MULTI-OBJECTIVE K-MEANS DATA CLUSTERING USING GENETIC ALGORITHM

By
Samer M. AlRamahi

Supervisor
Dr. Nasser H. Ruhhal

This Thesis was Submitted in Partial Fulfillment of the Requirements for The Master's Degree of Science in Industrial Engineering

Faculty of Graduate Studies
The University of Jordan

January, 2008

www.manaraa.com

This Thesis/Dissertation (Multi-objective k-means Data Clustering Using Genetic Algorithm) was successfully defended and approved on Dec. 27, 2007.

Examination Committee	**Signature**
Dr. Nasser Ruhhal, Chairman Assist. Prof. of Industrial Engineering	----------
Dr. Khaldoun Tahboub, Member Assist. Prof. of Industrial Engineering	----------
Dr. Sameh Al-Shihabi, Member Assist. Prof. of Industrial Engineering	----------
Dr. Nabeel Mandahawi, Member Assist. Prof. of Industrial Engineering Hashemite University of Jordan	----------

www.manaraa.com

DEDICATION

To my beloved family: My parents, brothers and sisters.

To my friends, I dedicated this piece of work as an accomplishment for all your endless caring, support and belief.

Lovingly yours,

Samer

www.manaraa.com

ACKNOWLEDGEMENT

To my instructor and supervisor Dr. Nasser H. Ruhhal, I hereby acknowledge your creative presence and thoughtful guidance that without, I would not have accomplished this challenge. Thank you indeed.

To all professors and supervisors that were of help and aid, I thank you all.

Your sincerely,

Eng. Samer Ramahi

www.manaraa.com

TABLE OF CONTENTS

www.manaraa.com

www.manaraa.com

LIST OF TABLES

www.manaraa.com

LIST OF FIGURES

www.manaraa.com

LIST OF ABBREVIATIONS

Abbreviation	Terminology
GA	Genetic Algorithm
k	Number of Clusters
d	Distance
X	Instance
C	Cluster center
MST	Minimal Spanning Tree
e	Error
O	Complexity
SSE	Sum of Square Error
SPEED1	Speed Setting that Controls the Speed of Rotation
TOTAL	Total Number of Bolts
SPEED2	Second Speed Setting
NUMBER2	Number of Bolts to be Counted at this Second Speed
SENS	Sensitivity of the Electronic Eye
TIME	Time to Count the Desired Number of Bolts
T20BOLT	Time to Count 20 Bolts

www.manaraa.com

MULTI-OBJECTIVE K-MEANS DATA CLUSTERING USING GENETIC ALGORITHM

By
Samer M. AlRamahi

Supervisor
Dr. Nasser H. Ruhhal

ABSTRACT

The importance of data clustering as one of the most effective data mining tools can be appreciated by realizing its applications in business and researches. In this thesis a multi-objective Pareto optimization enhanced by genetic algorithm used for the first time to overcome the problems that conventional *k*-means algorithms suffer which include ball shaped data clusters, dead initial points, pre determination the number of clusters and the dependency between finding optimal sum of square error and the selection of initial centers.

In order to overcome these problems the Mahalanobis distance that consider the input data covariance will be used to solve the problem of the ball shape output clusters so our model will work with ellipse shaped data as well as ball shaped. Initial centers for the *k*-means algorithm will be selected directly from the input data which may solve the problem of the dead points. The problem of determination of the number of clusters can be solved by finding all good possible solutions for the problem (Pareto set) which increases the probability of finding global minimum SSE.

A new algorithm is proposed to solve the problem and a program is coded based on the proposed algorithm. It is verified to ensure that it is working and validated using standard data. Case study using manufacturing problem was solved using the proposed methodology.

www.manaraa.com

Introduction

Data clustering is one of the most important data mining tools. The clustering problem has been addressed in many researches which reflect its usefulness as one of the data mining techniques which has many applications in business and researches include:

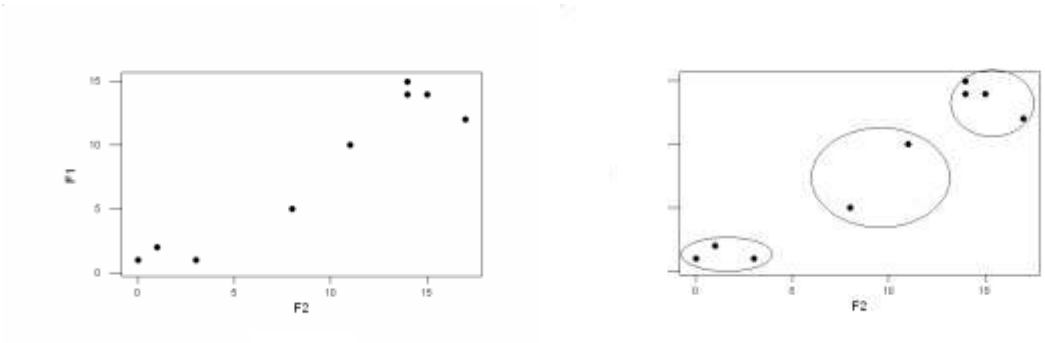
- Fault detection and quality improvement.
- Customer relationship management.
- Target marketing for businesses with small marketing budget.
- Segmentation of financial behavior into benign and suspicious categories.
- Data dimensions reduction tools.
- Gene expression clustering where large number of genes may have similar behavior.
- Image segmentation.
- Objects and characters recognition.
- Information search and retrieval.

Clustering is a very difficult problem that requires determination of similarity measures which is not easy specially that there is no prior knowledge about data. There is a large number of clustering techniques and everyone has its advantages and disadvantages. There is no single clustering algorithm that can handle all data types and shapes.

المنارة للاستشارات
www.manaraa.com

1.1. Definition

Data clustering according to Jain et al. (1999) is the unsupervised classification of patterns (observations, data items or feature vector) into groups (or clusters). An example of data clustering is shown in Fig (1.1). The input instances are shown in Fig (1.1-a) and clustered data is shown in Fig (1.1-b).



(a) (b)

Fig (1.1): Data clustering

1.2. Similarity measures

Similarity measures are a key point in data clustering algorithms. Defining the similarity measure in the data space is essential because it makes it possible to determine whether two instances are similar or dissimilar from each other. The most common measures are distance measures. According to Larose (2005), the most popular distance measures used for instances whose features are all continuous are *Minkowski distance, Euclidean distance, and City block distance*, where Jain et al (1999) added *Mahalanobis distance.*

www.manaraa.com

1.3. Data clustering techniques

In general clustering algorithms can be classified as: Hard and Fuzzy clustering.

There is a variety of hard clustering algorithm. The hierarchy shown in Fig (1.2) classifies these techniques into two major groups (hierarchical and partitioning algorithms). As shown in Fig (1.2) *k*-means is the most common partitioning techniques.

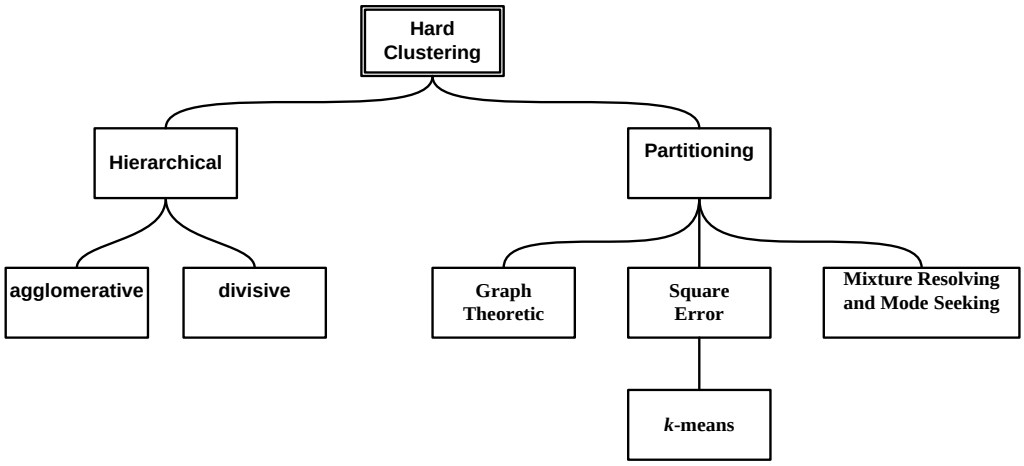


Fig (1.2): Hard clustering algorithms

1.4. An evolutionary algorithm for clustering

Evolutionary algorithms emulate the natural selection where the best can survive and reproduce the second generation. Any evolutionary algorithm starts with initial population then evolutionary operators (selection, combination, mutation) applied to the current generation to produce a new generation which is better than the previous one. A cycle of regenerating solutions starts until predetermined criterion is satisfied.

www.manaraa.com

The most common evolutionary algorithm is genetic algorithm which follows the same guidelines of evolutionary algorithms. Haupt et al (1998) studied the genetic algorithm (GA) and describes it as an algorithm starts with initial solution population (genes). Then a fitness function is calculated for each member of the population. To generate new population of solutions selection, crossover and mutation operators are applied to the current solution population. Detailed discretion for the algorithm is in chapter 2.

Genetic operators can be defined as follows:

Selection: is to select parents from the current solution population with a probability to select any solution proportional to its fitness.

Crossover: is to exchange segments of parents to generate children. Fig (1.3) shows a single point crossover.

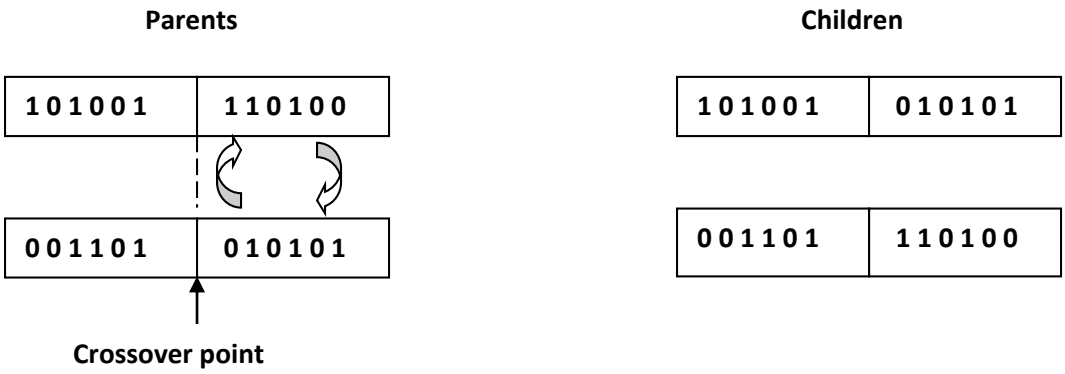


Fig (1.3): Crossover operator

www.manaraa.com

Mutation: is to mutate a segment or more as shown in Fig (1.4-a) to make sure that the algorithm explores the search space very well and not to stop in local optima as shown in Fig (1.4-b).

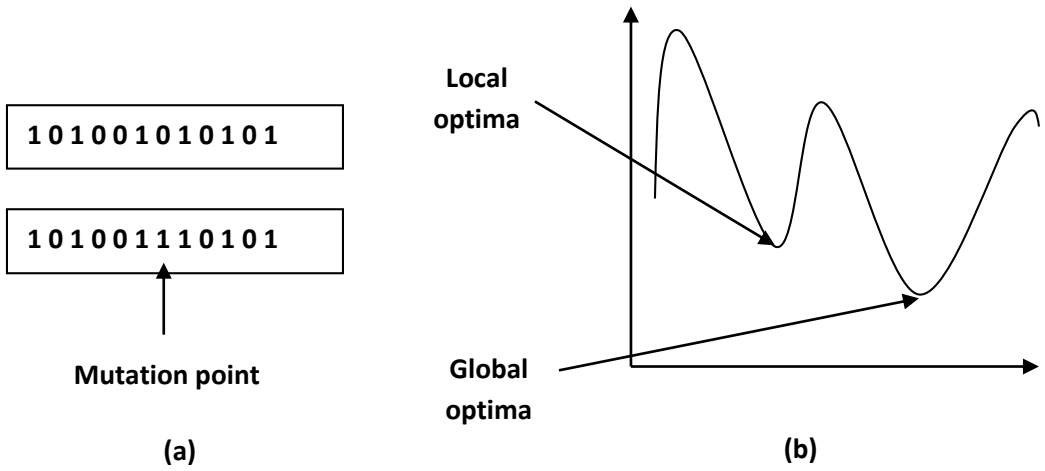


Fig (1.4): Mutation operator

.1.5 Multi-objective optimization for clustering

Most of the real world problems are multi-objective problem. Data clustering is one of these problems. Even though researchers for a long time solve the clustering problem as single objective problem, in clustering problem there are two objectives (Homogeneity within the cluster and Heterogeneity or separation between clusters).

To deal with multi-objective problems, one of the following criteria must be chosen:

- Use single weighted objective function.
- Use one objective and use the other as constraint.
- Use all objectives simultaneously (Pareto optimization).

www.manaraa.com

The first criterion is usually used with *k*-means algorithm when the Number of Clusters (*k*) is unknown. Second criterion is usually used with *k*-means algorithm when the Number of Clusters (*k*) already determined. In this thesis the third criterion will be used with *k*-means algorithm. Eskandari et al (2005) explain Pareto optimization as an optimization where all objectives are considered simultaneously there is no single solution for the problem but a set of solutions called *Pareto set* as shown in Fig (1.5).

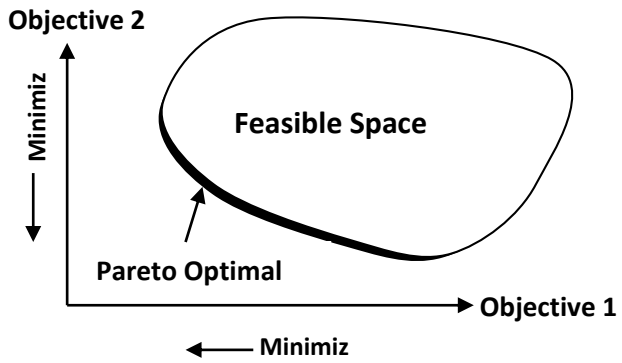


Fig (1.5): Pareto Optimal Set

1.6. Thesis problem

In this thesis a multi-objective Pareto optimization enhanced by genetic algorithm will be used to overcome the problems that conventional *k*-means algorithms suffer which include:

1- Data clusters are ball shaped.

2- Selection of the initial centers from the data space may face some dead points.

3- It needs to pre determine the number of clusters.

www.manaraa.com

4- *k*-means cannot guarantee finding the global minimum Sum of Square Error (SSE) which depend on initial clusters centers.

In order to overcome these problems the Mahalanobis distance that consider the input data covariance will be used to solve the problem of the ball shape output clusters so our model will work with ellipsoid shaped data as well as ball shaped. Initial centers for the *k*-means algorithm will be selected directly from the input data which may solve the problem of the dead points. The problem of determination of the number of clusters can be solved by finding all good possible solutions for the problem (Pareto set) which increases the probability of finding global minimum SSE.

This thesis contains six chapters. This chapter gives an introduction to clustering, genetic algorithm and multi-objective optimization. Chapter two reviews the literatures written in the same direction. Third chapter discusses the methodology used in this thesis including algorithms and computer program structure. Chapter four documents the verification and validation processes. Chapter six illustrates a case study. Last chapter reviews the conclusions and recommendations.

www.manaraa.com

2. Literature Review

In this chapter literatures studied data clustering, genetic algorithm and multi-objective optimization are reviewed. Clustering algorithms using genetic algorithm or multi-objective optimization are illustrated. Applications of data clustering in industry and services are reviewed.

2.1. Data clustering

Many researches addressed the clustering problem and they have a common definition of the problem. Jain et al. (1999) and Hwei et al. (2005) defined clustering as the unsupervised classification of patterns (or data items) into groups (or clusters) where resulting clusters should possess (1) homogeneity within clusters and (2) heterogeneity between clusters. While Larose (2005) reported that clustering refers to the grouping of records, observations, or cases into classes of similar objects and he defined the cluster as a collection of records that are similar to one another and dissimilar to records in other clusters. Larose distinguishes between clustering and classification he said that there is no target variable for clustering instead, clustering algorithms seek to segment the entire data set into relatively homogeneous subgroups or clusters, where the similarity of the records within the cluster is maximized, and the similarity to records outside this cluster is minimized. Ian et al. (2005) say that clustering is the process where the instances are to be divided into natural groups. Razib et al. (2006) define clustering as a process of dividing a set of objects into unknown groups, where the best number k of groups is determined by the clustering algorithm. That is, objects within each group should be highly similar to each other than to objects in any other group. It was reported that finding the k automatically is a hard

www.manaraa.com

algorithmic problem. Automatic clustering problem was defined as follows: Let $X = \{X_1, X_2, \ldots, X_n\}$ be a set of n objects. These objects are clustered into non-overlapping groups $C = \{C_1, C_2, \ldots, C_k\}$, where C is called a cluster, k is the unknown number of clusters, $C_i \cap C_j = \emptyset$ for $i \neq j$, $C_1 \cup C_2 \cup \ldots \cup C_k = X, C_i \in X, C_i \neq \emptyset$.

Defining similarity measures are a key point in data clustering problem. The most common measures are distance measures. According to Larose (2005) the most popular distance measures used for instances whose features are all continuous are:

- *Minkowski distance*: where distance between X_i and X_j can be defined in d dimensions and power P as follows:

$$d_p(X_i, X_j) = \left(\sum_{k=1}^{d} |X_{i,k} - X_{j,k}|^P\right)^{1/p}$$

Euclidean distance: which is a special case of Minkowski distance. It is usually used with two or three-dimensional data and it can be defined as follows:

$$d_2(X_i, X_j) = \left(\sum_{k=1}^{d} |X_{i,k} - X_{j,k}|^2\right)^{1/2}$$

- *City block distance* known as rectilinear or Manhaten distance which can be defined as follows:

$$d_2(X_i, X_j) = \sum_{k=1}^{d} |X_{i,k} - X_{j,k}|$$

www.manaraa.com

Jain et al. (1999) added the *Mahalanobis distance* measure which takes the covariance between dimensions variables in consideration and it can be defined as follows:

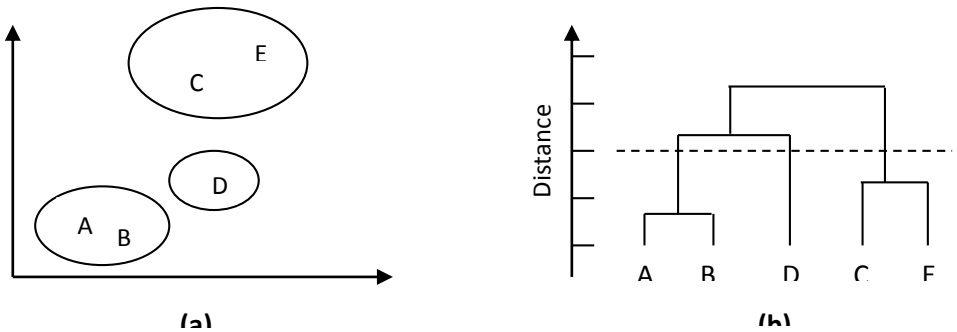
$$d(X_i, X_j) = (X_i - X_j)\Sigma^{-1}(X_i - X_j)^T$$

Where Σ is the covariance matrix of instances.

Clustering techniques were classified into:

- Hard clustering: where each instance is a member in one of the output clusters.
- Fuzzy clustering: where each instance has a variable degree of membership in each of the output clusters.

Hard clustering techniques were classified into hierarchical and partitional techniques where hierarchical clustering include single link and complete link and partitional clustering include square error techniques, graph theory, mixture resolving and mood seeking techniques. Hierarchal clustering was explained as a technique that builds a cluster hierarchy (tree) where every cluster node has child clusters. Two major hierarchal techniques were identified, *agglomerative* (bottom –up) which starts with one point clusters (singleton) then merges the most similar clusters and *divisive* (top-down) which starts with one cluster then split the most dissimilar clusters. Fig (2.1-b) shows the hierarchal clustering of instances shown in Fig (2.1-a).



