for a data-mining tool to extract the higher quality information.

The final step in the data mining process consists of interpreting the results. Once the extracted information is analyzed and interpreted, the most relevant information can be passed onto the decision-maker through a DSS. Result interpretation can consist not only of interpreting the output but also of further filtering the data and passing that information to the decision support system. In the case that the interpreted results are not satisfactory, it may be necessary to repeat any of the previous steps until the information generated contains the maximum added value to the data miner.

As such, data mining is a very complex process. Many steps need to be performed correctly before feeding of data to the data mining tool. Furthermore it is not guaranteed that the data-mining tool will yield significant results in any steps of the mining process. Certainly, performing many trials are recommended as this can reveal error corrections in any of the four steps. Any of the previously mentioned steps can be modified to continue investigating the data and searching for hidden patterns. This is the challenge of the data mining organization and though it can be a painstaking process, the more data that is mined, the more likely the data miner will learn from the process.

The use of tools such as DSS and a warehouse environment complement the data mining tools used to find useful facts buried in layers of data. To maximize the efficiency of data mining, both of these other tools must provide high quality delivery information to the data-mining tool. The use of good complementary tools to sift through data along with a powerful data-mining tool should be part of a well-designed environment.

VI. THE ALGORITHM UNDER EVALUATION

STAGES IN A CLUSTERING TASK

Figure below represents the typical sequencing of clustering activity. A *pattern* (or *feature vector*) is a single data item that is used by clustering algorithm. It typically consists of a vector of *d* measurements (where *d* the dimensionality of the data):

$$\mathbf{x} = (x_1, \ldots x_d).$$

Equation 1

The individual scalar components x_i *of* a patter X is are called features (or attributes). Pattern representation refers to the number of classes, the number of available patterns, the number, type and scale of the features available to clustering algorithms.

Feature selection is the process of identifying the most effective subset of original features to use in clustering. Feature extraction is the use of one or more transformations of the input features to produce new salient features. Either or both of these techniques can be used to obtain what is called a *feature set* (or feature vector).

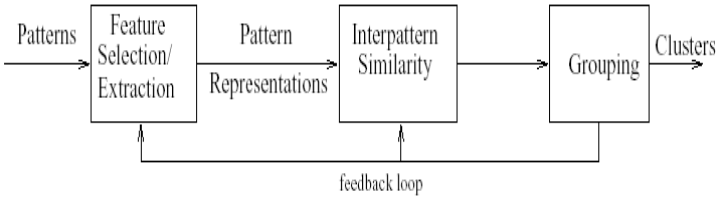


Figure 5: Stages in Clustering

Pattern proximity is usually measured by a distance function defined on pairs of patterns. A variety of distance functions are in use in various communities. A simple distance measure can often be used to reflect dissimilarity between two patterns, where other similarity measures can be used to characterize the conceptual similarity between two patterns. The Euclidian distance metric can be defined as follows:

$$d_2(\mathbf{x}_i, \mathbf{x}_j) = (\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2)^{1/2}$$

$$= \|\mathbf{x}_i - \mathbf{x}_j\|_2,$$

Equation 2

Where X_1 and X_2 are two patterns. Euclidian distance metric works well when the data set has "compact" or "isolated" clusters.
Another class of metrics characterizes conceptual similarity between two patterns. For example in Conceptual cluster (which we don't discuss in this paper), the similarity between X_1, X_2 is defined as

$$s(\mathbf{x}_i, \mathbf{x}_j) = f(\mathbf{x}_i, \mathbf{x}_j, \mathscr{C}, \mathscr{E}),$$

Equation 3

Where $\mathscr{C}$ is a set of pre-defined concepts.