(a) (b)

www.manaraa.com

Fig (2.1): Hierarchal clustering

Partitioning clustering algorithms were described as group of instances into clusters instead of building a hierarchy of instances. Partitioning clustering algorithms were categorized as following:

- *Graph-Theoretic algorithms*: the most common graph-theoretic divisive clustering algorithm builds the (Minimal Spanning Tree) MST and then removes the Largest Length Edges.

Fig (2.2) shows the same data set in Fig (2.1) clustered using graph-theoretic.

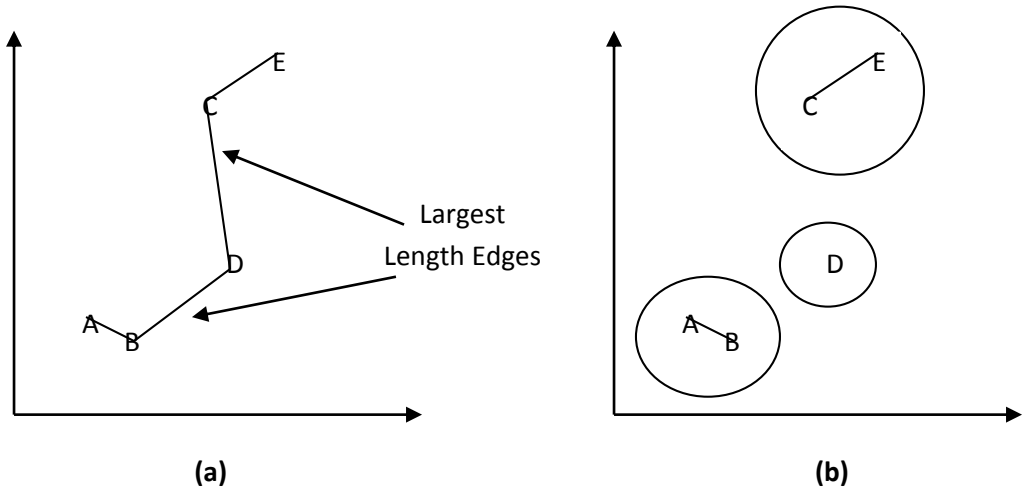


Fig (2.2): Graph-Theoretic clustering

- *Mixture-Resolving and Mode-Seeking algorithms*: these algorithms assume that the instances in each cluster are drawn from one of several distributions. The algorithms goal is to identify the parameters of each. Most of the work in these algorithms assumes that the

www.manaraa.com

mixture density is Gaussian and the Gaussian parameters are estimated using algorithms based on maximum likelihood.

- *Square Error algorithms*: minimum square error algorithms are the most frequently used clustering algorithms where *k*-means is the most common square error technique. These algorithms are based on minimizing the summation of square error (distance between each instance and its cluster center).

$$e^2 = \sum_{j=1}^{k} \sum_{i=1}^{nj} \left(X_i^j - C_j \right)^2$$

Where k is the number of clusters.

C_j is the center of cluster *j*.

nj is the number of instances in cluster *j*.

It was said that fuzzy clustering techniques is based on square error where algorithm starts with initial fuzzy partitioning by selecting the elements (Membership of instance *i* in cluster *j*) u_{ij} of the clustering matrix $U_{N\times K}$. Then matrix $U_{N\times K}$ can be recomputed to reduce the following weighted square error function:

$$E^2 = \sum_{i=1}^{N} \sum_{k=1}^{K} u_{ij} (Xi - Cj)^2$$

$$C_j = \sum_{i=1}^{N} u_{ij} X_i$$

Where C_j is the center of cluster *j*.

www.manaraa.com

N is the total number of instances

K is the total number of clusters.

2.1.1. *k*-means data clustering

k-means is a partitional algorithm where sum of squared error (sum of squared distances between instances in each clusters and the cluster center) is minimized and local optima is found. Larose (2005) reported that *k*-means algorithm is straightforward and effective algorithm for finding clusters in data. Jain et al. (1999) described the *k*-maens as a four steps algorithm as follows:

1- Choose *k* cluster centers to coincide with *k* randomly-chosen instances or *k* randomly defined points inside the hyper-volume containing the instance set.
2- Assign each instance to the closest cluster center.
3- Recompute cluster centers using the current cluster memberships.
4- If a convergence criterion is not met, go to step 2.

Where typical convergence criteria are: no (or minimal) reassignment of instances to new cluster centers, or minimal decrease in squared error.

Although *k*-means is very common, it has drawbacks. Cheung et al. (2003) confirm three drawbacks of *k*-means algorithm:

1- The data clusters are ball-shaped because it performs clustering based on the Euclidean distance only.

www.manaraa.com

2- There is the dead-unit problem. That is, if some units are initialized far away from the input data set in comparison with other units, they then immediately become dead.

3- It needs to pre-determine the cluster number.

Larose (2005) added that *k*-means cannot guarantee finding the global minimum SSE. To improve the probability of achieving a global minimum, the analyst should rerun the algorithm using a variety of initial cluster centers. Larose also assures that one potential problem for applying the *k*-means algorithm is: Who to decide *k* unless the analyst has a priori knowledge of the number of clusters, therefore, an "outer loop" should be added to the algorithm, which cycles through various promising values of *k*. Clustering solutions for each value of *k* can therefore be compared, with the value of *k* resulting in the smallest SSE being selected.

To overcome the above mentioned problems researchers improved traditional *k*-means algorithm. Cheung (2003) propose a new clustering technique named STep-wise Automatic Rival-penalised *k*-means algorithm (denoted as *k*-means hereafter), which is a generalization of the conventional *k*-means algorithm. The algorithm consists of two separate steps. The first one is a pre-processing procedure, which assigns each cluster at least a seed point. Then, the next step is to adjust the units adaptively by a learning rule that automatically penalizes the winning chance of all rival seed points in the subsequent competitions while tuning the winning one to adapt to an input. Cheung claimed that his algorithm can perform clustering problem without predetermine the correct number of clusters.

Other researchers try to reduce the complexity of the *k*-means algorithm. Fahim et al. (2006) propose adaptation of traditional *k*-means algorithm to reduce the complexity of the

www.manaraa.com

k-means algorithm. It was said that the *k*-means algorithm computes the distances between data point and all centers and they claimed that this is computationally very expensive. They try to benefit of previous iteration of *k*-means algorithm. The idea behind their algorithm is that in each iteration cluster centers moves as new points. This motion makes the center closer to some points and far apart from the other points, the points that become closer to the center will stay in that cluster, so there is no need to find its distances to other cluster centers. The points far apart from the center may change the cluster, so only for these points their distances to other cluster centers will be calculated, and assigned to the nearest center. It was claimed that they reduces the complexity of *k*-means algorithm from $O(nkl)$ (where n is the number of point, k is the number of clusters l is the number of iterations) to approximately $O(nk)$.

2.1.2. Application of data clustering in production and service

Data mining techniques have many applications in production and services. Harding et al. (2006) assure that data mining applications in manufacturing are varied in areas such as production processes, operations, fault detection, maintenance, decision support, product quality improvement, customer relationship management and others. Data clustering as one of the data mining techniques has its application in production and services area. According to Kusiak (2006) data clustering is used to group customer needs by some companies to support production strategy which is a short of the assemble-to-order strategy, aims at developing preassembled configurations at attractive prices. Liu et al. (2004) used *k*-means data clustering to group customer with similar lifetime value or loyalty, according to

www.manaraa.com

weighted variables (recency, frequency and monetary). West et al. (2005) studied the customer loyalty using clustering of supermarkets customer data including six variables which are:

1- Average weekly spending of a customer.

2- How many different product categories did the customer spend money in?

3- How many different sub categories did the customer purchase from?

4- How many products did the customer purchase per week?

5- How much money did the customer spend per week?

6- How often did the customer visit per week?

Customers' data from seven supermarket stores were used. The supermarkets are part of a national chain. The data was taken over a thirteen-week period starting in July 2000. It included information on spending, visits, categories shopped, and other transactional data. The clustering was done using the data mentioned above. Both *k*-means and other clustering algorithms were used. Customers were clustered into five groups as follows:

Group 1: Loyal big spenders.

Group 2: Infrequent customers.

Group 3: Semi-loyal potentially big spenders.

Group 4: Loyal moderate spenders.

Group 5: Potentially moderate to big spenders with limited loyalty.

Tian (2002) presents a new approach to software reliability modeling by grouping data into clusters of homogeneous failure intensities (failure frequency). He evaluated his model by testing of two large software systems from IBM.

www.manaraa.com

Ettler et al. (2000) review the application of data mining in complex industrial process such as cold rolling. A methodology to monitor operation moods depending on the operator was introduced. Two dimensional data for four shifts where x-axis is related to the hydraulic pressure in the roll bending system and y-axis corresponds to strip tension in the input side of the rolling mill were clustered. It was found out that there are two shifts succeeded to avoid problematic areas and the other two shifts fail and the worse operating shift was determined.

2.2. Genetic Algorithm

Genetic algorithm is an algorithm used to find the optimal solution by emulating the natural selection where the best can survive. Haupt et al. (1998) define genetic algorithm as an optimization and search technique based on the principles of genetics and natural selection. That basic genetic algorithm was described as follows:

1- Define cost function, cost, and variables.

2- Generate initial population.

3- Decode chromosomes.

4- Find cost for each chromosome.

5- Select mates.

6- Mating.

7- Mutation.

8- If convergence condition is not reached go to step (5).

GA is based on evolutionary operators which are selection, crossover and mutation

www.manaraa.com

appear in step (5), step (6) and step (7) respectively. Jain et al. (1999) describe the selection genetic operator as a process of propagation solutions from the current generation to the next generation based on their fitness. Selection employs a probabilistic scheme so that solutions with higher fitness have a higher probability of getting reproduced. Crossover operator was described as a process which takes as input a pair of chromosomes (called parents) and outputs a new pair of chromosomes (called children or offspring). A single point crossover was explained as an operation which exchanges the segments of the parents across a crossover point. Mutation operator was explained as a process that takes as input a chromosome and outputs a chromosome by complementing the bit value at a randomly selected location in the input chromosome. It was confirmed that both crossover and mutation are applied with some pre-specified probabilities which depend on the fitness values.

Eskandari et al. (2005) reported the advantages of GA over other optimization search techniques are the following:

1- GA-based approaches are capable of finding a number of optimal solutions rather than a single solution.
2- GA-based approaches are capable of exploring the search space more thoroughly with the smaller number performance(s) evaluations than those based on local search, such as simulated annealing.
3- GA-based approaches are less dependent on the good selection of the starting points.

2.2.1. Clustering using genetic algorithm

www.manaraa.com

Jain et al. (1999) reported a basic genetic algorithm for clustering problem as following:

1- Choose a random population of solutions where each solution corresponds to a valid *k*-partition of the data.
2- Associate a fitness value with each solution where typical fitness is inversely proportional to the squared error value.
3- Use the evolutionary operators selection, crossover and mutation to generate the next population of solutions.
4- Repeat step 2 until some termination condition is satisfied.

Hwei et al. (2005) say that non-GA based clustering algorithms cannot automatically form the natural group of the input data, this is due to the bad choice of the initial centers especially when the number of clusters is large so that difficult problems such as clustering problem often dealt by employing evolutionary techniques where GA is the best known evolutionary technique. Razib et al. (2006) propose a GA based algorithm to find one optimal solution for the clustering of gene ontology problem where the number of clusters *k* is unknown. There algorithm employs GA to cluster data using split-and-merge algorithm. Haiyan et al. (2003) propose a hybrid GA and simulated annealing to find the optimal or near optimal medoids. Two validation indexes were reviewed. One of the indexes is Silhouette width which reflecting the compactness and separation of the clusters. Silhouette width = $\frac{1}{N}\sum_{i=1}^{N}\frac{b(i)-a(i)}{\max\{a(i),b(i)\}}$ where *a(i)* is the average distance of point *i* to other points in the same

www.manaraa.com

cluster, *b(i)* is the average distance of point *i* to points in its nearest neighbor cluster. Du et al. (2005) used GA to find the optimal Pareto set of solutions for the clustering problem. Square error was minimized simultaneously with minimizing the number of clusters using genetic algorithm with linked-list structure.

2.3. Multi-objective optimization

Most of the literatures assure that most of the real world problems are multi-objective problems. Eskandari et al. (2005) added that improving one objective may eteriorate the performance in terms of one or more other objectives. It was said that traditional approaches for solving multi-objective optimization problems try to scalarize the multiple objectives into a single objective and change the problem formulation to a single objective optimization problem in which only a global optimal point is desired. It was mentioned that traditional way has major drawbacks and they determine them as follows:

1- The priority vector used in the scalarization process influences the final solutions.

2- Alternative solutions will not be available to decisions makers without at least changing some parameters such as priority vector.

3- Some optimal solutions may never be found if the objective function is not convex (real-life problems are seldom convex).

4- There are implications in the homogenization of different performance measures, such as cost, quality of products, and cycle times, to a common unit of measure.

www.manaraa.com

5- Traditional approaches may not work effectively if objectives are noisy or have discontinuous variable space.

To avoid the above mentioned drawbacks of traditional scalarization process Eskandari et al. (2005) propose the use of Pareto optimal set of solution to solve the simulation optimization problem.

An algorithm to find the optimal Pareto set of solution was proposed for their problem using GA which can be generalized for any problem as follows:

1- Create an initial population randomly or take user-specified individuals (or a combination of them).
2- Evaluate objective functions of the individuals and register all of them.
3- Calculate their fitness values according to the new dummy fitness assignment.
4- Determine nondominated individuals and register them.
5- Select the pairs of individuals as parents in the reproduction operation based on their assigned dummy fitness.
6- Perform crossover and mutation operations to generate the offspring to fill in the remaining positions in the new population.
7- If the stopping criterion is met, terminate the search; otherwise, return to step 3.
8- Use the screening algorithm to reduce the obtained large set of Pareto optimal solutions to a manageable size.

Haupt et al. (2004) review how to deal with multi-objective problem using the sum of waited objective functions and Pareto optimization. It was assured that using GA with multi-objective Pareto optimization needs a large population size to work well.

www.manaraa.com

2.3.1 Clustering using multi-objective optimization

Researchers dealt with clustering problem as a single objective problem in different ways. Fahim et al. (2006) try to enhance *k*-means algorithm which dealt with the clustering problem as a single objective optimization problem where (homogeneity) is optimize (heterogeneity) used as a constraint for the problem by predetermine number of clusters *k*.

Other researchers such as Haiyan et al. (2003) dealt with the clustering problem as a single objective optimization problem where a predefined index which scalarize the two objectives (homogeneity and heterogeneity) is optimize to fined the optimal solution without predetermine number of clusters. Hwei et al. (2005) worked in the same direction although it was assured that clustering problem is a multi-objective problem by nature where resulting partitions should comply with (1) homogeneity within each cluster and (2) heterogeneity between clusters.

Du et al. (2005) tried to find the optimal Pareto set of solutions for the clustering problem. Square error was minimized simultaneously with minimizing the number of clusters using genetic algorithm with linked-list structure for instances in the same cluster as follows:

Objectives: Minimize the intra distance (square error) - Minimizing the number of clusters.

Decision Variables: Assigned cluster of each data point.

2.4. Literature Discussion

Literature shows that clustering problem has been solved using different algorithms. One of these algorithms is *k*-means which is straightforward and effective algorithm for finding clusters in data, but *k*-means has some drawbacks including the dependency on initial

www.manaraa.com

clusters centers and ball shapes clusters. Genetic algorithm was employed to find the best initial centers for *k*-means. Most of the literatures deal with clustering problem which is multi-objective problem as single objective problem, some literatures scalarize clustering objectives to solve *k*-means problem when the number of clusters (*k*) is unknown. Others used the number of clusters (*k*) as a constraint to solve *k*-means algorithm when (*k*) already determined. Others solve the clustering problem using Pareto optimization but literatures never use *k*-means with Pareto optimization although *k*-means may help the algorithm to find local optimums. In this thesis Pareto optimization will be used to find a set of solution for *k*-means with different number of (*k*). This method will help *k*-means to be independent from the selection of initial centers by examining the space where all initial centers are possible. This method will give the decision maker alternatives rather than single solution.

In this thesis the square error will be minimized and inter distance between clusters will be maximized simultaneously using GA enhancing *k*-means as follows:

Objectives: Minimize the intra distance (square error) - Maximize inter distance.

Decision Variables: Assigned cluster each data point.

Constraint: $2 \leq$ Number of clusters in each solution $\leq$ Maximum number of clusters.

3. Methodology

This chapter discusses the methodology used in this thesis including proposed algorithm and program structure. In this thesis a new algorithm is proposed to solve clustering problem. The proposed algorithm is based on *k*-means algorithm which clusters data in each

www.manaraa.com

solution individual generated by the GA. The GA improves the solutions generations by changing initial clusters centers and number of clusters. Dominated solutions are removed in each phase from the solution generation to get the Pareto set in the final phase. The work map is as follows:

1- A new algorithm is proposed.

2- A program is coded as the proposed algorithm:

Java language is used to write the program which is a standalone program.

The program satisfies the following:

a- Program accepts data with different formats.

b- Program finds a set of solutions.

c- All solutions found by the program is non dominated solutions.

d- Program does not guarantee finding global optima but good solutions.

e- Program code is opened for further research.

3- The program was verified to ensure that the program is working.

4- The program was validated using standard data such as (iris data set, cpu data set and others).

5- A real life data (bolts data) was clustered using the program.

6- Performance measures such as time needed to solve the problem and Silhouette Width was calculated to evaluate the algorithm performance.

www.manaraa.com

3.1. Algorithm

In this thesis a new algorithm is proposed to solve clustering problem. Proposed algorithm is built by combining *k*-means algorithm with Pareto optimization using GA to satisfy the following requirements:

1- Algorithm should capable to use Euclidian distance and Mahalanobis distance to work with ball shaped data and ellipse shaped data as well.
2- GA must be used to avoid the dependency of traditional *k*-means on selection of initial centers.
3- GA and Pareto optimization must be used to avoid the need of traditional *k*-means to predetermine the number of clusters *k*.
4- GA must be used to increase the probability of finding the optimal set of solutions for the problem.

In order to satisfy the first requirement the algorithm is built to recommend a type of distance to use and accept user defined type of distance. To satisfy the other three requirements GA based Pareto optimization is subjugated to search a space of feasible solutions. Feasible space contain all possible number of clusters *k* and all possible combinations of initial cluster centers where these centers is selected directly from the data

www.manaraa.com

to avoid dead initial centers. Proposed algorithm is shown in Fig (3.1).

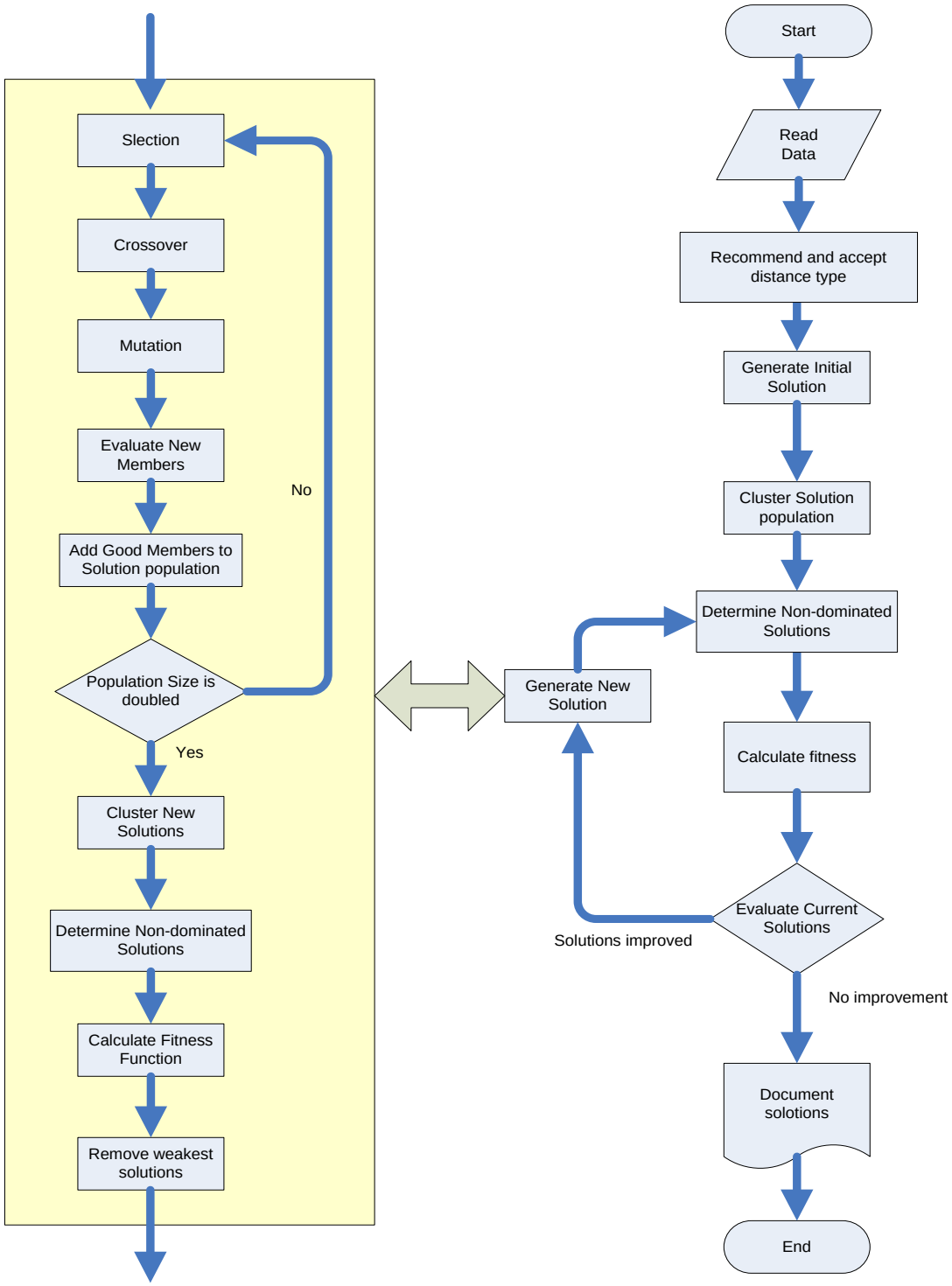


Fig (3.1): Proposed algorithm

www.manaraa.com

- *Recommend and accept distance type*: Proposed algorithm accepts Euclidian and Mahalanobis distance where Euclidian distance is recommended when the number of features or attributes is less than or equal to three as shown in following pseudocode.

```
IF ( Number_of_ attributes <= 3 ) THEN
        Recommend Euclidian_distance
ELSE
       Recommend Mahalanobis_distance
END IF
GET distance_type
```

- *Read data file*: Proposed algorithm reads data with different formats such as space separated, tap separated and comma separated files.

- *Generation of initial solution*:

1. Chromosome encoding: Chromosome encoding must satisfy the following requirements:

    1- Binary encoding is to be used to ease applying of crossover and mutation operator.

    2- Chromosomes representing different solutions with different number of clusters *k* can mate and crossover and mutation operators can be applied on them, number of clusters *k* in resulting chromosome is not depending on parents' chromosomes.

    3- Chromosome encoding must include determination of the initial cluster centers so *k*-means algorithm can be applied to cluster them.

www.manaraa.com

In order to satisfy the above mentioned requirements chromosome encoding used by Hwei et al.(2005) where data points in the data set are candidates for the cluster centers. The chromosome length is equal to the size of the data set. The *i*th gene of a chromosome corresponds to the *i*th data point in the data set. If a data point of index *i* is an initial cluster center the value of the corresponding *i*th gene in the chromosome is set to (1) otherwise (0). The number of clusters, denoted by *k*, is assumed to lie in the range [k_{min}, k_{max}], where k_{min} is chosen as 2, and k_{max} is chosen by the user or automatically $l/2$ or $\sqrt{l}$ will be used depending on the database size. Where l is the number of instances in the data set.

2. Solutions initialization: Random set of solutions is built by the algorithm with different number of cluster *k* for each solution. The solutions set size must be large enough to explore space with small number of generations of GA and it must be small to speed up the algorithm so set size is chosen to be equal to (2 X (data size + k_{max})).

- *Cluster Solution Set:* After initial set of solutions initialized every chromosome in the set is clustered using *k*-means algorithm which at least guarantees finding local *SSE* minima. The following algorithm shows the use of *k*-means to cluster data.

```
FOR  i = 1 to Number_of_solutions
    DETERMINE Number_of_clusters [i]
    Assign initial Center from Data_point array to Cluster_center array
    WHILE (counter < 100)
          IF ( Clustered_chromosome [counter] = Clustered_chromosome [counter -1] ) THEN
             BREAK WHILE
```


www.manaraa.com

```
            END IF
            FOR  j = 1 to Data_size
            Assign Data_point [j] to closest Cluster_center
             END FOR
             FOR  j = 1 to  Number_of_clusters [i]
                  Cluster_center[j] = mean of Data_point (s) assigned to cluster
             END FOR
    END WHILE
END FOR
```

In above *k*-means algorithm there are two stopping critera as following:

1- Time criterion: Maximum number of iterations is set to be (100) to avoid long time looping.

2- No farther improvement criterion: If there is no change in the cluster members the algorithm breaks the loop.

- *Determine dominated solutions*: For any solution if there is another solution in the solution set with less homogeneity and larger heterogeneity the solution become dominated and the solution is out of the Pareto set of solutions. Homogeneity and heterogeneity measures used in the algorithm are as follows:

www.manaraa.com

1. *Homogeneity* = $\frac{1}{N}\sum_{i=1}^{N} d(X_i, C_i)^2$

2. *Heterogeneity* = $\frac{1}{N}\sum_{i=1}^{N} d(X_i, C_{\min})^2$

Where X_i is *ith* data point.

C_i is the cluster center of the *ith* data point.

C_{min} is the cluster center of the closest cluster to the *ith* data point.

k is the number of clusters.

N is the number of data points.

Determination of the dominated solutions is shown in the following algorithm.

```
FOR  i = 1 to Number_of_solutions
      Calculate Homogeneity [i] and  Heterogeneity [i]
END FOR
FOR  i = 1 to Number_of_solutions
   FOR  j = i+1 to Number_of_solutions
        IF (Homogeneity [i] <= Homogeneity [j]  and  Heterogeneity [i] > = Heterogeneity [j]) THEN
         Solution [j] is Dominated
   END IF
   END FOR
END FOR
```


www.manaraa.com

- *Fitness calculations*: Dummy fitness proposed by Eskandari et al. (2005) is used in the algorithm to support Pareto optimization with some changes to suit clustering problem. All nondominated individuals of the population are assigned the initial dummy fitness value *Fi* of zero, which implies that there is no individual in the current population that is better than these identified individuals with respect to both homogeneity and heterogeneity. The remaining individuals dominated points which are compared to the nondominated points. Each dominated point is compared with the closest nondominated point and initial dummy fitness equal to the distance between them is assigned to the dominated solution. Final dummy fitness is calculated inversely proportional to initial fitness and repeated solution are given final fitness equal to zero. The sum of the final fitness is equal to one to ease the application of selection operator. Calculation of fitness value of each solution is shown in the following algorithm.

```
FOR  i = 1 to Number_of_solutions
     CASE: (nondominated solution)  F[i] = 0
          (doinated solution)  {F[i] = distance to closest nondominated solution
                                    Sum = sum + F[i]}
     END CASE
END FOR
FOR  i = 1 to Number_of_solutions
     CASE: (not repeated solutions )  {F[i] =sum – F[i]
                                         sum_2 = sum_2 + F[i]}
          (repeated solution)  F[i] = 0
     END CASE
END FOR
```


www.manaraa.com

- *Generation of new population*: To generate a new population GA operators are applied as follows:

1. Selection: Random solutions are selected for mating process with a probability to select any solution equal to its fitness since the sum of fitnesses of solutions is equal to one.

2. Crossover: A single point crossover scheme is used in the algorithm with probability to perform crossover for selected parents equal to (0.9).

3. Mutation: A single point mutation is used with probability to mutate each solution equal to (0.2). To keep the population size after population regeneration process nondominated solutions are determined and fitnesses are calculated for new generation. The fittest solutions are kept in the new solution generation.

- *Stopping criteria*: There are two stopping criteria as follows:

1- Time criterion: Maximum number of population regeneration is set to be (999) to avoid infinite looping.

2- No farther improvement criterion: Improvement in average homogeneity, average heterogeneity and number of nondominated solutions are calculated in each generation. If the improvement is less than 1% in average homogeneity and heterogeneity and less than 2% of the population size of the number of nondominated solutions measure for three sequential generations, algorithm stops.

www.manaraa.com

- *Solution reporting*: Only not repeated nondominated solutions are reported to the user. Solutions will be reported using several ways such as:

1- Solutions table.

2- Solutions statistics.

3- Pareto graph.

4- Two dimensions graph.

3.2. Program

A program was coded using proposed algorithm. Java language is used to write the program so the program can be used as a standalone program or it can be used as java applet that can be added to web page so other researchers can take the advantage to use the program. Program code is opened for future work.

3.2.1. Computer resources

- *Hardware resources*: The program is coded and operated using a computer with the following resources:

1- Processer is intel centrino 1.86 GHz.

2- RAM 1 GB.

- *Software resources*: Softwares used to code or operate the program are the following:

www.manaraa.com

1- Operating system: Microsoft Windows XP, version 2002.

2- Compiler: Sun Microsystems, Java SE Development Kit (JDK), version 6.

3- Java IDE: NetBeans IDE, version 5.5.

3.2.2. Program structure

The program was coded using Java which is object oriented language. The program contains six classes and three applets. Program structure is shown in Fig(3.2).

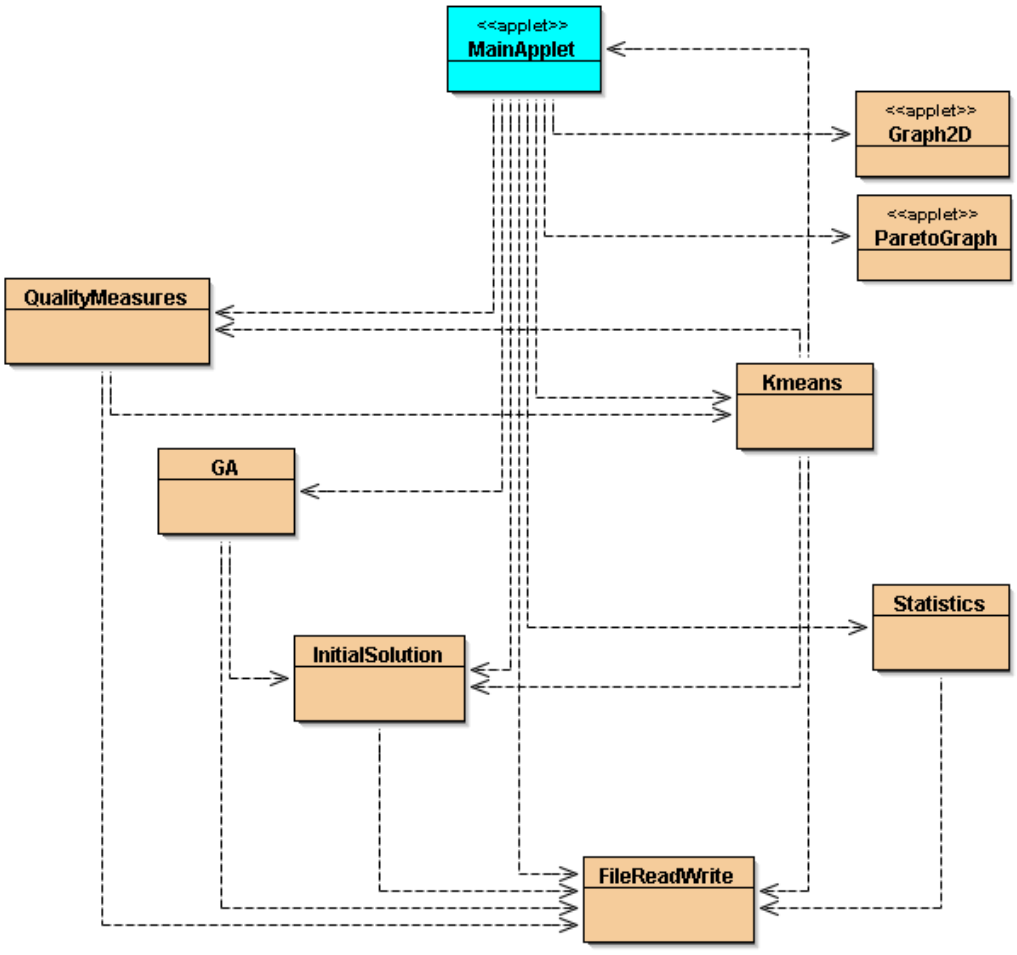


Fig (3.2): Program structure

www.manaraa.com

- *Program classes*:

1- FileReadWrite: This class contains functions that specialized in reading input files and writing to output files.

Main functions:

3- OpenFile (open input file).

4- ReadFile (read input file).

5- SaveToFile (save a string to output file).

2- InitialSolution: this class contains a function that specialized in formatting random initial set of solutions and encodes initial chromosomes.

Main functions:

1- initiate (format random initial set of solutions).

3- Kmeans: This class contains functions that specialized in calculating distance measures, function that perform *k*-means clustering and function that determine dominated solutions.

Main functions:

1- Distance (calculate distance measure).

2- Kmeans (cluster solution chromosome).

3- DetermineDominated (determine dominated solutions).

www.manaraa.com

4- GA: This class contains a function that calculates fitness and functions specialized in perform genetic algorithm operators selection, crossover and mutation.

Main functions:

1- InitiateSelectionMatrix (form matrix of selection).
2- Select (apply selection operator to a generation of solutions).
3- Crossover (apply crossover operator to parents chromosomes to generate children chromosomes).
4- Mutation (apply mutation operators to children chromosomes).
5- Fitness (calculate fitness for individuals in a generation of solutions).

5- QualityMeasures: This class contains functions that calculate objective functions homogeneity and heterogeneity and calculate silhouette width as a quality measure for clustering.

Main functions:

1- HomogeneityHeterogeneity (calculate homogeneity and heterogeneity for individuals in a generation of solutions).
2- Silhouette (calculate silhouette width for individuals in the final set of solutions).

www.manaraa.com

6- Statistics: This class contains functions that report statistics for the final set of solutions.

Main functions:

1- NumberOfMembers (count number of members in each cluster of a solution individual).

2- Range (calculate the maximum and minimum value for each attribute for each cluster of a solution individual).

- *Program applets*:

1- ParetoGraph: This applet is specialized in drawing Pareto Graph of the final set of solutions.

2- Graph2D: This applet is specialized in drawing of the final set of solutions individuals.

3- MainApplet: This applet communicates with the user throw advanced GUI and calls functions from other classes to control operations in the main algorithm.

3.3. Data

In this thesis four data sets were used (Simple data, Iris data, CPU data and Bolts data). Three data sets were used in the program verification and validation phase. The other set was used in the case study. Simple data set is hypothetical in (X,Y) dimensions specially to verify and validate the program during coding phase. Iris data is usually used by the literatures, it contains four attributes (sepal length, sepal width, petal length, petal width). Iris data contains 150 instances and it will used during validation phase to check the time cost and the generated solutions of the program. CPU data contains six attributes and 209

www.manaraa.com

instances and it will used during validation phase to check the time cost and the generated solutions of the program. Iris and CPU data is taken from data sets of WEKA project from university of Waikato. Bolts data is data from an experiment on the affects of machine adjustments on the time to count bolts. Bolts data has six attributes which include machine adjustments and total time to count 20 bolts. Bolts data is taken from StatLib website. Table (3.1) shows the details of the data set.

Table (3.1): Data sets

	Simple data	Iris data	CPU data	Bolts data
No. of attributes	2	4	7	6
No. of instances	5	150	208	40
Data Source	Hypothetical	WEKA project	WEKA project	StatLib

4. Program verification and validation

This chapter illustrates the verification and validation processes including manual calculations for validation prepuces using Euclidian and Mahalanobis distances. Time cost measuring and stability of the number of generated solutions are reviewed.

www.manaraa.com

4.1 Program verification

The program was verified using simple data set. The result was as follows:

1- GUI: The program interface was as shown in Fig (4.1).

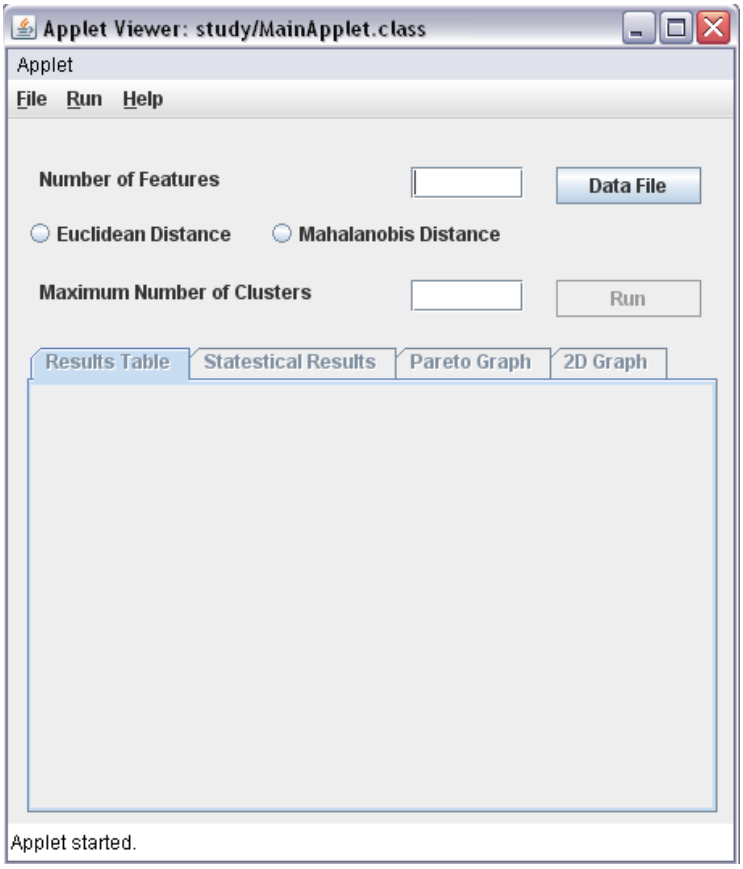


Fig (4.1): Program GUI

Text fields was filed to cluster the simple data set as following:

- Number of features = 2 since data have two features.
- The program recommend using Euclidian distance so Euclidian distance was selected.

www.manaraa.com

- Maximum number of clusters = 4 since data contains 5 points.