The Grouping step represents the organization of patterns into clusters based on pattern similarity. There are many clustering methods available, and each of them may give a different grouping of a dataset. The choice of a particular method will depend on the type of output desired, the known performance of method with particular types of data, the hardware and software facilities available and the size of the dataset. The taxonomy of clustering algorithms can be seen in the *figure 6.*

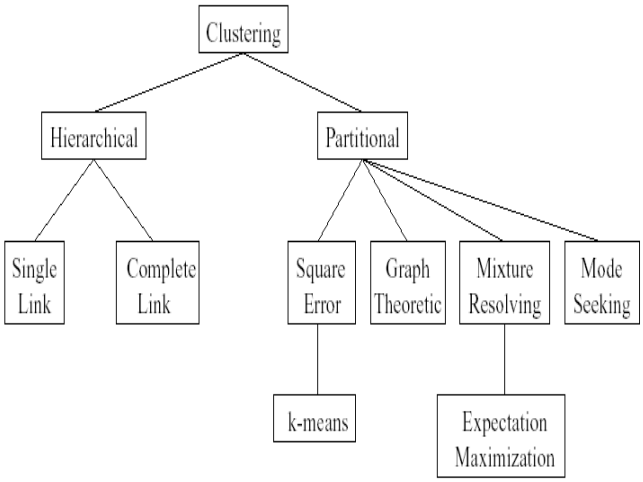


Figure 6: Taxonomy of clustering approaches

In general, clustering methods may be divided into two categories based on the cluster structure, which they produce. The non-hierarchical methods divide a dataset of N objects into M clusters, with or without overlap.
These methods are sometimes divided into *partitioning* methods, in which the classes are mutually exclusive, and the less common *clumping* method, in which overlap is allowed. Each object is a member of the cluster with which it is most similar; however the threshold of similarity has to be defined. The hierarchical methods produce a set of nested clusters in which each pair of objects or clusters is progressively nested in a larger cluster until only one cluster remains. The hierarchical methods can be further divided into *agglomerative* or *divisive* methods. In *agglomerative* methods, the hierarchy is build up in a series of N-1 agglomerations, or Fusion, of pairs of objects, beginning with the un-clustered dataset. The less common *divisive* methods begin with all objects in a single cluster and at each of N-1 steps divide some clusters into two smaller clusters, until each object resides in its own cluster.

VII. K-MEANS ALGORITHM

K-Means algorithm is very popular for data clustering. The Algorithm goes like this

1. Select k Center in the problem space (it can be random).
2. Partition the data into k clusters by grouping points that are closest to those k centers.
3. Use the mean of these k clusters to find new centers.
4. Repeat steps 2 and 3 until centers do not change.

This algorithm normally converges in short iterations.

Naive k-means algorithm

One of the most popular heuristics for solving the k-means problem is based on a simple iterative scheme for finding a locally optimal solution. This algorithm is often called the k-means algorithm. There are a number of variants to this algorithm, so to clarify which version we are using, we will refer to it as the naïve k-means algorithm as it is much simpler compared to the other algorithms described here. This algorithm is also referred to as the Lloyd's algorithm in.

The naive k-means algorithm partitions the dataset into 'k' subsets such that all records, from now on referred to as points, in a given subset "belong" to the same center. Also the points in a given subset are closer to that center than to any other center. The partitioning of the space can be compared to that of Voronoi partitioning except that in Voronoi partitioning one partitions the *space* based on distance and here we partition the *points* based on distance.

The algorithm keeps track of the centroids of the subsets, and proceeds in simple iterations. The initial partitioning is randomly generated, that is, we randomly initialize the centroids to some points in the region of the space. In each iteration step, a new set of centroids is generated using the existing set of centroids following two very simple steps. Let us denote the set of centroids after the i^{th} iteration by $C^{(i)}$. The following operations are performed in the steps:

(i) Partition the points based on the centroids C (i), that is, find the centroids to which each of the points in the dataset belongs. The points are partitioned based on the Euclidean distance from the centroids.

(ii) Set a new centroid $c(i+1) \in C(i+1)$ to be the mean of all the points that are closest to $c(i) \in C(i)$ The new location of the centroid in a particular partition is referred to as the new location of the old centroid.

The algorithm is said to have converged when recomputing the partitions does not result in a change in the partitioning. In the terminology that we are using, the algorithm has converged completely when $C^{(i)}$ and $C^{(i-1)}$ are identical. For configurations where no point is equidistant to more than one center, the above convergence condition can always be reached. This convergence property along with its simplicity adds to the attractiveness of the k-means algorithm.