2- The program was ran and the results was as following:

- Main results:

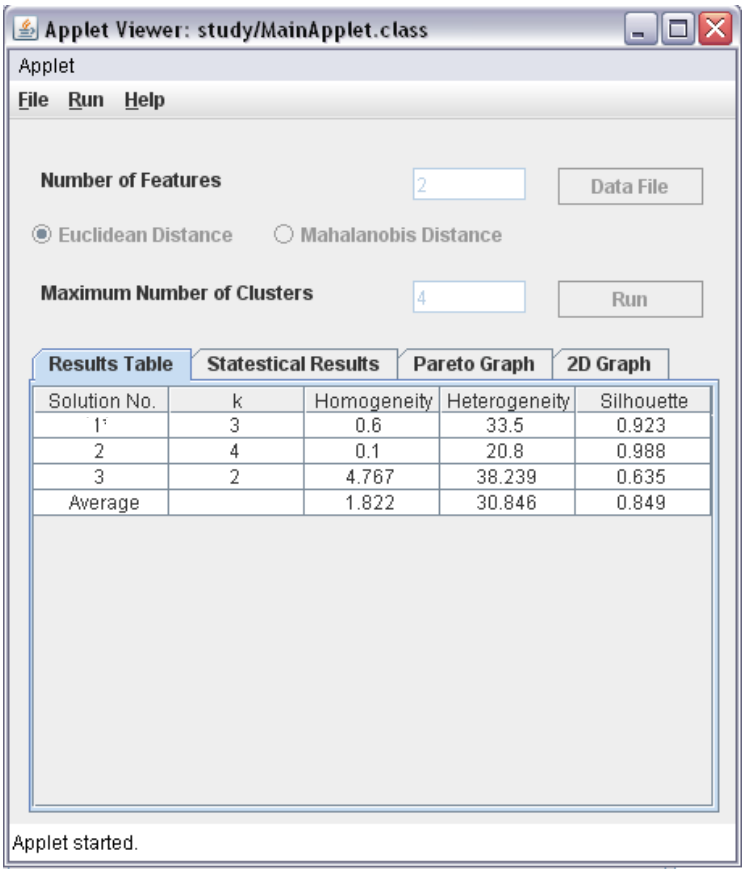


Fig (4.2): Main Results for simple data using Euclidian distance

- Statistical results: General statistics produced by the program are as following:

Input file C:\Documents and Settings\Administrator\My Documents\simple data.txt

Number of generations : 5

Time needed to solve the problem : 94.0 m Sec

www.manaraa.com

Solution No (1): 1 1 2 3 3

Homogeneity = 0.6

Heterogeneity = 33.5

Silhouette Width = 0.92

Solution No (2): 1 1 2 3 4

Homogeneity = 0.1

Heterogeneity = 20.8

Silhouette Width = 0.99

Solution No (3): 1 1 2 2 2

Homogeneity = 4.77

Heterogeneity = 38.24

Silhouette Width = 0.63

Average Homogeneity = 1.82

Average Heterogeneity = 30.85

Average Silhouette Width = 0.85

Total Number of Sulotions = 3

Statistics for solution number one are as following:

www.manaraa.com

Solution No (1): 1 1 2 3 3

Number of Clustures = 3

Cluster Number (1) : _____

Feature Number (1) : Mean = 1.0 , Min = 1.0 , Max = 1.0

Feature Number (2) : Mean = 1.5 , Min = 1.0 , Max = 2.0

Cluster Number (2) : _____

Feature Number (1) : Mean = 5.0 , Min = 5.0 , Max = 5.0

Feature Number (2) : Mean = 6.0 , Min = 6.0 , Max = 6.0

Cluster Number (3) : _____

Feature Number (1) : Mean = 7.5 , Min = 7.0 , Max = 8.0

Feature Number (2) : Mean = 1.0 , Min = 0.0 , Max = 2.0

Homogeneity = 0.6

Heterogeneity = 33.5

Silhouette Width = 0.92

Clusters Summary

Cluster Number (1): Number of Members = 2 (40 %)

www.manaraa.com

Cluster Number (2): Number of Members = 1 (20 %)

Cluster Number (3): Number of Members = 2 (40 %)

Statistics for solution number two are as following:

Solution No (2): 1 1 2 3 4

Number of Clustures = 4

Clusture Number (1) : _____

Feature Number (1) : Mean = 1.0 , Min = 1.0 , Max = 1.0

Feature Number (2) : Mean = 1.5 , Min = 1.0 , Max = 2.0

Clusture Number (2) : _____

Feature Number (1) : Mean = 5.0 , Min = 5.0 , Max = 5.0

Feature Number (2) : Mean = 6.0 , Min = 6.0 , Max = 6.0

Clusture Number (3) : _____

Feature Number (1) : Mean = 8.0 , Min = 8.0 , Max = 8.0

Feature Number (2) : Mean = 2.0 , Min = 2.0 , Max = 2.0

Clusture Number (4) : _____

Feature Number (1) : Mean = 7.0 , Min = 7.0 , Max = 7.0

المنارة للاستشارات
www.manaraa.com

Feature Number (2) : Mean = 0.0 , Min = 0.0 , Max = 0.0

Homogeneity = 0.1

Heterogeneity = 20.8

Silhouette Width = 0.99

Clusters Summary

Cluster Number (1): Number of Members = 2 (40 %)

Cluster Number (2): Number of Members = 1 (20 %)

Cluster Number (3): Number of Members = 1 (20 %)

Cluster Number (4): Number of Members = 1 (20 %)

Statistics for solution number three are as following:

Solution No (3): 1 1 2 2 2

Number of Clustures = 2

Clusture Number (1) : _____

Feature Number (1) : Mean = 1.0 , Min = 1.0 , Max = 1.0

Feature Number (2) : Mean = 1.5 , Min = 1.0 , Max = 2.0

www.manaraa.com

Clusture Number (2) : _____

Feature Number (1) : Mean = 6.67 , Min = 5.0 , Max = 8.0

Feature Number (2) : Mean = 2.67 , Min = 0.0 , Max = 6.0

Homogeneity = 4.77

Heterogeneity = 38.24

Silhouette Width = 0.63

Clusters Summary

Cluster Number (1): Number of Members = 2 (40 %)

Cluster Number (2): Number of Members = 3 (60 %)

- Pareto Graph: Pareto graph for the three solutions is shown in Fig (4.3).

www.manaraa.com

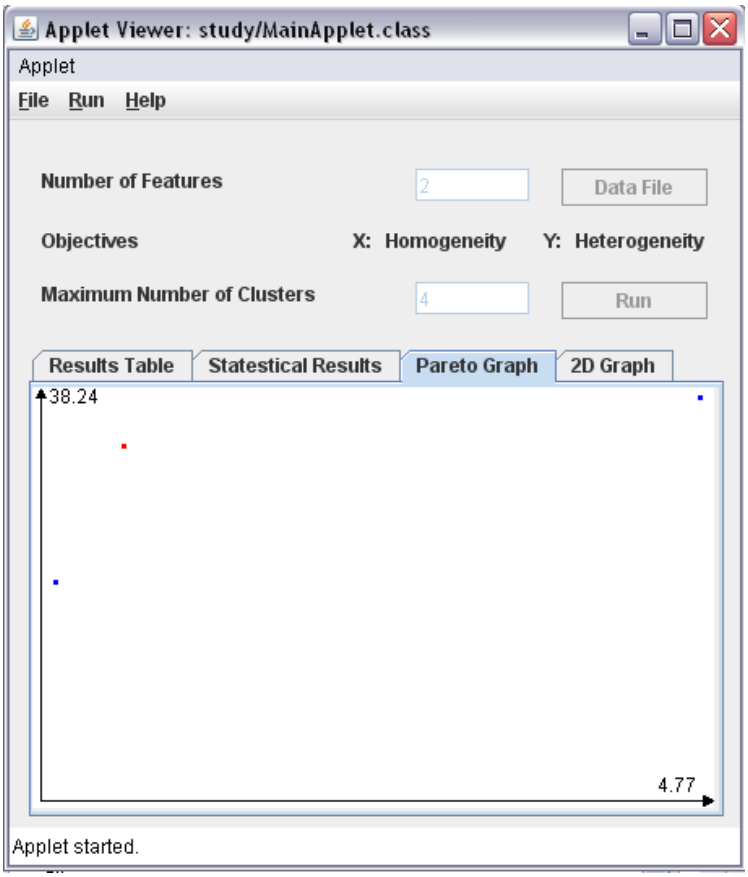


Fig (4.3): Pareto Graph for simple data

- 2D Graph: Graph for solutions number one, two and three are shown in Fig (4.4).

www.manaraa.com

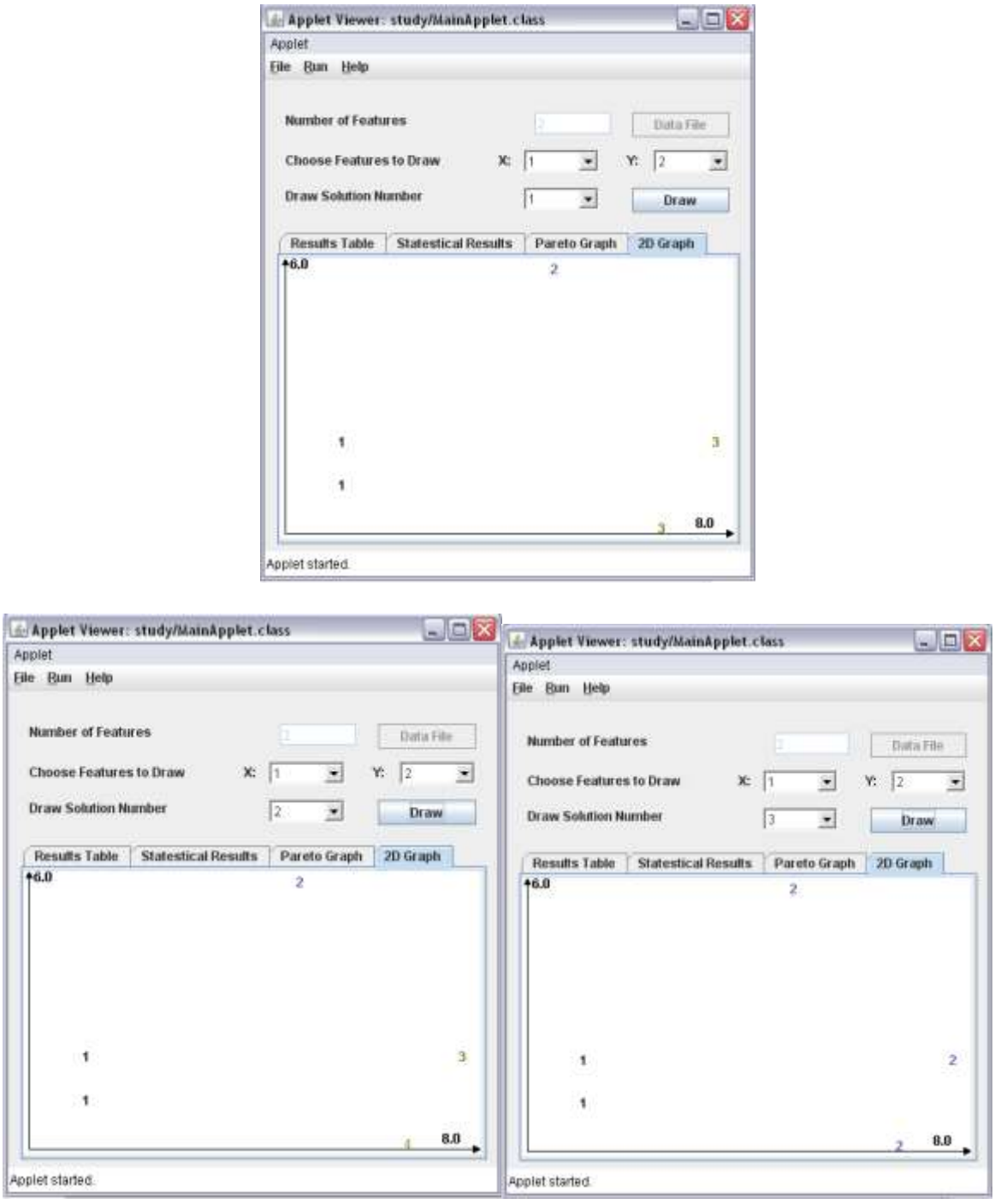


Fig (4.4): 2D Graph for simple data

www.manaraa.com

- *Verification summary*: The program ran using simple data to verify wither the program is running and gives results. The program gave three results for the problem in the main results table. The results statistics was generated by the program and the Pareto graph for the three solutions was drawn. The three solutions were drawn by the program. Since the program succeeded to generate the solutions for the problem, it means that the verification phase is successfully finished.

4.2. Program validation

To validate program manual calculation is performed to the basic data to assure the authenticity of the program results, after that standard data is to be clustered to study the performance of the algorithm.

- *Simple data clustering results:*

1- Solution (1):

Number of clusters = 3

Clusters centers:

Cluster (1): $X=\frac{1+1}{2}=1 \quad , \quad Y=\frac{1+2}{2}=1.5$

Cluster (2): $X=5 \quad , \quad Y=6$

Cluster (3): $X=\frac{8+7}{2}=7.5 \quad , \quad Y=\frac{2+0}{2}=1$

Homogeneity = $\frac{1}{N}\sum_{i=1}^{N} d(X_i, C_i)^2$

www.manaraa.com

$$= \frac{1}{5} \times \left| \begin{array}{l} ((1-1)^2 + (1-1.5)^2) + ((1-1)^2 + (2-1.5)^2) + 0 \\ + ((8-7.5)^2 + (2-1)^2) + ((7-7.5)^2 + (0-1)^2) \end{array} \right| = 0.6$$

Heterogeneity = $\frac{1}{N} \sum_{i=1}^{N} d(X_i, C_{\min})^2$

$$= \frac{1}{5} \times \left| \begin{array}{l} ((1-5)^2 + (1-6)^2) + ((1-5)^2 + (2-6)^2) + ((5-7.5)^2 + \\ (6-1)^2) + ((8-5)^2 + (2-6)^2) + ((7-1)^2 + (0-1.5)^2) \end{array} \right| = 33.5$$

Silhouette width = $\frac{1}{N} \sum_{i=1}^{N} \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$

$$= \frac{1}{5} \times \left| \frac{41-1}{41} + \frac{32-1}{32} + \frac{32.5-0}{32.5} + \frac{25-5}{25} + \frac{38.5-5}{38.5} \right| = 0.922$$

2- Solution (2):

Number of clusters = 4

Clusters centers:

Cluster (1): $X = \frac{1+1}{2} = 1 \quad , \quad Y = \frac{1+2}{2} = 1.5$

Cluster (2): $X = 5 \quad , \quad Y = 6$

Cluster (3): $X = 8 \quad , \quad Y = 2$

Cluster (4): $X = 7 \quad , \quad Y = 0$

Homogeneity = $\frac{1}{N} \sum_{i=1}^{N} d(X_i, C_i)^2$

www.manaraa.com

$$=\frac{1}{5}\times\left[((1-1)^2+(1-1.5)^2)+((1-1)^2+(2-1.5)^2)+0+0+0\right]=0.1$$

Heterogeneity = $\frac{1}{N}\sum_{i=1}^{N} d(X_i, C_{\min})^2$

$$=\frac{1}{5}\times\left|\begin{array}{l}((1-7)^2+(1-0)^2)+((1-5)^2+(2-6)^2)+((5-8)^2+\\(6-2)^2)+((8-7)^2+(2-0)^2)+((7-8)^2+(0-2)^2)\end{array}\right|=20.8$$

Silhouette width = $\frac{1}{N}\sum_{i=1}^{N}\frac{b(i)-a(i)}{\max\{a(i),b(i)\}}$

$$=\frac{1}{5}\times\left[\frac{37-1}{37}+\frac{32-1}{32}+\frac{25-0}{25}+\frac{5-0}{5}+\frac{5-0}{5}\right]=0.988$$

3- Solution (3):

Number of clusters = 2

Clusters centers:

Cluster (1): $X=\frac{1+1}{2}=1$, $Y=\frac{1+2}{2}=1.5$

Cluster (2): $X=\frac{8+7+5}{3}=6.667$, $Y=\frac{2+0+6}{3}=2.667$

Homogeneity = $\frac{1}{N}\sum_{i=1}^{N} d(X_i, C_i)^2$

$$=\frac{1}{5}\times\left|\begin{array}{l}((1-1)^2+(1-1.5)^2)+((1-1)^2+(2-1.5)^2)\\+((5-6.667)^2+(6-2.667)^2)+((8-6.667)^2\\+(2-2.667)^2)+((7-6.667)^2+(0-2.667)^2)\end{array}\right|=4.767$$

www.manaraa.com

Heterogeneity = $\frac{1}{N}\sum_{i=1}^{N} d(X_i, C_{\min})^2$

$$= \frac{1}{5} \times \left[\begin{array}{l} ((1-6.667)^2 + (1-2.667)^2) + ((1-6.667)^2 + (2-2.667)^2) + ((5-1)^2 \\ +(6-1.5)^2) + ((8-1)^2 + (2-1.5)^2) + ((7-1)^2 + (0-1.5)^2) \end{array} \right] = 38.24($$

Silhouette width = $\frac{1}{N}\sum_{i=1}^{N} \frac{b(i)-a(i)}{\max\{a(i), b(i)\}}$

$$= \frac{1}{5} \times \left[\frac{4.2667-1}{4.2667} + \frac{4.0333-1}{4.0333} + \frac{3.65-3.25}{3.65} + \frac{5.15-1.5}{5.15} + \frac{3.85-2.25}{3.85} \right] = 0.63$$

Table (4.1) compare the results that computer generates automatically with the results that calculated manually when Euclidian distance is used.

Table (4.1): Comparison between computer results (C) and manual results (M)

Solution No.	*k*	Cluster Center				Homogeneity		Heterogeneity		Silhouette width	
		X		Y							
		C	M	C	M	C	M	C	M	C	M
1	3	1 5 7.5	1 5 7.5	1.5 6 1	1.5 6 1	0.6	0.6	33.5	33.5	0.923	0.923
2	4	1 5 8 7	1 5 8 7	1.5 6 2 0	1.5 6 2 0	0.1	0.1	20.8	20.8	0.988	0.988
3	2	1 6.67	1 6.667	1.5 2.67	1.5 2.667	4.767	4.767	38.24	38.24	0.635	0.637

www.manaraa.com

To check the correctness of Mahalanobis distance calculation, the program is restarted using Mahalanobis distance. The program achieves the same results using Euclidian distance. The results are shown in Fig (4.5).

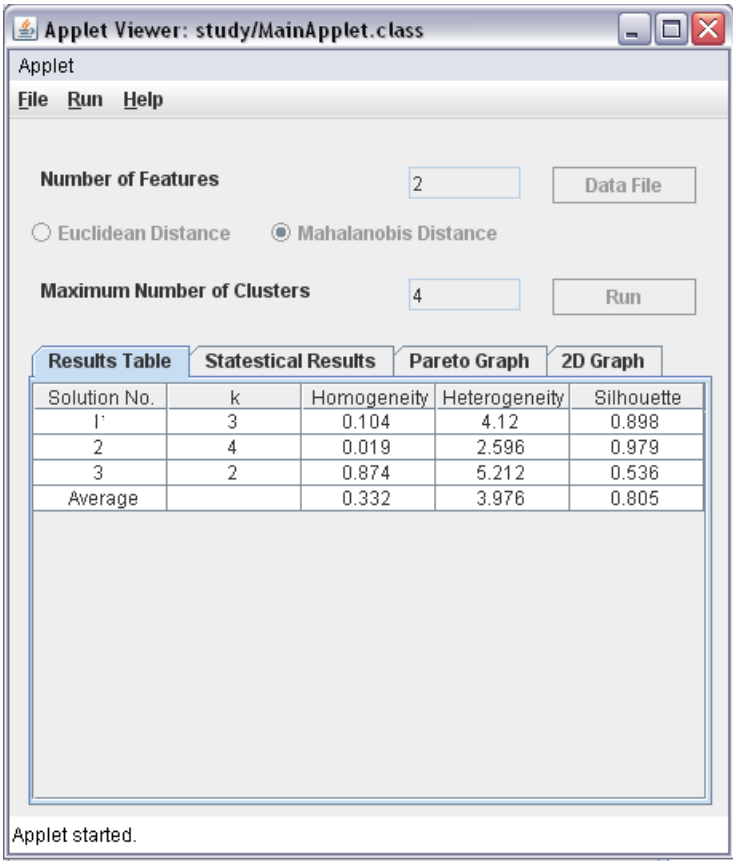


Fig (4.5): Main Results for simple data using Mahalanobis distance

- Mahalanobis distance calculations:

$$X = \frac{1+1+5+8+7}{5} = 4.4 \qquad Y = \frac{1+2+6+2+0}{5} = 2.2$$

$$\sum(0,0) = \frac{1}{4} \times ((1-4.4)^2 + (1-4.4)^2 (5-4.4)^2 + (8-4.4)^2 + (7-4.4)^2) = 10.8$$

www.manaraa.com

$$\sum(1,1)=\frac{1}{4}\times((1-2.2)^2+(2-2.2)^2+(6-2.2)^2+(2-2.2)^2+(0-2.2)^2=5.2$$

$$\sum(0,1)=\sum(1,0)=\frac{1}{4}\times((1-4.4)\times(1-2.2)+(1-4.4)\times(2-2.2)+(5-4.4)\times(6-2.2)+$$
$$(8-4.4)\times(2-2.2)+(7-4.4)\times(0-2.2))=0.15$$

$$\sum=\begin{bmatrix}1.08 & 0.15\\ 0.15 & 5.2\end{bmatrix} \qquad \sum^{-1}=\begin{bmatrix}0.09263 & -0.00267\\ -0.00267 & 0.19238\end{bmatrix}$$

- Homogeneity for solution (2) = $\frac{1}{N}\sum_{i=1}^{N} d(X_i,C_i)^2$

$$=\frac{1}{5}\times\begin{bmatrix}([1-1 \quad 1-1.5]\times\begin{bmatrix}0.09263 & -0.00267\\ -0.00267 & 0.19238\end{bmatrix}\times\begin{bmatrix}1-1\\ 1-1.5\end{bmatrix})+\\ ([1-1 \quad 2-1.5]\times\begin{bmatrix}0.09263 & -0.00267\\ -0.00267 & 0.19238\end{bmatrix}\times\begin{bmatrix}1-1\\ 2-1.5\end{bmatrix})+0+0+0\end{bmatrix}=0.019$$

Computer results show that the homogeneity for solution (2) is equal to 0.019.