The naive k-means needs to perform a large number of "nearest-neighbor" queries for the points in the dataset. If the data is'd' dimensional and there are 'N' points in the dataset, the cost of a single iteration is O (kdN). As one would have to run several iterations, it is generally not feasible to run the naïve k-means algorithm for large number of points. Sometimes the convergence of the centroids (i.e. $C^{(i)}$ and $C^{(i+1)}$ being identical) takes several iterations. Also in the last several iterations, the centroids move very little. As running the expensive iterations so many more times might not be efficient, we need a measure of convergence of the centroids so that we stop the iterations when the convergence criteria are met. Distortion is the most widely accepted measure.

Clustering error measures the same criterion and is sometimes used instead of distortion. In fact k-means algorithm is designed to optimize distortion. Placing the cluster center at the mean of all the points minimizes the distortion for the points in the cluster. Also when another cluster center is closer

to a point than its current cluster center, moving the cluster from its current cluster to the other can reduce the distortion further. The above two steps are precisely the steps done by the k-means cluster. Thus k-means reduces distortion in every step locally. The k-Means algorithm terminates at a solution that is locally optimal for the distortion function. Hence, a natural choice as a convergence criterion is distortion. Among other measures of convergence used by other researchers, we can measure the sum of Euclidean distance of the new centroids from the old centroids as in. In this thesis we always use clustering error/distortion as the convergence criterion for all variants of k-means algorithm

.

Definition 1: *Clustering error* is the sum of the squared Euclidean distances from points to the centers of the partitions to which they belong.

Mathematically, given a clustering ϕ , we denote by $\phi(x)$ the centroid this clustering associate with an arbitrary point *x* (so for k-means, $\phi(x)$ is simply the center closest to *x*). We then define a measure of quality for ϕ :

$$distortion_{\phi} = \frac{1}{N}\sum_{x}|x-\phi(x)|^2$$

Equation 4

Where |a| is used to denote the norm of a vector 'a'. The lesser the difference in distortion over successive iterations, the more the centroids have converged. Distortion is therefore used as a measure of goodness of the partitioning.

In spite of its simplicity, k-means often converges to local optima. The quality of the solution obtained depends heavily on the initial set of centroids, which is the only non-deterministic step in the algorithm. Note that although the starting centers can be selected arbitrarily, k-means is fully deterministic, given the starting centers. A bad choice of initial centers can have a great impact on both performance and distortion. Also a good choice of initial centroids would reduce the number of iterations that are required for the solution to converge.

Many algorithms have tried to improve the quality of the k-means solution by suggesting different ways of sampling the initial centers, but none has been able to avoid the problem of the solution converging to a local optimum. For example, [9] gives a discussion on how to choose the initial centers, other techniques using stochastic global optimizations methods (e.g. simulated annealing, genetic algorithms), have also been developed. None of these algorithms is widely accepted.

Kd-trees

A very important data structure that is used in our algorithm is a kd-tree. A kd-tree is a data structure for storing a set of finite points from a d-dimensional space. It was introduced and examined it in detail in [12, 13].

Kd-trees are simple data structures with the following properties:

1. They are binary trees;
2. The root node contains all the points;
3. A node is split along a split-plane such that points to the left are part of the left sub-tree, points to the right are part of the right sub-tree;
4. The left and right sub-trees are recursively split until there is only one point in the leaf or a certain condition is satisfied.

This is the basic kd-tree structure. There exist several variants of the kd-tree based on the way in which they choose the splitting plane, the termination criteria, etc.

Originally designed to decrease the time in nearest neighbor queries, kd-trees have found other applications as well. Omohumdro has recommended it in a survey of possible techniques to increase speed of neural network learning in [14]. In the context of k-means, kd-trees are used as a data structure to save the points in the dataset by methods described in [10], [4] and [2, 3]. A variant of the kd-tree is also used in [1].

Though kd-trees give substantial advantage for lower dimensions, the performance of kd-trees decreases/drops in higher dimensions. Other data structures like AD trees [5] have been suggested for higher dimensions [2] but these have never been used for k-means.

After this brief introduction to the kd-trees (which is the primary data structure used in our algorithm), we discuss the two main approaches that try to counter the shortcomings of the k-means algorithm.

The Greedy K-means Algorithm

In [1] is presented a variant of the k-means algorithm that is called the global k-means algorithm. The local convergence properties of k-means have been improved in this algorithm. Also it does not require the initial set of centroids to be decided. The idea is that the global minima can be reached through a series of local searches based on the global clustering with one cluster less.

Assumption: The assumption used in the algorithm is that the global optima can be reached by running k-means with the (k-1) clusters being placed at the optimal positions for the (k-1) clustering problem and the k^{th} cluster being placed at an appropriate position that is yet to be discovered.

Let us assume that the problem is to find K clusters and $K' \leq K$. We Use the above assumption, the global optima for $k = K'$ clusters is computed as a series of local searches. Assuming that we have solved the k-means clustering problem for $K' - 1$ clusters, we have to place a new cluster at an appropriate location. To discover the appropriate insertion location, which is not known, we run k-means algorithm until convergence with each of the points in the entire set of the points in the dataset being added as the candidate new cluster, one at a time, to the $K' - 1$ clusters. The converged K clusters that have the minimum distortion after the convergence of k-means in the above local searches are the clusters of the global k-means.

We know that for $k = 1$, the optimal clustering solution is the mean of all the points in the dataset. Using the above method we can compute the optimal positions for the $k = 2, 3, 4, \ldots K$, clusters. Thus the process involves computing the optimal k-means centers for each of the $K = 1, 2, 3\ldots K$ clusters. The algorithm is entirely deterministic.

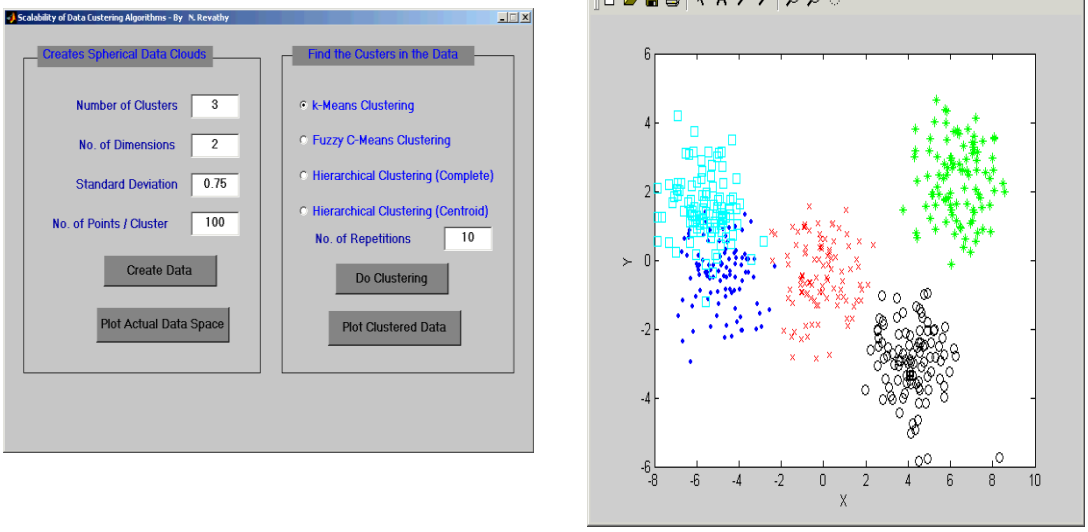
Though the attractiveness of the global k-means lies in it finding the global solution, the method involves a heavy cost. K-means is run N times, where N is the number of points in the dataset, for every cluster to be inserted. The complexity can be reduced considerably by not running the K-means with the new cluster being inserted at each of the dataset points but by finding another set of points that could act as an appropriate set for insertion location of the new cluster. In [1] is suggested the use of centers of regions, formed by a variant of the kd-tree, as the insertion points for new centroids.