- *Time cost and stability of generated solutions*:

Time cost and stability of the generated solution is a measure of the goodness of the stopping criterion. Iris and CPU data were used to measure time consumption and the satiability in the solution number generated by the algorithm. Stable number of solutions needs more time to generate and vice versa. Good stopping criterion will generate final solutions with acceptable time cost and stability according to the number of solutions. Different number of k_{max} was used to check the best number of k_{max} to use.

www.manaraa.com

- Iris data clustering results:

Iris data was clustered with the program using deferent number of k_{max}. The program was operated ten times for each number of k_{max}. Results for time (T) in seconds, number of solutions (N) and number of generation (G) are shown in Table (4.2).

Table (4.2): Iris data clustering results.

K_{max}	Measure	Operation Number										Mean
		1	2	3	4	5	6	7	8	9	10	
$\sqrt{l}$ (13)	T	10	17	25	7	11	11	16	7	7	7	11.8
	N	55	46	53	46	56	48	47	44	54	49	49.8
	G	12	20	26	8	13	13	20	8	8	8	13.6
$l/2$ 75	T	309	142	461	253	179	435	232	99	123	426	265.9
	N	134	141	177	168	154	197	121	130	123	177	152.2
	G	50	23	72	44	29	74	26	15	16	59	40.8
$l-1$ 149	T	147	591	612	298	1015	409	1125	623	626	731	617.7
	N	137	204	185	140	212	175	207	179	195	182	181.6
	G	11	42	44	19	69	32	88	46	49	56	45.6

- CPU data clustering results:

CPU data was clustered with the program using different number of k_{max}. The program was operated ten times for each number of k_{max}. Results for time (T) in seconds,

www.manaraa.com

number of solutions (N) and number of generation (G) are shown in Table (4.3).

Table (4.3): CPU data clustering results.

K_{max}	Measure	Operation Number										Mean
		1	2	3	4	5	6	7	8	9	10	
$\sqrt{l}$ (15)	T	34	23	18	25	17	21	12	17	23	16	20.6
	N	25	19	19	20	16	19	22	22	25	20	20.7
	G	23	16	11	15	11	14	9	12	15	9	13.5
$l/2$ 105	T	155	77	228	92	262	144	135	146	110	154	150.3
	N	25	24	22	28	26	31	29	29	24	23	26.1
	G	17	9	19	10	31	18	17	18	14	21	17.4
$l-1$ 208	T	296	239	190	208	205	139	230	279	146	164	209.6
	N	25	23	26	23	26	23	28	27	27	21	24.9
	G	25	19	17	15	17	10	20	23	11	13	17

- *Validation summary*: Manual calculations were performed to the simple data results to validate the program calculations of homogeneity, heterogeneity and Silhouette width. The manual calculations were identical to those generated by the program as shown in Table (4.1) with some deviations according to rounding. The program was ran again using Mahalanobis to check whether program calculations of Mahalanobis distance are correct or not. Manual calculation was found identical to computer generated. Iris data was clustered using the

www.manaraa.com

algorithm and the following can be noticed:

1- Time cost is acceptable when k_{max} is considered as $\sqrt{l}$ =13.

2- Time cost was high when k_{max} is considered as $l/2$ =75 and $l-1$ =149.

3- Number of generated solutions increases when k_{max} increases.

4- Using large value of k_{max} will generate unreasonable solutions.

5- Number of generated solutions is almost stable with some deviations due to the randomness that GA included.

CPU data was clustered using the algorithm and the following can be noticed:

1- Time cost is acceptable when k_{max} is considered as $\sqrt{l}$ =15.

2- Time cost was high when k_{max} is considered as $l/2$ =105 and $l-1$ =208.

3- Number of generated solutions not increases when k_{max} increases.

4- Using large value of k_{max} will generate the same solutions.

5- Number of generated solutions is almost stable with some deviations due to the randomness that GA included.

In all cases using $k_{max} = \sqrt{l}$ generates reasonable and stable number of solutions with acceptable time cost.

www.manaraa.com

5. Case study

In this chapter a case study will be illustrated where multi-objective *k*-means clustering enhanced genetic algorithm is used. The problem was taken from StatLib website.

5.1. Problem statement

Data are collected from an experiment on the effects of machine adjustments on the time to count bolts.

A manufacturer of automotive accessories provides hardware, e.g. nuts, bolts, washers and screws, to fasten the accessory to the car or truck. Hardware is counted and packaged automatically. Specifically, bolts are dumped into a large metal dish. A plate that forms the bottom of the dish rotates counterclockwise. This rotation forces bolts to the outside of the dish and up along a narrow ledge. Due to the vibration of the dish caused by the spinning bottom plate, some bolts fall off the ledge and back into the dish. The ledge spirals up to a point where the bolts are allowed to drop into a pan on a conveyor belt. As a bolt drops, it passes by an electronic eye that counts it. When the electronic counter reaches the preset number of bolts, the rotation is stopped and the conveyor belt is moved forward.

There are several adjustments on the machine that affect its operation. These include; a speed setting that controls the speed of rotation (SPEED1) of the plate at the bottom of the dish, a total number of bolts (TOTAL) to be counted, a second speed setting (SPEED2) that is used to change the speed of rotation (usually slowing it down) for the last few bolts, the number of bolts to be counted at this second speed (NUMBER2), and the sensitivity of the

www.manaraa.com

electronic eye (SENS). The sensitivity setting is to insure that the correct number of bolts is

counted. Too few bolts packaged causes customer complaints. Too many bolts packaged increases costs. For each run conducted in this experiment the correct number of bolts was counted. From an engineering standpoint if the correct number of bolts is counted, the sensitivity should not affect the time to count bolts. The measured response is the time (TIME) in seconds, it takes to count the desired number of bolts. In order to put times on an equal footing the response to be analyzed is the time to count 20 bolts (T20BOLT). The data for 40 combinations of settings are shown in Table (5.1). RUN is the order in which the data were collected.

Our objective is to analyze data, find what adjustments have the greatest effect on the time to count 20 bolts, cluster data set using conventional *k*-means and multi-objective *k*-means, find regression models for resulting clusters and compare R^2_{adj} for regression models.

www.manaraa.com

Table (5.1): Bolts Data

i	RUN	SPEED1	TOTAL	SPEED2	NUMBER2	SENS	TIME	T20BOLT
1	25	2	10	1.5	0	6	5.7	11.4
2	24	2	10	1.5	0	10	17.56	35.12
3	30	2	10	1.5	2	6	11.28	22.56
4	2	2	10	1.5	2	10	8.39	16.78
5	40	2	10	2.5	0	6	16.67	33.34
6	37	2	10	2.5	0	10	12.04	24.08
7	16	2	10	2.5	2	6	9.22	18.44
8	22	2	10	2.5	2	10	3.94	7.88
9	33	2	30	1.5	0	6	27.02	18.01
10	17	2	30	1.5	0	10	19.46	12.97
11	28	2	30	1.5	2	6	18.54	12.36
12	27	2	30	1.5	2	10	25.7	17.13
13	14	2	30	2.5	0	6	19.02	12.68
14	13	2	30	2.5	0	10	22.39	14.93
15	4	2	30	2.5	2	6	23.85	15.9
16	21	2	30	2.5	2	10	30.12	20.08
17	23	6	10	1.5	0	6	13.42	26.84
18	35	6	10	1.5	0	10	34.26	68.52
19	19	6	10	1.5	2	6	39.74	79.48
20	34	6	10	1.5	2	10	10.6	21.2
21	31	6	10	2.5	0	6	28.89	57.78
22	9	6	10	2.5	0	10	35.61	71.22
23	38	6	10	2.5	2	6	17.2	34.4
24	15	6	10	2.5	2	10	6	12
25	39	6	30	1.5	0	6	129.45	86.3
26	8	6	30	1.5	0	10	107.38	71.59
27	26	6	30	1.5	2	6	111.66	74.44
28	11	6	30	1.5	2	10	109.1	72.73
29	6	6	30	2.5	0	6	100.43	66.95
30	20	6	30	2.5	0	10	109.28	72.85
31	10	6	30	2.5	2	6	106.46	70.97
32	32	6	30	2.5	2	10	134.01	89.34
33	1	4	20	2	1	8	10.78	10.78
34	3	4	20	2	1	8	9.39	9.39
35	5	4	20	2	1	8	9.84	9.84
36	7	4	20	2	1	8	13.94	13.94
37	12	4	20	2	1	8	12.33	12.33
38	18	4	20	2	1	8	7.32	7.32
39	29	4	20	2	1	8	7.91	7.91
40	36	4	20	2	1	8	15.58	15.58

www.manaraa.com

5.2. Problem analysis and solution

To solve the problem we need to build regression models to predict time to count 20 bolts. First a stepwise regression will be performed to find adjustments having greatest effect on total time to count 20 bolts then clustering-regression models will be built in two methods as following.

- Clustering-regression using conventional *k*-means: Clustering-regression model will be built using conventional *k*-means.

- Clustering-regression using multi-objective *k*-means: In this method clustering using multi-objective *k*-means enhanced by genetic algorithm will be performed to the data. Clustering results will be used to build *k* regression model for each *k* cluster.

Solutions having two clusters where clusters have enough records to build a good regression model will be used to solve problem.

5.2.1. Stepwise regression

Minitab software is used to build a stepwise regression model. The response in the model is the total time to count 20 bolts and the predictors were speed No.1, total number of bolts, speed No.2, number of bolts counted in speed No.2 and sensor adjustments. Stepwise

www.manaraa.com

regression (forward and backward) was used to select variables for the regression model and the results were as following:

Stepwise Regression:

```
 Alpha-to-Enter: 0.15  Alpha-to-Remove: 0.15

 Response is T20BOLT  on  5 predictors, with N =   40


    Step          1        2
Constant     -8.750  -20.512

SPEED1         10.7     10.7
T-Value        6.13     6.29
P-Value       0.000    0.000

TOTAL                   0.59
T-Value                 1.73
P-Value                0.091

S              19.7     19.2
R-Sq          49.74    53.52
R-Sq(adj)     48.42    51.00
C-p             1.7      0.8
```

A regression model was built using speed No.1 and total number of bolts variables to predict time to count 20 bolts.

5.2.2. Clustering-regression using conventional *k*-means

Weka software was used to cluster predictors' data using conventional *k*-means. Clustering repeated 10 times with different seeds and the results was as shown in Table (5.2).

Table (5.2): *k*-means clustering results

Seed	1	2	3	4	5	6	7	8	9	10
SSE	9.33	11	11	9.33	9.33	11	11	11	9.33	11

The solution of *k*-means having minimum SSE (with seed 1, 4, 5 and 9) is to be used. Clustering solution is as following:

www.manaraa.com

Solution: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

Regression models for the clustering solution of conventional *k*-means are shown in the following Table (5.3).

Table (5.3): Regression models for conventional *k*-means solution

Cluster No.	Stepwise result	Model T20BOLT =	R^2_{adj}
C1	$Total^2$	21.9 - 0.00712 $Total^2$	15.5%
C2	Speed1 Speed1 X Total	- 89.4 + 20.2 Speed1 + 0.243 Speed1 X Total	74.3%

5.2.3. Clustering-regression using multi-objective *k*-means

Now we will build *k* regression model by clustering prediction variables data into *k* cluster using our program and build a new model for each cluster using Minitab software.

- *Clustering parameters*:

  - Maximum number of clusters = $\sqrt{l} \approx 6$.
  - Distance type is Mahalanobis distance.
  - Variables: SPEED1 and TOTAL.

- *Clustering results*: Table (5.4) shows the results summary.

www.manaraa.com

Table (5.4): Multi-objective *k*-means result summary

Solution No.	*K*	Homogeneity	Heterogeneity	Silhouette
1	2	1.21	4.647	0.416
2	4	0.22	3.14	0.835
3	4	0.22	3.14	0.835
4	3	0.56	3.96	0.722
5	2	1.21	4.647	0.416
6	5	0.0	2.2	1.0
7	2	1.21	4.647	0.416
8	2	0.96	4.293	0.572
9	4	0.22	3.14	0.835
10	3	0.56	3.96	0.722
11	2	1.21	4.647	0.416
Average		0.689	3.857	0.653

One of the solutions must be picked from the solutions set. The criterion to select one of the solutions is to check all solutions having two clusters where clusters have enough records to build a good regression model. Table (5.5) shows number of instance in each cluster and cluster details for all two clusters solutions.

Table (5.5): Number of instances in clusters of each solution of bolts data

Solution No.	*K*	C1	C2
1	2	32	8
5	2	8	32
7	2	32	8
8	2	16	24
11	2	32	8

Solution details are as follows:

www.manaraa.com

- Solution (1): 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1

Clusture Number (1) : Number of Members = 32 (80 %)

Feature Number (1) : Mean = 3.5 , Min = 2.0 , Max = 6.0

Feature Number (2) : Mean = 17.5 , Min = 10.0 , Max = 30.0

Clusture Number (2) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 6.0 , Min = 6.0 , Max = 6.0

Feature Number (2) : Mean = 30.0 , Min = 30.0 , Max = 30.0

- Solution (5): 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

Clusture Number (1) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 2.0 , Min = 2.0 , Max = 2.0

Feature Number (2) : Mean = 10.0 , Min = 10.0 , Max = 10.0

Clusture Number (2) : Number of Members = 32 (80 %)

Feature Number (1) : Mean = 4.5 , Min = 2.0 , Max = 6.0

Feature Number (2) : Mean = 22.5 , Min = 10.0 , Max = 30.0

- Solution (7): 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Clusture Number (1) : Number of Members = 32 (80 %)

Feature Number (1) : Mean = 4.5 , Min = 2.0 , Max = 6.0

Feature Number (2) : Mean = 17.5 , Min = 10.0 , Max = 30.0

Clusture Number (2) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 2.0 , Min = 2.0 , Max = 2.0

Feature Number (2) : Mean = 30.0 , Min = 30.0 , Max = 30.0

www.manaraa.com

- Solution (8): 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

Clusture Number (1) : Number of Members = 16 (40 %)

Feature Number (1) : Mean = 2.0 , Min = 2.0 , Max = 2.0

Feature Number (2) : Mean = 20.0 , Min = 10.0 , Max = 30.0

Clusture Number (2) : Number of Members = 24 (60 %)

Feature Number (1) : Mean = 5.33 , Min = 4.0 , Max = 6.0

Feature Number (2) : Mean = 20.0 , Min = 10.0 , Max = 30.0

- Solution (11): 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Clusture Number (1) : Number of Members = 32 (80 %)

Feature Number (1) : Mean = 3.5 , Min = 2.0 , Max = 6.0

Feature Number (2) : Mean = 22.5 , Min = 10.0 , Max = 30.0

Clusture Number (2) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 6.0 , Min = 6.0 , Max = 6.0

Feature Number (2) : Mean = 10.0 , Min = 10.0 , Max = 10.0

Table (5.6) shows the regression results for the clustering solutions including stepwise regression results and regression model.

www.manaraa.com

Table (5.6): Regression models for multi-objective *k*-means solution

Solution	Cluster No.	Stepwise result	Model T20BOLT =	R^2_{adj}
1	C1	SPEED1 SPEED1^3	47.3 - 16.3 SPEED1 + 0.448 SPEED1^3	48.1%
	C2	--	--	--
5	C1	--	--	--
	C2	SPEED1^3 TOTAL^3	16.7 + 0.288 SPEED1^3 + 0.00111 TOTAL^3	79.4%
7	C1	SPEED1^2 SPEED1^3 TOTAL^3	39.5 - 7.38 Speed1^2 + 1.26 Speed1^3 + 0.00112 Total^3	75.1%
	C2	--	--	--
8	C1	TOTAL^2	21.9 - 0.00712 TOTAL^2	9.5%
	C2	SPEED1^3 SPEED1 X TOTAL	25.6 + 0.266 SPEED1$^{(3)}$ + 0.243 SPEED1*TOTAL	74.3%
11	C1	SPEED1 SPEED1^3 SPEED1 X TOTAL	66.2 - 24.4 SPEED1 + 0.841 SPEED1^3 - 0.142 SPEED1 X TOTAL	94.3%
	C2	--	--	--

www.manaraa.com

Fig (5.1) shows a comparison between all two cluster solutions.

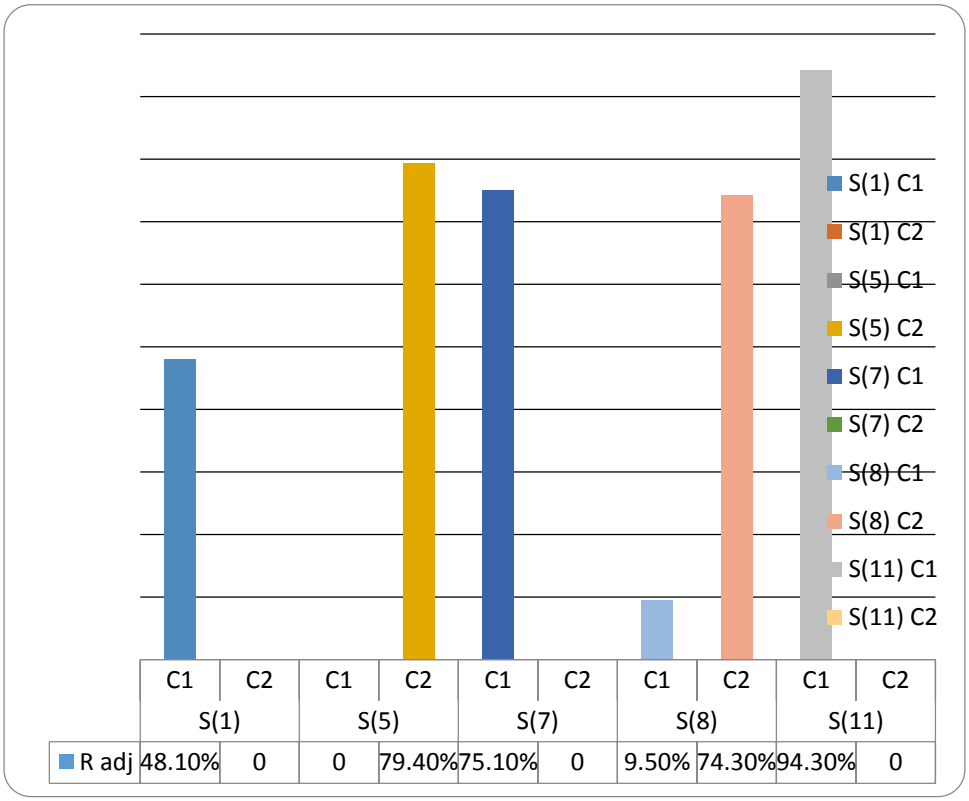


Fig (5.1): comparison of R^2_{adj} for two clusters Solutions

As shown in Fig (5.1) solution No.11 is the best solution among the alternatives. Solution No.11 will be compared to the conventional model in the results analysis.