The variant of the kd-tree splits the points in a node using the plane that passes through the mean of the points in the node and is perpendicular to the principal component of the points in the node. A node is not split if it has less than a pre-specified number of points or an upper bound to the number of leaf nodes is reached. The idea is that even if the kd-tree were not used for nearest neighbor queries, merely the construction of the kd-tree based on this strategy would give a very good preliminary clustering of the data. We can thus use the kd-tree nodescenters as the candidate/initial insertion positions for the new clusters.

The time complexity of the algorithm can also be improved by taking a *greedy approach*. In this approach, running k-means for each possible insertion position is avoided. Instead reduction in the distortion when the new cluster is added is taken into account without actually running k-means. The point that gives the maximum decrease in the distortion when added as a cluster center is taken to be the new insertion position. K-means is run until convergence on the new list of clusters with this added point as the new cluster. The assumption is that the point that gives the maximum decrease in distortion is also the point for which the converged clusters would have the least distortion. This results in a substantial improvement in the running time of the algorithm, as it is unnecessary to run k-means for all the possible insertion positions. However, the solution may not be globally optimal but an approximate global solution.

VIII. THE MAIN INTERFACE DESIGNED FOR ANALYSIS

The following GUI Interface was designed to modify some input parameters during evaluating the algorithms. The Controls in the left side of the form were used to control the artificial data creation. The right side controls were used to select an algorithm for evaluation.



The Main Interface

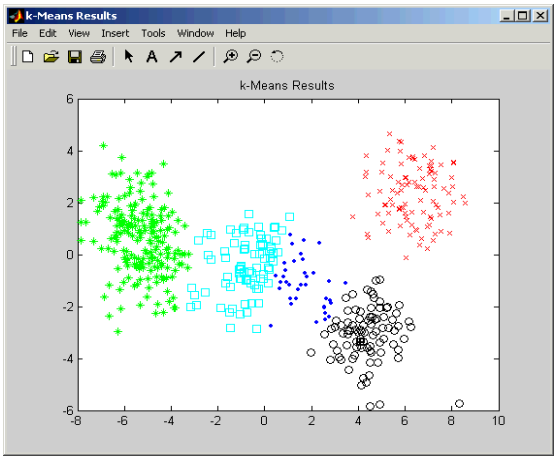
A SAMPLE RESULT

The 2 D Plot of Artificial Data

The following graph shows a 2 dimensional plot of original artificial data clusters belongs to five classes.

K-means clustering

K-Means Clustering Results:
Total Number of Clusters : 5 No's
Number of Dimension of Data: 2
Standard Deviation (width): 1
Total Number Points/Cluster: 100 Points/Cluster
Total Number Points : 500 Points
Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.8368
Self Rand index of Calculated Classes : 0.8860
Total Number of Repetitions : 10 sec
The Time Taken for Clustering: 0.160000 sec



The above graph shows a 2 dimensional plot of data as per the classification by k-Means Clustering algorithm.

K-Means Results:

Total Number of Clusters : 3 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 100 Points/Cluster
Total Number Points : 300 Points

Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 1.0000
Self Rand index of Calculated Classes : 1.0000

Total Number of Repetitions : 10 sec

The Time Taken for Clustering: 0.161000 sec

K-Means Results:

Total Number of Clusters : 4 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 100 Points/Cluster
Total Number Points : 400 Points

Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.7909
Self Rand index of Calculated Classes : 0.6871

Total Number of Repetitions : 10 sec

The Time Taken for Clustering: 0.060000 sec

K-Means Results:

Total Number of Clusters : 5 No's
Number of Dimension of Data: 2

Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 100 Points/Cluster
Total Number Points : 500 Points

Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.8543
Self Rand index of Calculated Classes : 0.7902

Total Number of Repetitions : 10 sec
The Time Taken for Clustering: 0.100000 sec

K-Means Results:

Total Number of Clusters : 3 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 200 Points/Cluster
Total Number Points : 600 Points

Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.9444
Self Rand index of Calculated Classes : 0.8887

Total Number of Repetitions : 10 sec

The Time Taken for Clustering: 0.050000 sec

K-Means Results:

Total Number of Clusters : 4 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 200 Points/Cluster
Total Number Points : 800 Points

Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.8467
Self Rand index of Calculated Classes : 0.8264

Total Number of Repetitions : 10 sec

The Time Taken for Clustering: 0.171000 sec

K-Means Results:

Total Number of Clusters : 5 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 200 Points/Cluster
Total Number Points : 1000 Points

Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.7376

Self Rand index of Calculated Classes : 0.7289

Total Number of Repetitions : 10 sec
The Time Taken for Clustering: 0.201000 sec

K-Means Results:

Total Number of Clusters : 3 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 300 Points/Cluster
Total Number Points : 900 Points

Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.9448
Self Rand index of Calculated Classes : 0.8895

Total Number of Repetitions : 10 sec

The Time Taken for Clustering: 0.080000 sec

K-Means Results:

Total Number of Clusters : 4 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 300 Points/Cluster
Total Number Points : 1200 Points

Hubert &Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.8031
Self Rand index of Calculated Classes : 0.8036

Total Number of Repetitions : 10 sec

The Time Taken for Clustering: 0.170000 sec

K-Means Results:

Total Number of Clusters : 5 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 300 Points/Cluster
Total Number Points : 1500 Points

Hubert &Arabia adjusted Rand index
Rand index of Calculated and True Classes: 0.7139
Self Rand index of Calculated Classes : 0.7103

Total Number of Repetitions : 10 sec

The Time Taken for Clustering: 0.391000 sec

K-Means Results:

Total Number of Clusters : 3 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 400 Points/Cluster
Total Number Points : 1200 Points

Hubert &Arabia adjusted Rand index
Rand index of Calculated and True Classes: 0.9900
Self Rand index of Calculated Classes : 1.0000

Total Number of Repetitions : 10 sec
The Time Taken for Clustering: 0.160000 sec

K-Means Results:

Total Number of Clusters : 4 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 400 Points/Cluster
Total Number Points : 1600 Points

Hubert &Arabia adjusted Rand index
Rand index of Calculated and True Classes: 0.8764
Self Rand index of Calculated Classes : 0.9965

Total Number of Repetitions : 10 sec

The Time Taken for Clustering: 0.230000 sec

K-Means Results:
Total Number of Clusters : 5 No's
Number of Dimension of Data: 2
Standard Deviation (width): 7.500000e-001
Total Number Points/Cluster: 400 Points/Cluster
Total Number Points : 2000 Points

Hubert & Arabie adjusted Rand index
Rand index of Calculated and True Classes: 0.8300
Self Rand index of Calculated Classes : 0.7509
Total Number of Repetitions : 10 sec
The Time Taken for Clustering: 0.421000 sec

IX. THE OVERALL RESULTS

Dimension of data	:	2
Total Clusters	:	5
Standard Deviation	:	0.75
Number of Repetitions	:	10

Results

Algorithm	Total Number Points/ Cluster	Total Number Points	Rand index	Self Rand index	The Time Taken for Clustering (in Seconds)	Avg. Classification Time per point (in Sec)
k-Means	500	2500	0.795	0.769	0.691	0.00038
	1000	5000	0.858	0.789	1.101	
	1500	7500	0.825	0.862	1.973	
	2000	10000	0.838	0.833	3.616	
	2500	12500	0.656	0.726	5.038	
	3000	15000	0.888	0.891	7.531	

Average classification time for a single data point for the algorithm were as follows:

K-Means : 0.000380
Average Performance per Point : 0.000440

X. CONCLUSION AND SCOPE FOR FURTHER ENHANCEMENT

An elaborate exploration was made on classical and modern data mining algorithms. K-Means algorithm was reviewed and implemented on Mat lab for studying the scalability performance of this algorithm.

For the final results, the algorithm was tested with number of points between 500 to 3000 per class. For all iterations, the number of classes was five and the number of dimension was two. While testing the algorithm between 500 to3000 points per class, the test was done repeatedly. In this case, the results were acceptable below 2500 points per class. But, greater than 2500 points per class, the performance was not seemed to be linear. For greater 3000 points per class, the algorithm took very long time. This proves the inability of that algorithm for handling bulk amount of data.

The performance of K-Means algorithm was same in the terms of speed. From the implementation results it was obvious that the algorithm which were implemented and tested were not scalable. The performance of the algorithm was found to be accurate if the total number of data, the total number of classes in the data and the dimension of each of the points in the data were comparatively very low.

REFERENCES

[1] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A. "Fast Discovery of Association Rules".
[2]Blockeel, H., De Raedt, L. "Relational Knowledge Discovery in Databases", Proceedings BENELEARN-96.
[3]Chattratichat, J. et al, "Large Scale Data Mining: Challenges and Responses", Proceedings KDD '2010.
[4]Gabowski, H., Lossack and Weibkopf, "Automatic Classification and Creation of Classification Systems Using Methodologies of Knowledge Discovery in Databases,"
[5]Galal, G., Cook, D.J., Holder, L.B. "Improving Scalability in a Scientific Discovery System by exploiting Parallelism", Proceedings KDD '2011.
[6]Holsheimer, M., Kersten, M., Mannila, H., Toivonen, H. "A Perspective on Databases and Data Mining", Proceedings KDD '95.
[7]John, G.H., Lent, B. "Sipping from the Data Firehouse", Proceedings KDD '2010.
[8]Porter, A. L., Kongthon, A., and Lu, J. C.) "Research Profiling – Improving the Literature Review: Illustrated for the Case of Data Mining of Large Datasets,"
[9]Toivonen, H. "Discovery of frequent patterns in large data collections", PhD Thesis, 2006.
[10]Teófilo Campos, "PCA for face recognition" Creativision research group, IME - USP - Brazil
[11]Srikant, R., Agrawal, R. "Mining Generalized Association Rules", Proceedings VLDB '2008.
[12] Toivonen, H. "Discovery of frequent patterns in large data collections", PhD Thesis, 2010.

ABOUT AUTHORS

Dr Revathy Nanjappanhad completed B.Sc., Computer Science in the year 2000 and Master of Computer Applications (MCA) in the year 2003 under Bharathiar University. Completed M.Phil. in Computer Science from Alagappa University in the year 2005. Completed Ph.D in Computer Science from Mother Teresa Women's University, Kodaikanal in the year 2013 and the area of research is Neural Networks. Other areas of interest are Mobile Computing, Data Mining and Artificial Intelligence. At present working as an Associate professor in the Department of Master of Computer Applications at Hindusthan College of Arts and Science at Coimbatore-641 028 and published 10 papers in International Journals, presented 8 papers in International Conferences and 56 papers in National Conferences.

Guhan Thangavelu had completed B.Sc., Computer Science in the year 1999 and Master of Computer Applications (MCA) in the year 2002 under Bharathiar University. Completed M.phil. In Computer Science from Alagappa University in the year 2006. Completed M.E. in Computer Science and Engineering from Anna University, Coimbatore in the year 2009 and the area of research is Data Mining. Currently pursuing Ph.D in Computer Science and Engineering from Anna University. Other areas of interest are Networking, Software Engineering and Artificial Intelligence. At present working as an Assistant professor (Senior Grade) in the Computer Science and Engineering at Sri Ramakrishna Engineering College, Coimbatore.Published 4 papers in International Journals, presented 8 papers in International Conferences and 32 papers in National Conferences.

Dr Selvarajan had completed B.E., Computer Science and Engineering, M.Tech from BITS Pilani and Ph.D in Computer Science and Engineering, from Anna University, Chennai. Areas of interest are Networking, Data Mining, Software Engineering, Cloud Computing, Image processing and Artificial Intelligence. At present working as Principal, Muthyammal Engineering College, Rasipuram, Namakkal District. Published 35 papers in International Journals, presented 30 papers in International Conferences and 89 papers in National Conferences.