5.3. Analysis

When a conventional *k*-means regression models was used to built regression models to predict the time to count 20 bolts which are speed No.1 and total number of bolts. The models were as follows:

Cluster (1): T20BOLT = 21.9 - 0.00712 Total^2.

R^2_{adj} for the model is equal to 15.50%.

www.manaraa.com

Cluster (2): T20BOLT = - 89.4 + 20.2 Speed1 + 0.243 Speed1 X Total

R^2_{adj} for the model is equal to 74.3%.

In Multi-objective *k*-means clustering-regression model (Solution No.11), the results were as following:

- Cluster (1): There was one adjustment effects the time to count 20 bolts which is speed No.1 and total number of bolts. The model was as follows:

T20BOLT = 66.2 - 24.4 SPEED1 + 0.841 SPEED1^3 - 0.142 SPEED1 X TOTAL

R^2_{adj} for the model is equal to 94.30%.

- Cluster (2): In this cluster SPEED1 and TOTAL variables are constants and the total time to count 20 bolts is weakly correlated to other of the predictors which mean that the deviation of the response is due to variability only.

To compare the standard deviation in this cluster to other data in the data set we can use clustering solution No.6 which cluster the data set into five equal size clusters where SPEED1 and TOTAL (effective variables) are constants in each cluster, then we can measure the standard deviation in each cluster and compare the variability in each cluster. Solution No.6 details are as following:

Solution (3): 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5

Clusture Number (1) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 2.0 , Min = 2.0 , Max = 2.0s

Feature Number (2) : Mean = 10.0 , Min = 10.0 , Max = 10.0

Clusture Number (2) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 2.0 , Min = 2.0 , Max = 2.0

www.manaraa.com

Feature Number (2) : Mean = 30.0 , Min = 30.0 , Max = 30.0

Clusture Number (3) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 6.0 , Min = 6.0 , Max = 6.0

Feature Number (2) : Mean = 10.0 , Min = 10.0 , Max = 10.0

Clusture Number (4) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 6.0 , Min = 6.0 , Max = 6.0

Feature Number (2) : Mean = 30.0 , Min = 30.0 , Max = 30.0

Clusture Number (5) : Number of Members = 8 (20 %)

Feature Number (1) : Mean = 4.0 , Min = 4.0 , Max = 4.0

Feature Number (2) : Mean = 20.0 , Min = 20.0 , Max = 20.0

Table (5.7) shows the instances in each cluster and standard deviation in each cluster in the solution No.6. Fig (5.2) shows box blot for T20BOLTS for different predictors regions.

Table (5.7): Comparison of variability in clusters of solution No.6

Cluster No.	SPEED1	TOTAL	Instances	Σ
Cluster (1)	2	10	from 1 to 8	9.650362
Cluster (2)	2	30	from 9 to 16	2.793634
Cluster (3)	6	10	from 17 to 24	25.84046
Cluster (4)	6	30	from 25 to 32	7.862615
Cluster (5)	4	20	from 33 to 40	2.88746

www.manaraa.com

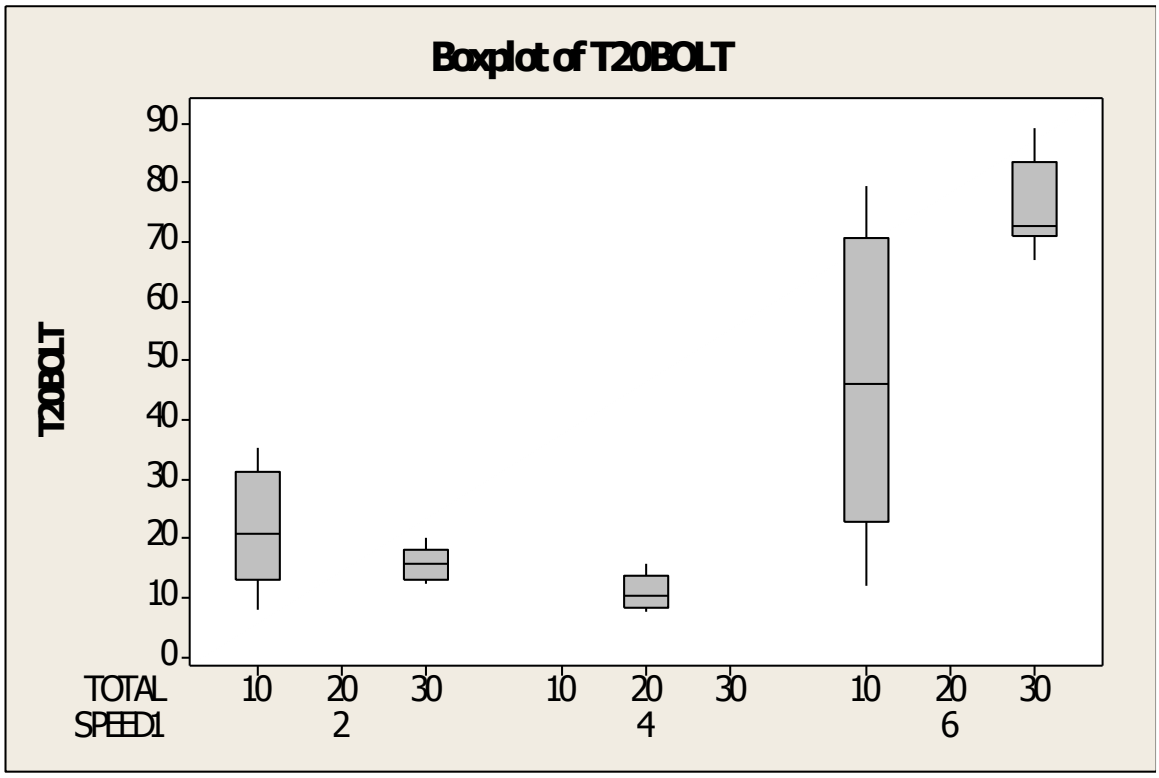


Fig (5.2): Box blot for T20BOLTS

We can see that the standard deviation in the third cluster is much higher than other clusters. Instances of cluster No.3 is the same as instances in the cluster No.2 in solution No.11. In clustering- regression models method when instances from 17 to 24 which have a higher variability compared to other data in the data set are isolated the model improved very much and the disturbance of the instances having SPEED1 = 6 and TOTAL = 10 simultaneously is removed. We can conclude that adjusting SPEED1 to be equal to 6 and TOTAL to be equal to 10 simultaneously cannot be used according to the high variability in the response time. Fig (5.3) shows a comparison between conventional *k*-means models and multi-objective *k*-means model.

www.manaraa.com

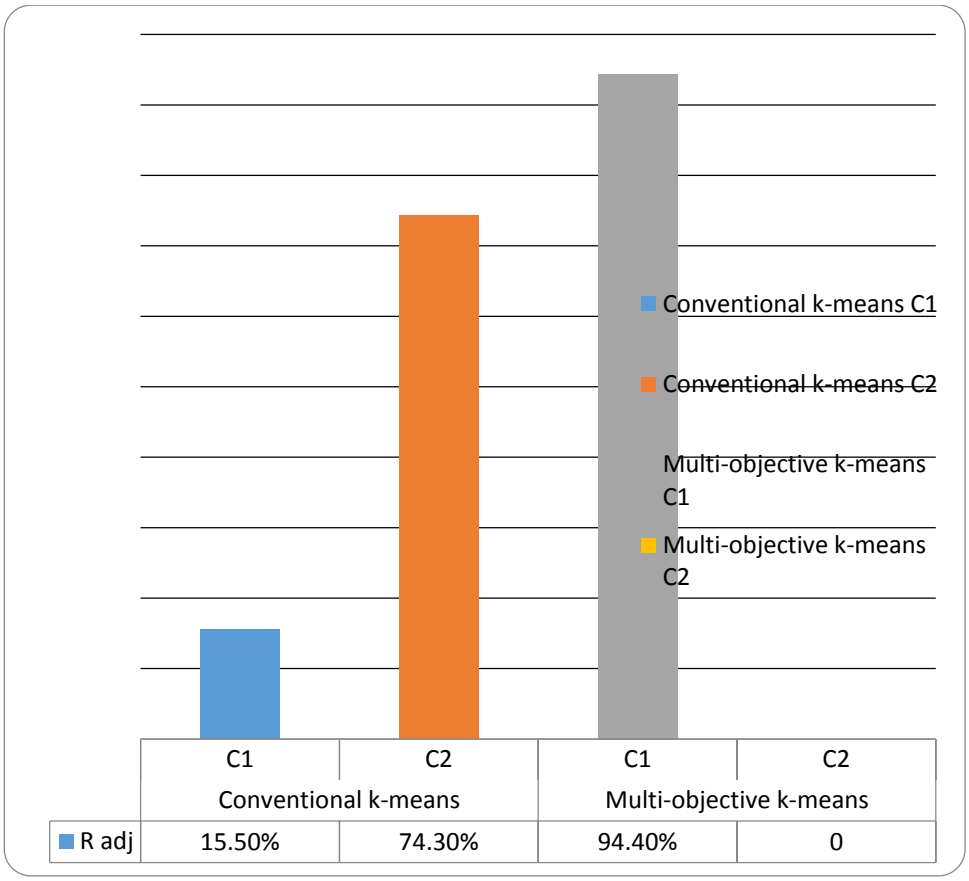


Fig (5.3): Comparison of R^2_{adj} for two models

As we can see in the results, multi-objective model has advantages over conventional model as following:

- R^2_{adj} for multi-objective model is much higher than conventional.
- Multi-objective model associated the influence to a specific predictors region.
- With multi-objective model we can predict wither a new point is following the model or it's an influential point.
- Clustering alternatives was helpful in this study.

www.manaraa.com

6. Conclusions and recommendations

6.1. Concluding remarks

In this thesis clustering problem has been solved using multi-objective Pareto optimization enhanced by genetic algorithm. The developed methodology succeeded to find clustering solutions alternatives by minimizing homogeneity and maximize heterogeneity simultaneously. Developed methodology consists of five main steps. In step 1, methodology initiates a random generation of chromosomes where each chromosome is consisting of a series of gene. Chromosome length is equal to data set size and every gene in the chromosome is a candidate in initial clusters centers set. In step 2, solution generation is clustered using *k*-means algorithm where local optimum of clustering problem is found, then in step 3 homogeneity and heterogeneity of each solution is measured and dummy fitness is evaluated. In step 4, genetic algorithm used to improve solution generations until stopping criterion is reached. Finally in step 5, nondominated solutions in the final generation are reported.

6.1.1 Advantages of developed methodology

The developed multi-objective *k*-means data clustering using genetic algorithm has the following advantages:

1- There is no priority vector used or scalarization process that influences the final solutions.
2- Alternative solutions are available to decision makers.
3- Methodology can handle elapse shaped clusters as well as ball-shaped because it uses Mahalanobis distance beside Euclidean distance.

www.manaraa.com

4- There is no need to pre-determine the cluster number.

6.1.2. Limitation of the developed methodology

The developed multi-objective *k*-means data clustering using genetic algorithm has the following limitations:

1- Time cost limitation: time cost of the developed methodology is high especially when the data set size is large.

2- Number of solutions in the last phase is large especially when the size of the data set is large. There is a need for exploring criterion among final solution set.

6.2. Thesis contribution

In this thesis, a new methodology is proposed to solve the clustering problem using multi-objective optimization with genetic algorithm. The methodology gives the decision maker clustering alternatives with no need to pre-determine the number of clusters. This methodology had a number of contributions that can be summarized in the following points:

1- The combination between Pareto optimization and *k*-means algorithm as shown in chapter 3 is new.

2- Using clustering alternatives was used to extend clustering-regression method as shown in chapter 5 is new.

6.3. Future work

After developing of multi-objective *k*-means clustering using genetic algorithm, the following studies are recommended:

www.manaraa.com

1- Enhancing the developed methodology using new algorithms to reduce time cost.

2- Using the developed methodology with adapted objectives.

3- Using the developed methodology with other clustering algorithms such as hierarchical clustering.

4- Using the developed methodology with fuzzy clustering.

www.manaraa.com

REFERENCES

Bolts data set submitted by W. Robert Stephenson, Iowa State University, from **http://www.lib.stat.cmu.edu**.

Cheung Y.M. (2003), k^*-Mans: A new generalized *k*-means clustering algorithm, **Pattern Recognition Letters**, 15(24), 2883-2893.

Du J., Korkmaz E.E., Alhajj R., and Barker K. (2005). Alternative Clustering by Utilizing Multi-objective Genetic Algorithm with Linked-List Based Chromosome Encoding: P. Perner and A. Imiya (Eds.), **Machine Learning and Data Mining in Pattern Recognition**, pp. 346–355.Berlin: Springer-Verlag.

Ettler P., Valečková M., Kárný M. and Puchr I. (2000), Towards a Knowledge-Based Control of a Complex Industrial Process, **American Control Conference**, Vol 3, Chicago, Illinois, 2063-2066.

Eskandari H., Rebelo L. and Mollaghasemi M. (2005). Multiobjective Simulation Optimization Using Enhanced Genetic Algorithm, **Winter Simulation Conference**, Vol 00, Orlando, FL, 833-841.

Fahim A.M., Salem A.M., Torkey F.A. and Ramadan M.A. (2006), An efficient enhanced k-means clustering algorithm, **Journal of Zhejiang University A**, 7(10), 1626-1633.

Haiyan P., Jun Z. and Danfu H. (2003), Genetic Algorithm Applied to Multi-Class Clustering for Gene Expression Data, **Geno., Prot. & Bioinfo**, 1(4), 279-287.

www.manaraa.com

Harding J.A., Shabaz M., Srini S. and Kusiak A. (2006). Data Mining in Manufacturing: A Review, **Journal of Manufacturing Science and Engineering**, 128(4), 969-976.

Haupt R.L. and Haupt S.E. (1998). **Practical genetic algorithms**, (2ed ed). United States of America: John Wiley & Sons, Inc.

Hwei J.L, Fu W.Y and Yang T.K (2005). An Efficient GA-based Clustering Technique, **Journal of Science and Engineering**, 8(2), 113-122.

Ian H.W. and Eibe F. (2005). **Data Mining: practical machine learning tools and techniques**, (2ed ed). United States of America: Morgan Kaufmann.

Jain A.K., Murty M.N. and Flynn P.J. (1999), Data Clustering: A Review, **ACM Computing Surveys**, 31(3), 264-323.

Kusiak A. (2006), Data Mining In Design Of Products And Production System, **INCOM'2006: 12th IFAC/IFIP/IFORS/IEEE Symposium on Control Problems in Manufaturing**, Vol 1, Saint-Etienne, France, 49-53.

Lazarevic A., Xiaowei Xu, Fiez T. and Obradovic, Z. (1999), Clustering-Regression-Ordering Steps for Knowledge Discovery in Spatial Databases, **International Joint Conference on Neural Networks**,Vol 4, Washington, DC, USA, 2530-2534.

Larose D.T. (2005), **Discovering Knowledge in Data,** (1st Ed), Hoboken, New Jersey: John Wiley & Sons.

www.manaraa.com

Liu D.R. and Shih Y.Y. (2005), Integrating AHP and data mining for product recommendation based on customer lifetime value, **Information & Management**, 42(3), 387-400.

Razib M.O., Safaai D., Rosli M.I., Zalmiyah Z. and Saberi M.M. (2006). Automatic Clustering of Gene Ontology by Genetic Algorithm, **International Journal of Information Technology**, 3(1), 37-46.

Tian J. (2002), Better Reliability Assessment and Prediction Through Data Clustering, **IEEE Transactions on Software and Engineering**, 28(10), 997-1007.

West C., MacDonalnd S., Lingras P. and Adams G. (2005), Relationship between Product Based Loyalty and Clustering based on Supermarket Visit and Spending Patterns, **International Journal of Computer Science and Applications**, 2(2), 85-100.

Weka Machine Learning Project, University of Waikato, from **http://www.cs.waikato.ac.nz**.

www.manaraa.com

Appendices

APPENDIX

A.1. Simple Data

1. Hypothetical data.

2. Attributes: X, Y.

3. Data

1,1 – 1,2 – 5,6 – 8,2 – 7,0

A.2. Iris Data

1. Title: Iris Plants Database

2. Sources:

(a) Creator: R.A. Fisher

(b) Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)

(c) Date: July, 1988

3. Past Usage:

- Publications: too many to mention!!! Here are a few.

1. Fisher,R.A. "The use of multiple measurements in taxonomic problems"

Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to Mathematical Statistics" (John Wiley, NY, 1950).

2. Duda,R.O., & Hart,P.E. (1973) Pattern Classification and Scene Analysis.

(Q327.D83) John Wiley & Sons. ISBN 0-471-22361-1. See page 218.

3. Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New System

Structure and Classification Rule for Recognition in Partially Exposed

Environments". IEEE Transactions on Pattern Analysis and Machine

Intelligence, Vol. PAMI-2, No. 1, 67-71.

www.manaraa.com

6. Number of Attributes: 4 numeric, predictive attributes and the class

7. Attribute Information:

sepal length in cm, sepal width in cm, petal length in cm, petal width in cm, class: Iris Setosa, Iris Versicolour, Iris Virginica

8. Missing Attribute Values: None

9. Data

5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
5.4,3.9,1.3,0.4,Iris-setosa
5.1,3.5,1.4,0.3,Iris-setosa
5.7,3.8,1.7,0.3,Iris-setosa
5.1,3.8,1.5,0.3,Iris-setosa
5.4,3.4,1.7,0.2,Iris-setosa
5.1,3.7,1.5,0.4,Iris-setosa
4.6,3.6,1.0,0.2,Iris-setosa
5.1,3.3,1.7,0.5,Iris-setosa
4.8,3.4,1.9,0.2,Iris-setosa
5.0,3.0,1.6,0.2,Iris-setosa
5.0,3.4,1.6,0.4,Iris-setosa
5.2,3.5,1.5,0.2,Iris-setosa
5.2,3.4,1.4,0.2,Iris-setosa
4.7,3.2,1.6,0.2,Iris-setosa
4.8,3.1,1.6,0.2,Iris-setosa
5.4,3.4,1.5,0.4,Iris-setosa
5.2,4.1,1.5,0.1,Iris-setosa
5.5,4.2,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.0,3.2,1.2,0.2,Iris-setosa
5.5,3.5,1.3,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
4.4,3.0,1.3,0.2,Iris-setosa
5.1,3.4,1.5,0.2,Iris-setosa
5.0,3.5,1.3,0.3,Iris-setosa
4.5,2.3,1.3,0.3,Iris-setosa
4.4,3.2,1.3,0.2,Iris-setosa
5.0,3.5,1.6,0.6,Iris-setosa
5.1,3.8,1.9,0.4,Iris-setosa
4.8,3.0,1.4,0.3,Iris-setosa
5.1,3.8,1.6,0.2,Iris-setosa
4.6,3.2,1.4,0.2,Iris-setosa
5.3,3.7,1.5,0.2,Iris-setosa
5.0,3.3,1.4,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
5.5,2.3,4.0,1.3,Iris-versicolor
6.5,2.8,4.6,1.5,Iris-versicolor
5.7,2.8,4.5,1.3,Iris-versicolor

www.manaraa.com

5.8,2.7,3.9,1.2,Iris-versicolor

6.3,3.3,4.7,1.6,Iris-versicolor

6.0,2.7,5.1,1.6,Iris-versicolor

4.9,2.4,3.3,1.0,Iris-versicolor

5.4,3.0,4.5,1.5,Iris-versicolor

6.6,2.9,4.6,1.3,Iris-versicolor

6.0,3.4,4.5,1.6,Iris-versicolor

5.2,2.7,3.9,1.4,Iris-versicolor

6.7,3.1,4.7,1.5,Iris-versicolor

5.0,2.0,3.5,1.0,Iris-versicolor

6.3,2.3,4.4,1.3,Iris-versicolor

5.9,3.0,4.2,1.5,Iris-versicolor

5.6,3.0,4.1,1.3,Iris-versicolor

6.0,2.2,4.0,1.0,Iris-versicolor

5.5,2.5,4.0,1.3,Iris-versicolor

6.1,2.9,4.7,1.4,Iris-versicolor

5.5,2.6,4.4,1.2,Iris-versicolor

5.6,2.9,3.6,1.3,Iris-versicolor

6.1,3.0,4.6,1.4,Iris-versicolor

6.7,3.1,4.4,1.4,Iris-versicolor

5.8,2.6,4.0,1.2,Iris-versicolor

5.6,3.0,4.5,1.5,Iris-versicolor

5.0,2.3,3.3,1.0,Iris-versicolor

5.8,2.7,4.1,1.0,Iris-versicolor

5.6,2.7,4.2,1.3,Iris-versicolor

6.2,2.2,4.5,1.5,Iris-versicolor

5.7,3.0,4.2,1.2,Iris-versicolor

5.6,2.5,3.9,1.1,Iris-versicolor

5.7,2.9,4.2,1.3,Iris-versicolor

5.9,3.2,4.8,1.8,Iris-versicolor

6.2,2.9,4.3,1.3,Iris-versicolor

6.1,2.8,4.0,1.3,Iris-versicolor

5.1,2.5,3.0,1.1,Iris-versicolor

6.3,2.5,4.9,1.5,Iris-versicolor

5.7,2.8,4.1,1.3,Iris-versicolor

6.1,2.8,4.7,1.2,Iris-versicolor

6.3,3.3,6.0,2.5,Iris-virginica

6.4,2.9,4.3,1.3,Iris-versicolor

5.8,2.7,5.1,1.9,Iris-virginica

6.6,3.0,4.4,1.4,Iris-versicolor

7.1,3.0,5.9,2.1,Iris-virginica

6.8,2.8,4.8,1.4,Iris-versicolor

6.3,2.9,5.6,1.8,Iris-virginica

6.7,3.0,5.0,1.7,Iris-versicolor

6.5,3.0,5.8,2.2,Iris-virginica

6.0,2.9,4.5,1.5,Iris-versicolor

7.6,3.0,6.6,2.1,Iris-virginica

5.7,2.6,3.5,1.0,Iris-versicolor

4.9,2.5,4.5,1.7,Iris-virginica

5.5,2.4,3.8,1.1,Iris-versicolor

5.5,2.4,3.7,1.0,Iris-versicolor

المنارة للاستشارات
www.manaraa.com

5.6,2.8,4.9,2.0,Iris-virginica
6.3,3.4,5.6,2.4,Iris-virginica
7.3,2.9,6.3,1.8,Iris-virginica
7.7,2.8,6.7,2.0,Iris-virginica
6.4,3.1,5.5,1.8,Iris-virginica
6.7,2.5,5.8,1.8,Iris-virginica
6.3,2.7,4.9,1.8,Iris-virginica
6.0,3.0,4.8,1.8,Iris-virginica
7.2,3.6,6.1,2.5,Iris-virginica
6.7,3.3,5.7,2.1,Iris-virginica
6.9,3.1,5.4,2.1,Iris-virginica
6.5,3.2,5.1,2.0,Iris-virginica
7.2,3.2,6.0,1.8,Iris-virginica
6.7,3.1,5.6,2.4,Iris-virginica
6.4,2.7,5.3,1.9,Iris-virginica
6.2,2.8,4.8,1.8,Iris-virginica
6.9,3.1,5.1,2.3,Iris-virginica
6.8,3.0,5.5,2.1,Iris-virginica
6.1,3.0,4.9,1.8,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
5.7,2.5,5.0,2.0,Iris-virginica
6.4,2.8,5.6,2.1,Iris-virginica
6.8,3.2,5.9,2.3,Iris-virginica
5.8,2.8,5.1,2.4,Iris-virginica
7.2,3.0,5.8,1.6,Iris-virginica
6.7,3.3,5.7,2.5,Iris-virginica
6.4,3.2,5.3,2.3,Iris-virginica
7.4,2.8,6.1,1.9,Iris-virginica
6.7,3.0,5.2,2.3,Iris-virginica
6.5,3.0,5.5,1.8,Iris-virginica
7.9,3.8,6.4,2.0,Iris-virginica
6.3,2.5,5.0,1.9,Iris-virginica
7.7,3.8,6.7,2.2,Iris-virginica
6.4,2.8,5.6,2.2,Iris-virginica
6.5,3.0,5.2,2.0,Iris-virginica
7.7,2.6,6.9,2.3,Iris-virginica
6.3,2.8,5.1,1.5,Iris-virginica
6.2,3.4,5.4,2.3,Iris-virginica
6.0,2.2,5.0,1.5,Iris-virginica
6.1,2.6,5.6,1.4,Iris-virginica
5.9,3.0,5.1,1.8,Iris-virginica
6.9,3.2,5.7,2.3,Iris-virginica
7.7,3.0,6.1,2.3,Iris-virginica

A.3. CPU Data

As used by Kilpatrick, D. & Cameron-Jones, M. (1998). Numeric prediction using instance-based learning with encoding length selection. In Progress in Connectionist-Based Information Systems. Singapore: Springer-Verlag.

1. attribute MYCT real, attribute MMIN real, attribute MMAX real, attribute CACH real, attribute CHMIN real , attribute CHMAX real, attribute class real.

2. Data

125,256,6000,256,16,128,199
26,8000,32000,64,8,32,290
23,32000,64000,128,32,64,1238
29,8000,32000,32,8,32,253
23,16000,32000,64,16,32,381
400,1000,3000,0,1,2,23
29,8000,32000,32,8,32,253
23,16000,32000,64,16,32,381
400,512,3500,4,1,6,24
29,8000,32000,32,8,32,253
23,16000,64000,64,16,32,749
29,8000,16000,32,8,16,132

www.manaraa.com

50,2000,8000,8,1,5,44
75,2000,16000,128,1,38,157
60,2000,8000,65,1,8,70
50,1000,4000,8,3,5,30
90,256,1000,0,3,10,18
50,4000,16000,65,1,8,117
50,1000,8000,8,3,5,41
105,256,2000,0,3,10,20
350,64,64,0,1,4,15
50,2000,16000,8,3,5,74
105,1000,4000,0,3,24,28
200,512,16000,0,4,32,64
50,2000,16000,8,3,6,74
105,2000,4000,8,3,19,33
167,524,2000,8,4,15,23
50,2000,16000,8,3,6,74
75,2000,8000,8,3,24,47
143,512,5000,0,7,32,29
133,1000,12000,9,3,12,54
75,3000,8000,8,3,48,54
143,1000,2000,0,5,16,22
133,1000,8000,9,3,12,41
175,256,2000,0,3,24,20
110,5000,5000,142,8,64,124
810,512,512,8,1,1,18
300,768,3000,0,6,24,23
143,1500,6300,0,5,32,35
810,1000,5000,0,1,1,28
300,768,3000,6,6,24,25
143,3100,6200,0,5,20,39
320,512,8000,4,1,5,36
300,768,12000,6,6,24,52
143,2300,6200,0,6,64,40
200,512,8000,8,1,8,38
300,768,4500,0,1,24,27
110,3100,6200,0,6,64,45
700,384,8000,0,1,1,34
300,384,12000,6,1,24,50
320,128,6000,0,1,12,28
700,256,2000,0,1,1,19
300,192,768,6,6,24,18
320,512,2000,4,1,3,21
140,1000,16000,16,1,3,72
180,768,12000,6,1,31,53
320,256,6000,0,1,6,28
200,1000,8000,0,1,2,36
330,1000,3000,0,2,4,23
320,256,3000,4,1,3,22
110,1000,4000,16,1,2,30
300,1000,4000,8,3,64,30
320,512,5000,4,1,5,28
110,1000,12000,16,1,2,56
300,1000,16000,8,2,112,73
320,256,5000,4,1,6,27
220,1000,8000,16,1,2,42
330,1000,2000,0,1,2,20
25,1310,2620,131,12,24,102
800,256,8000,0,1,4,34
330,1000,4000,0,3,6,25
25,1310,2620,131,12,24,102
800,256,8000,0,1,4,34
140,2000,4000,0,3,6,28
50,2620,10480,30,12,24,74
800,256,8000,0,1,4,34
140,2000,4000,0,4,8,29
50,2620,10480,30,12,24,74
800,256,8000,0,1,4,34
140,2000,4000,8,1,20,32
56,5240,20970,30,12,24,138
800,256,8000,0,1,4,34
140,2000,32000,32,1,20,175
64,5240,20970,30,12,24,136
125,512,1000,0,8,20,19
140,2000,8000,32,1,54,57
50,500,2000,8,1,4,23
75,2000,8000,64,1,38,75
140,2000,32000,32,1,54,181
50,1000,4000,8,1,5,29
75,2000,16000,64,1,38,113

www.manaraa.com

225,1000,4000,2,3,6,26

140,2000,32000,32,1,54,181

140,2000,4000,8,1,20,32

57,4000,16000,1,6,12,82

57,4000,24000,64,12,16,171

26,16000,32000,64,16,24,361

26,16000,32000,64,8,24,350

26,8000,32000,0,8,24,220

26,8000,16000,0,8,16,113

480,96,512,0,1,1,15

203,1000,2000,0,1,5,21

115,512,6000,16,1,6,35

1100,512,1500,0,1,1,18

1100,768,2000,0,1,1,20

600,768,2000,0,1,1,20

400,2000,4000,0,1,1,28

400,4000,8000,0,1,1,45

900,1000,1000,0,1,2,18

900,512,1000,0,1,2,17

900,1000,4000,4,1,2,26

900,1000,4000,8,1,2,28

900,2000,4000,0,3,6,28

225,2000,4000,8,3,6,31

225,2000,4000,8,3,6,31

180,2000,8000,8,1,6,42

185,2000,16000,16,1,6,76

180,2000,16000,16,1,6,76

25,2000,12000,8,1,4,59

25,2000,12000,16,3,5,65

17,4000,16000,8,6,12,101

17,4000,16000,32,6,12,116

1500,768,1000,0,0,0,18

1500,768,2000,0,0,0,20

800,768,2000,0,0,0,20

50,2000,4000,0,3,6,30

50,2000,8000,8,3,6,44

50,2000,8000,8,1,6,44

50,2000,16000,24,1,6,82

50,2000,16000,24,1,6,82

50,8000,16000,48,1,10,128

100,1000,8000,0,2,6,37

100,1000,8000,24,2,6,46

100,1000,8000,24,3,6,46

50,2000,16000,12,3,16,80

50,2000,16000,24,6,16,88

50,2000,16000,24,6,16,88

150,512,4000,0,8,128,33

115,2000,8000,16,1,3,46

115,2000,4000,2,1,5,29

92,2000,8000,32,1,6,53

92,2000,8000,32,1,6,53

92,2000,8000,4,1,6,41

75,4000,16000,16,1,6,86

60,4000,16000,32,1,6,95

60,2000,16000,64,5,8,107

60,4000,16000,64,5,8,117

50,4000,16000,64,5,10,119

72,4000,16000,64,8,16,120

72,2000,8000,16,6,8,48

40,8000,16000,32,8,16,126

40,8000,32000,64,8,24,266

35,8000,32000,64,8,24,270

38,16000,32000,128,16,32,426

48,4000,24000,32,8,24,151

38,8000,32000,64,8,24,267

30,16000,32000,256,16,24,603

112,1000,1000,0,1,4,19

84,1000,2000,0,1,6,21

56,1000,4000,0,1,6,26

56,2000,6000,0,1,8,35

56,2000,8000,0,1,8,41

56,4000,8000,0,1,8,47

56,4000,12000,0,1,8,62

56,4000,16000,0,1,8,78

38,4000,8000,32,16,32,80

38,4000,8000,32,16,32,80

38,8000,16000,64,4,8,142

38,8000,24000,160,4,8,281

38,4000,16000,128,16,32,190

www.manaraa.com

105,2000,4000,8,3,8,31

200,1000,2000,0,1,2,21

105,2000,6000,16,6,16,41

200,1000,4000,0,1,4,25

105,2000,8000,16,4,14,47

200,2000,8000,64,1,5,67

52,4000,16000,32,4,12,99

250,512,4000,0,1,7,24

70,4000,12000,8,6,8,67

250,512,4000,0,4,7,24

59,4000,12000,32,6,12,81

250,1000,16000,1,1,8,64

59,8000,16000,64,12,24,149

160,512,4000,2,1,5,25

26,8000,24000,32,8,16,183

160,512,2000,2,3,8,20

26,8000,32000,64,12,16,275

160,1000,4000,8,1,14,29

26,8000,32000,128,24,32,382

160,1000,8000,16,1,14,43

116,2000,8000,32,5,28,56

160,2000,8000,32,1,13,53

50,2000,32000,24,6,26,182

240,512,1000,8,1,3,19

50,2000,32000,48,26,52,227

240,512,2000,8,1,5,22

50,2000,32000,112,52,104,341

50,4000,32000,112,52,104,360

30,8000,64000,96,12,176,919

30,8000,64000,128,12,176,978

180,262,4000,0,1,3,24

180,512,4000,0,1,3,24

180,262,4000,0,1,3,24

180,512,4000,0,1,3,24

124,1000,8000,0,1,8,37

98,1000,8000,32,2,8,50

125,2000,8000,0,2,14,41

480,512,8000,32,0,0,47

480,1000,4000,0,0,0,25

www.manaraa.com

A.4. Program

1- GA.java

```
public class GA {

   public static int[][] Crossover(int P1[],int P2[])

   {

      FileReadWrite N = new FileReadWrite();

      int C1[]=new int [N.NumberOfData];

      int C2[]=new int [N.NumberOfData];

      int C[][]=new int [2][N.NumberOfData];

      for (int i=0 ; i<N.NumberOfData ; i++)

      {

         C1[i] = P1[i];

         C2[i] = P2[i];

      }

    int   StartPoint = 0;

    int   CrossoverLength;

    if (Math.random() >= 0.1) CrossoverLength = (int)(N.NumberOfData/2);

    else CrossoverLength = 0;

     for (int j=StartPoint ; j<(StartPoint+CrossoverLength) ; j++)

     {

     C1[j] = P2[j];

     C2[j] = P1[j];

     C[0]=C1;

     C[1]=C2;

     }
```


www.manaraa.com

```
 return C;
}

public static void Mutation(int C[])
{
 FileReadWrite N = new FileReadWrite();
 InitialSolution I = new InitialSolution();
 int NC = 0;
 for (int i=0 ; i<N.NumberOfData ; i++) if (C[i] == 1) NC++;
 double Mutate = Math.random();
 if (Mutate >= 0.8)
 {
    int MutationPoint;
    if (NC == 2 )
    while (true)
    {
     MutationPoint = (int)(N.NumberOfData*Math.random());
     if (C[MutationPoint] == 0) break;
    }
    else
       if (NC == I.MaxCluster )
       while (true)
       {
        MutationPoint = (int)(N.NumberOfData*Math.random());
        if (C[MutationPoint] == 1) break;
       }
```


www.manaraa.com

```
        else MutationPoint = (int)(N.NumberOfData*Math.random());
  C[MutationPoint] = Math.abs(C[MutationPoint]-1);
   }
}
public static double [][] InitiateSelectionMatrix(double Fitness[])
{
    InitialSolution I =new InitialSolution();
    MainProgram M = new MainProgram();
    double SelectionMatrix [][]= new double [Fitness.length][2];
    SelectionMatrix[0][0]=0;
    SelectionMatrix[0][1]=Fitness[0];
    for (int i=1;i<Fitness.length;i++)
    {
    SelectionMatrix[i][0]=SelectionMatrix[i-1][1];
    SelectionMatrix[i][1]=SelectionMatrix[i][0]+Fitness[i];
    }
    return SelectionMatrix;
}
public static int[] Select(double SelectionMatrix[][])
{
    int P[]=new int [2];
    InitialSolution I =new InitialSolution();
    double R1 = Math.random();
    double R2 = Math.random();
    for (int i=0;i<SelectionMatrix.length;i++)
    {
```


www.manaraa.com

```
        if (R1>=SelectionMatrix[i][0] && R1<=SelectionMatrix[i][1])
        {
           P[0]=i+I.NumberOfSolutions;
           break;
        }
     }
     for (int i=0;i<I.NumberOfSolutions;i++)
     {
        if (R2>=SelectionMatrix[i][0] && R2<=SelectionMatrix[i][1])
        {
           P[1]=i+I.NumberOfSolutions;
           break;
        }
     }
     return P;
}
public static double [] Fitness(double InterIntraDistance[][],int Dominated[])
{
   double Fitness[]=new double [InterIntraDistance.length];
   double Distance =0;
   double min;
   double D=(InterIntraDistance.length);

    for (int i=0;i<InterIntraDistance.length;i++)
    {
      min=-1;
```


www.manaraa.com

```
if (Dominated[i] == 0)
{
                    for (int j=0;j<InterIntraDistance.length;j++)
  {
                        if (Dominated[j]==1)
                {
                    Distance =Math.sqrt(Math.pow((InterIntraDistance[j][0]-
InterIntraDistance[i][0]),2)+Math.pow((InterIntraDistance[j][1]-InterIntraDistance[i][1]),2));
                    if (Distance < min || min == -1) min = Distance;
                }
            }
                    Fitness[i] = min;
  }
  else
     Fitness[i]=0;
 }
double sum=0;
 for (int i=0;i<InterIntraDistance.length;i++)
    sum=sum+Fitness[i];
if (sum==0)
{
   for (int i=0;i<InterIntraDistance.length;i++)
      Fitness[i]=1;

   sum=D;
}
```


www.manaraa.com

```
    for (int i=0;i<InterIntraDistance.length;i++)
       for (int j=0;j<i;j++)
          if (Dominated[i] == -1)
          {
           Fitness[i]=sum;
          }
    for (int i=0;i<InterIntraDistance.length;i++)
       Fitness[i]=sum-Fitness[i];
     sum=0;
     for (int i=0;i<InterIntraDistance.length;i++)
        sum=sum+Fitness[i];
        for (int i=0;i<InterIntraDistance.length;i++)
           Fitness[i]=Fitness[i]/sum;
        for (int i=0;i<InterIntraDistance.length;i++)
          if (InterIntraDistance[i][1]==0)Fitness[i]=-1;
    return Fitness;
  }}
```

2- InitialSolution.java

```
public class InitialSolution {
   public static int set[][];
   public static int NumberOfSolutions;
   public static int MaxCluster;
   public static int NOCC=1;
     public static void initiate ()
   {
```


www.manaraa.com

```
    FileReadWrite N = new FileReadWrite();
    NumberOfSolutions = (int) (4 * Math.sqrt(N.NumberOfData * MaxCluster));
    set = new int [NumberOfSolutions][N.NumberOfData];
    for (int i=0 ; i<NumberOfSolutions ; i++)
       for (int j=0 ; j<N.NumberOfData ; j++)
            set[i][j] =0;
     for (int i=0 ; i<NumberOfSolutions ; i++)
     {
       if (NOCC == MaxCluster) NOCC=1;
       NOCC++;
       int counter =0;
while (counter<NOCC){
       int l=(int)((N.NumberOfData-1)*Math.random());
        if (set[i][l]==0)
        {
            counter++;
            set[i][l] =1;
        }
       for (int c=0;c<i;c++)
          if (set[c]==set[i]) i--;
}3- Kmeans.java
public class Kmeans {
   public static double ClusterCenter [][];
   public static double CovMatrix[][];
   public static double CovInv[][];
```


www.manaraa.com

```
 public static int NumberOfDominated;
public static void CovMatrix()
{
   FileReadWrite N = new FileReadWrite();
   CovMatrix = new double [N.NumberOfFeatures][N.NumberOfFeatures];
   MainApplet M = new MainApplet();
   double Mean[] = new double[N.NumberOfFeatures];
   double XY[][] = new double [N.NumberOfFeatures][N.NumberOfFeatures];
   if (M.EuclideanMahalanobis==0)
    for(int i=0;i<N.NumberOfFeatures;i++)
        for(int j=0;j<N.NumberOfFeatures;j++)
            if (i==j) CovMatrix[i][j]=1;
            else CovMatrix[i][j]=0;
   if (M.EuclideanMahalanobis==1)
   {
      for (int i=0 ; i< N.NumberOfFeatures ; i++)
   {
        for (int j=0 ; j< N.NumberOfData ; j++)
            Mean[i] = Mean[i] + N.Data[j][i];
        Mean[i] = Mean[i]/N.NumberOfData;
   }
    for (int i=0 ; i< N.NumberOfFeatures ; i++)
      for (int j=0 ; j< N.NumberOfFeatures ; j++)
      {
```


www.manaraa.com

```
            for (int l=0 ; l<N.NumberOfData ; l++)
               CovMatrix[i][j]=CovMatrix[i][j]+(N.Data[l][i]-Mean[i])*(N.Data[l][j]-Mean[j]);
            CovMatrix[i][j] = CovMatrix[i][j]/(N.NumberOfData-1);
         }
      }
   }
   public static void Inv()
   {
         FileReadWrite N = new FileReadWrite();
      int dimention = N.NumberOfFeatures ;
      CovInv = new double [dimention][dimention];
         double A[][]=new double[dimention] [2*dimention];
   for(int i=0;i<dimention;i++){
   for(int j=0;j<2*dimention;j++){
   if (j>=dimention){
   if (i==j-dimention) A[i][j]=1;
   else A[i][j]=0;}
   else
   A[i][j]=CovMatrix[i][j];
}
   for(int i=0;i<dimention;i++){
                  if (A[i][i]==0) {
                        JOptionPane.showMessageDialog(null,"Singular Matrix");
                        System.exit(0);
                  }
   double temp=A[i][i];
```


www.manaraa.com

```
            for(int j=0;j<2*dimention;j++) A[i][j]=A[i][j]/temp;
    for(int k=0;k<dimention;k++){
                        if (k!=i){
                                double temp2=A[k][i];
                        for(int l=0;l<2*dimention;l++)A[k][l]=-A[i][l]*temp2+A[k][l];
                        }
}
            }
    for(int i=0;i<dimention;i++)
    for(int j=dimention;j<2*dimention;j++) CovInv[i][j-dimention] = Math.floor( A[i][j] * 100 + .5 ) / 100;


}
    public static double Distance(double X1[],double X2[])
    {
    FileReadWrite N = new FileReadWrite();
    double d=0;
    double temp[] = new double [N.NumberOfFeatures];
    double X1X2[] = new double [N.NumberOfFeatures];
    for(int i=0;i<N.NumberOfFeatures;i++)
       X1X2[i] = X1[i]-X2[i];
    for(int i=0;i<N.NumberOfFeatures;i++)
       for(int j=0;j<N.NumberOfFeatures;j++)
          temp[i]=temp[i]+X1X2[j]*CovInv[j][i];
    for(int i=0;i<N.NumberOfFeatures;i++)
       d=d+temp[i]*X1X2[i];
    return d;
    }
```


www.manaraa.com

```
public static int [] Kmeans (int X[])
{
 FileReadWrite N = new FileReadWrite();
 double MinDistance;
 int Clustered[]=new int [N.NumberOfData];
 int ClusterNumber=0;
 for(int i=0;i<N.NumberOfData;i++)
    if (X[i]==1) ClusterNumber++;
 ClusterCenter = new double [ClusterNumber][N.NumberOfFeatures];
 int counter=0;
 for(int i=0;i<N.NumberOfData;i++)
   if (X[i]==1)
   {
    for(int j=0;j<N.NumberOfFeatures;j++)
        ClusterCenter [counter][j]=N.Data[i][j];
    counter++;
   }
 int ID;
 for(int i=0;i<N.NumberOfData;i++)
 {
  MinDistance = Distance(N.Data[i],ClusterCenter[0]);
  ID=1;
  for(int j=0;j<ClusterNumber;j++)
  if (Distance(N.Data[i],ClusterCenter[j])<=MinDistance)
```


www.manaraa.com

```
      MinDistance = Distance(N.Data[i],ClusterCenter[j]);
      ID=j+1;
   }
   Clustered[i]=ID;
  }
 int signal = 0;
 int terminate=0;
  while (terminate<100)
  {
  signal=0;
  counter =0;
  double sum[]=new double[N.NumberOfFeatures];
  for(int i=0;i<ClusterNumber;i++)
  {
     for(int l=0;l<N.NumberOfFeatures;l++)
     sum[l] = 0;
     counter=0;
     for(int j=0;j<N.NumberOfData;j++)
   {
     if (Clustered[j]==i+1)
     {
     for(int l=0;l<N.NumberOfFeatures;l++)
     sum[l]=sum[l]+N.Data[j][l];
     counter++;
     }
```


www.manaraa.com

```
 }
for(int l=0;l<N.NumberOfFeatures;l++)
   ClusterCenter[i][l]=sum[l]/counter;
}
for(int i=0;i<N.NumberOfData;i++)
{
 MinDistance = Distance(N.Data[i],ClusterCenter[0]);
 ID=1;
 for(int j=0;j<ClusterNumber;j++)
 if (Distance(N.Data[i],ClusterCenter[j])<MinDistance)
 {
    MinDistance = Distance(N.Data[i],ClusterCenter[j]);
    ID=j+1;
 }
 if (Clustered[i]!=ID) signal++;
 Clustered[i]=ID;
}
terminate++;
if (signal==0) break;
}
int I=-1;
for (int i=0;i<Clustered.length;i++)
   if (Math.abs(Clustered[i])==Clustered[i])
   {
```


www.manaraa.com

```
    int temp=Clustered[i];
    Clustered[i]=I--;
    for (int j=i+1;j<Clustered.length;j++)
      if (Clustered[j]==temp) Clustered[j]=Clustered[i];
    }
 for (int i=0;i<Clustered.length;i++)  Clustered[i]=Math.abs(Clustered[i]);
  return Clustered;
}
public static double IntraDist (int []S , double [][]CC)
{
FileReadWrite N = new FileReadWrite();
double sum=0;
for(int i=0;i<N.NumberOfData;i++)
   sum= sum + Distance(N.Data[i],CC[S[i]-1]);
return sum;
}
public static double InterDist (double [][]CC)
{
double sum=0;
for(int i=0;i<CC.length;i++)
       for(int j=0;j<CC.length;j++)
          if (i!=j) sum= sum + Distance(CC[i],CC[j]);
return sum/2;
}
```


www.manaraa.com

```
public static int [] DetermineDominated(double InterIntraDistance[][])
{
   InitialSolution I =new InitialSolution();
   MainApplet M = new MainApplet();
   FileReadWrite N = new FileReadWrite();
   QualityMeasures Q = new QualityMeasures();
   NumberOfDominated = 0;
   int Dominated [] = new int [InterIntraDistance.length];
   for(int i=0;i<InterIntraDistance.length;i++)
      Dominated[i] = 1;
   for(int i=0;i<InterIntraDistance.length;i++)
      for(int j=0;j<InterIntraDistance.length;j++){
      double J0=Math.floor( InterIntraDistance[j][0] * 1000 + .5 ) / 1000;
      double J1=Math.floor( InterIntraDistance[j][1] * 1000 + .5 ) / 1000;
      double I0=Math.floor( InterIntraDistance[i][0] * 1000 + .5 ) / 1000;
      double I1=Math.floor( InterIntraDistance[i][1] * 1000 + .5 ) / 1000;
      if ((J0 < I0 && J1 > I1) || (J0 == I0 && J1 > I1) || (J0 < I0 && J1 == I1))
        {
          Dominated[i] = 0;
        }
```


www.manaraa.com

```
    for(int i=0;i<InterIntraDistance.length;i++)
        for(int j=0;j<i;j++)
          if (InterIntraDistance[j][0] == InterIntraDistance[i][0] && InterIntraDistance[j][1] == InterIntraDistance[i][1])
            {
             Dominated[i] = -1;
            }
    return Dominated;
  }
 public static double [] Mean(double InterIntraDistance[][],int D[],double F[])
  {
    double I[]=new double [3];
    double sum1=0,sum2=0;
    int counter=0;
    for (int i=0;i<D.length;i++)
      if (D[i] ==1)
      {
      sum1=sum1+InterIntraDistance[i][0];
      sum2=sum2+InterIntraDistance[i][1];
      counter++;
      }
    I[0]=sum1/counter;
    I[1]=sum2/counter;
    I[2]=counter;
    return I;
```


www.manaraa.com

4-MainApplet.java

```
public class MainApplet extends javax.swing.JApplet {
    private ParetoGraph pareto = new ParetoGraph();
    private Graph2D Graph2D = new Graph2D();
    public void init() {
                initComponents();
    }
    private void initComponents() {
   }
    public static double InterIntraDistance[][];
    public static int TempDominated [];
    public static int Dominated [];
    public static double TempInterIntraDistance[][];
    public static double SelectionMatrix[][];
    public static double TempSelectionMatrix[][];
    public static int TempSet[][];
    public static int ClusteredData[][];
    public static int TempClusteredData[][];
    public static double Fitness [];
    public static double TempFitness [];
    public static double Improvement[][];
    public static int P[]=new int[2];
    public static int EuclideanMahalanobis;
    public static int FinalClustered [][];
```


www.manaraa.com

```
    Calendar StartCal = Calendar.getInstance();
    Calendar EndCal = Calendar.getInstance();
    public static double time;
    public static TimeZone tz = TimeZone.getTimeZone("CST");
    public static int GAcounter=0;
    public String file="";
    public int SavedOrSolved;
    private String SS;
    private String headings [] = new String [] {"Solution No.","k","Homogeneity","Heterogeneity","Silhouette"};
    private Object[][] data;
    private double FillData[][];
    private FileReadWrite F;
    private InitialSolution I;
    private Kmeans K;
    private GA G;
    private QualityMeasures Q;
    private Statistics St;
    private Graph2D graph;
    private void OpenFile(){
        F = new FileReadWrite();
        I = new InitialSolution();
        K = new Kmeans();
        G = new GA();
        Q = new QualityMeasures();
```


www.manaraa.com

```
    St = new Statistics();
    try{
       F.NumberOfFeatures = Integer.parseInt(this.jTextField2.getText());
       if (F.NumberOfFeatures<=0){
          JOptionPane.showMessageDialog(null, "Enter a valid number of features");
          return;
       }
    } catch(Exception e){
       JOptionPane.showMessageDialog(null, "Enter the number of features");
       return;
    }
    F.OpenFile();
    F.ReadFile();
    this.jButton2.setEnabled(true);
    this.jMenuItem5.setEnabled(true);
  }
public void Run() {
    try{
       start();
       SavedOrSolved=0;
        if (this.jRadioButton1.isSelected()) EuclideanMahalanobis = 0;
       else if (this.jRadioButton2.isSelected()) EuclideanMahalanobis = 1;
       else{
          JOptionPane.showMessageDialog(null, "Enter distance type");
          return;
```


www.manaraa.com

```
        }
        if (jTextField1.getText().equals("Auto") || jTextField1.getText().equals("auto"))
        {
          if (F.NumberOfData * F.NumberOfFeatures > 500)I.MaxCluster = (int)
(Math.ceil(Math.sqrt(F.NumberOfData)));
          else I.MaxCluster = (int)(F.NumberOfData/2);
          if (I.MaxCluster<3) I.MaxCluster = 3;
        }
        else
        {I.MaxCluster = Integer.parseInt(this.jTextField1.getText());
        if (I.MaxCluster>=F.NumberOfData || I.MaxCluster<3){
          JOptionPane.showMessageDialog(null, "Enter a valid number of clusters");
          return;
        }
        }
      } catch(Exception e){
        JOptionPane.showMessageDialog(null, "Enter the maximum number of clusters");
        return;
      }
      I.initiate();
      SelectionMatrix= new double[I.NumberOfSolutions][2];
      TempSelectionMatrix= new double[2*I.NumberOfSolutions][2];
      TempSet = new int[2*I.NumberOfSolutions][F.NumberOfData];
      TempInterIntraDistance =new double [2*I.NumberOfSolutions][2];
      ClusteredData = new int [I.NumberOfSolutions][F.NumberOfData];
      TempClusteredData = new int [2*I.NumberOfSolutions][F.NumberOfData];
```


www.manaraa.com

```
InterIntraDistance = new double [I.NumberOfSolutions][2];
Fitness = new double [I.NumberOfSolutions];
TempFitness = new double [2*I.NumberOfSolutions];
Dominated =new int [I.NumberOfSolutions];
TempDominated =new int [2*I.NumberOfSolutions];
Improvement = new double [1000][3];
K.CovMatrix();
K.Inv();
   for (int i=0 ; i < I.NumberOfSolutions ; i++) {
   ClusteredData[i] = K.Kmeans(I.set[i]);
   InterIntraDistance[i] = Q.HomogeneityHeterogeneity(ClusteredData[i]);
}
Dominated = K.DetermineDominated(InterIntraDistance);
int signal=0;
while (GAcounter<999) {
   GAcounter++;
   Fitness = G.Fitness(InterIntraDistance, Dominated);
   Improvement[GAcounter]=K.Mean(InterIntraDistance, Dominated, Fitness);
   double i1=Math.abs(Improvement[GAcounter][0]-Improvement[GAcounter-1][0]);
   double i2=Math.abs(Improvement[GAcounter][1]-Improvement[GAcounter-1][1]);
   double i3=Math.abs(Improvement[GAcounter][2]-Improvement[GAcounter-1][2]);
   double j1=Improvement[GAcounter][0]/100;
   double j2=Improvement[GAcounter][1]/100;
   double j3=I.NumberOfSolutions/50;
```


www.manaraa.com

```
boolean test = (i1<j1) && (i2<j2) && (i3<j3);
if (test) {
        signal++;
        if (signal==3) break;
        } else signal=0;
SelectionMatrix = G.InitiateSelectionMatrix(Fitness);
for (int i=I.NumberOfSolutions; i < 2*I.NumberOfSolutions ; i++) {
   for (int j=0; j < 2 ; j++)
      TempInterIntraDistance[i][j]=InterIntraDistance[i-I.NumberOfSolutions][j];
   for (int j=0; j < F.NumberOfData ; j++) {
      TempSet[i][j]= I.set[i-I.NumberOfSolutions][j];
      TempClusteredData[i][j] = ClusteredData[i-I.NumberOfSolutions][j];
   }
}
int counter = 0;
while (true) {
   P=G.Select(SelectionMatrix);
   int C[][]=new int [2][F.NumberOfData];
   C=G.Crossover(TempSet[P[0]], TempSet[P[1]]);
   G.Mutation(C[0]);
   G.Mutation(C[1]);
   int counter1 = 0 , counter2 = 0;
   for(int j=0;j<F.NumberOfData;j++)
   {
```


www.manaraa.com

```
            if (C[0][j]==1) counter1++;
            if (C[1][j]==1) counter2++;
        }
        if (counter1 <= I.MaxCluster && counter1 >= 2) {
            for (int k=0;k<F.NumberOfData;k++)
            I.set[counter][k]=C[0][k];
            counter++;
            if (counter==I.NumberOfSolutions) break;
        }
        if (counter2 <= I.MaxCluster && counter2 >= 2) {
            for (int k=0;k<F.NumberOfData;k++)
            I.set[counter][k]=C[1][k];
            counter++;
            if (counter==I.NumberOfSolutions) break;
        }
    }
    for (int i=0 ; i < I.NumberOfSolutions ; i++) {
        ClusteredData[i] = K.Kmeans(I.set[i]);
        InterIntraDistance[i]=Q.HomogeneityHeterogeneity(ClusteredData[i]);
    }
    for (int i=0; i < I.NumberOfSolutions ; i++) {
        for (int j=0; j < 2 ; j++)
            TempInterIntraDistance[i][j]=InterIntraDistance[i][j];
        for (int j=0; j < F.NumberOfData ; j++) {
            TempSet[i][j]= I.set[i][j];
```


www.manaraa.com

```
            TempClusteredData[i][j] = ClusteredData[i][j];
        }
    }
    TempDominated = K.DetermineDominated(TempInterIntraDistance);
    TempFitness = G.Fitness(TempInterIntraDistance, TempDominated);
   double MaxFitness=0;
    int ID=0;
    for (int i=0 ; i < I.NumberOfSolutions ; i++) {
        MaxFitness=-1;
        ID=0;
        for (int l=0 ; l < 2*I.NumberOfSolutions ; l++)
            if (TempFitness[l]>MaxFitness) {
            MaxFitness = TempFitness[l];
            ID=l;
            }
        for (int j=0; j < F.NumberOfData ; j++) {
            I.set[i][j]=TempSet[ID][j];
            ClusteredData[i][j] = TempClusteredData[ID][j];
        }
        for (int j=0; j < 2 ; j++)
            InterIntraDistance[i][j]=TempInterIntraDistance[ID][j];
        Dominated[i]=TempDominated[ID];
        TempFitness[ID]=-1;
    }
end();
```


www.manaraa.com

تصنيف البيانات متعدد الأهداف المستند إلى وسطاء المجموعات باستخدام الخوارزمية الجينية

إعداد

سامر مصطفى الرمحي

المشرف

الدكتور ناصر حسني رحال

ملخـــص

ABSTRACT IN ARABIC

تظهر أهمية تصنيف البيانات كإحدى الطرق الفعالة لاستخراج المعلومات من البينات من خلال التطبيقات المتعدده لها في مجال الأعمال و البحث العلمي. في هذه الرساله سيتم تصنيف البيانات لأول مرة باستخدام طريقه وسطاء المجموعات عن طريق إيجاد مجموعة الحلول المثلى متعددة الأهداف مدعومة بالخوارزمية الجينية لتخطي العقبات التي تعاني منها طرق وسطاء المجموعات التقليدية ، و أهم تلك المعوقات هي الشكل الكروي للمجموعات الناتجة و المراكز الأولية الميته الناشئه عن الاختيار الخاطئ لمراكز المجموعات الأولية و إعتمادية المجموعات الناتجة على ذلك الإختيار و ضرورة الاختيار المسبق لعدد المجموعات.

لتخطي تلك العقبات سيتم قياس المسافات بين البيانات آخذين بعين الاعتبار مدى تشتت تلك البيانات كما سيتم إختيار المراكز الأولية للمجموعات من مجموعة البيانات مباشرة للحد من المراكز الميته وسيتم استعمال طريقة إيجاد مجموعة الحلول المثلى متعددة الأهداف مدعومة بالخوارزمية الجينية لتخطي ضرورة تحديد عدد المجموعات و اعتمادية المجموعات الناتجة على اختيار المراكز الأولية.

في هذه الرسالة سيتم طرح خوارزمية جديدة لحل مسألة تصنيف البيانات و سيتم كتابة برنامج إستناداً على تلك الخوارزمية. سيتم التأكد من صحة عمله ونتائجه باستخدام مجموعات بيانات قياسية. كما سيتم دراسة إحدى الحالات الصناعية واستخدام البرنامج في حل تلك المسألة.

المنارة للاستشارات

www.manaraa.com