UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

A NEW GENETIC ALGORITHM BASED CLUSTERING FOR BINARY AND IMBALANCED CLASS DATA SETS

Sabariah Saharan

A thesis submitted for the degree of Doctor of Philosophy

Abstract

This research was initially driven by the lack of clustering algorithms that specifically focus on binary data. To overcome this gap in knowledge, a promising technique for analysing this type of data became the main subject in this research, namely Genetic Algorithm (GA). This type of algorithm has an intrinsic search parallelism that avoids getting stuck at the local optima and poor initialization. For the purpose of this research, GA was combined with the Incremental K-means (IKM) algorithm to cluster the binary data streams. However, prior to this proposed method, a well-known GA based clustering method, GCUK was applied to gauge the performance of this algorithm to cluster the binary data, with new application for binary data set. Subsequently, this led to a proposed new method known as Genetic Algorithm-Incremental K-means (GAIKM) with the objective function based on a few sufficient statistics that may be easily and quickly calculated on binary numbers. Different from the other clustering algorithms for binary data, this proposed method has an advantage in terms of fast convergence by implementing the IKM. Additionally, the utilization of GA provides a continuous process of searching for the best solutions, that can escape from being trapped at the local optima like the other clustering methods. The results show that GAIKM is an efficient and effective new clustering algorithm compared to the clustering algorithms and to the IKM itself. The other main contribution in this research is the ability of the proposed GAIKM to cluster imbalanced data sets, where standard clustering algorithms cannot simply be applied to this data as they could cause misclassification results. In conclusion, the GAIKM outperformed other clustering algorithms, and paves the way for future research in missing data and outliers and also by implementing the GA multi-objective optimization.

Acknowledgement

First and foremost, I am grateful to Almighty God for giving me many great opportunities in life, including this challenging yet wonderful PhD journey.

I am deeply thankful to Professor Jennifer Brown, Professor Roberto Baragona and A/Prof Marco Reale for their helpful supervision, generous assistance and encouragement to pursue this research that led to this thesis.

I owe very much to Professor Roberto Baragona for his thoughtful advice, knowledge and wisdom. His guidance and assistance were invaluable in helping me acquire the necessary skills to become a PhD researcher.

I would like to express my great appreciation to my parents, family and friends. It was hard being away from them, but their constant love and care touched me in many ways. Without their support, this work would not have been possible. To my closest friends in Christchurch, thank you for being an instant family to us these past years.

I am very fortunate to have my loving, understanding husband, ***Mohd Haneff Soid***, and all my daughters, ***Irdina Haniss, Arissa Humaira***, and ***Sarah Syuhada*** and my only son,

Imran Wafi, by my side throughout my study. I am forever in their debt for supporting me throughout countless weekends that were supposed to be reserved especially for them. Not forgetting my mum, ***Hajah Habibah Ma'som*** for her endless and countless pray, my siblings, and all my in laws. I would not have been able to complete this journey without their love, patience and pray.

I would like to acknowledge my employer, Universiti Tun Hussein Onn Malaysia (UTHM) and the Government of Malaysia for funding my PhD program and living expenses for my family and I in Christchurch. Without these financial support I would not have been able to undertake this program.

Finally I am also grateful to many others, all my friend in Christchurch and Malaysia who are really kind and helpful throughout this journey and all of whom can not be named here, but you are always in my heart!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

The increasing volume of data collected by individuals, firms and organizations at a specific range of time has triggered the urge to explore and process this type of information. Many statistical techniques have been proposed by researchers to analyze this large volume of data such as analysis of variance, linear regression, discriminant analysis, multi-dimensional scaling, classification and clustering.

One of the most popular statistical techniques used to explore and process this increasingly large volume of information is clustering. Clustering can be referred to as an unsupervised classification whereby the data points are clustered using the only information that is available in the data set and the number of clusters,K, may be known a priori or not. This technique is deemed one of the most challenging problems in machine learning [Hruschka et al., 2009]. There are more than a thousand different scientific papers and theses published related to this topic, and until today, there is a continuous proposal of new clustering algorithms.

The aim of clustering is to divide the data points into K clusters, whereby the characteristics of the data within the cluster should be as similar as possible, and they should be different from the other data in other clusters. The higher the similarity within the cluster and the higher the difference between clusters, the better the clustering. In this study, the interest was to cluster the variables to see a similar pattern of these variables. For example given a set of n variables, $X = x_1, x_2, \ldots, x_n$ to be clustered with each of these $X_i \in R^P$ is an attribute vector used to describe the variables. These variables will be clustered into a set of clusters, $C = C_1, C_2, \ldots, C_K$ where K is the number of clusters. The clusters are disjoint $C, C_i \cap C_j = \oslash$ for $i \neq j$. The numbers of K may be known a priori or not. Let $\bar{x_k}$ be the mean of the cluster C_k. The main goal of clustering is to find the minimum distance between the x_i to the closest center $\bar{x_k}$ as follows:

$$\sum_{k=1}^{K} \sum_{x_i \in C_k} \parallel x_i - \bar{x_k} \parallel^2$$

A well-known clustering method is K-means [Macqueen, 1967] algorithm which uses an iterative process. This algorithm is easy to implement and efficient [Jain, 2010, Jain et al., 1999]. The method of K-means can be described by the following steps:

1. Choose the initial centers $(\bar{x}_1, \ldots, \bar{x}_k)$ for each of the clusters $C_1, \ldots, C_K$.
2. Find the new cluster membership by assigning each data point to the closest centres.
3. Recalculate the clusters centres.
4. Stop the process if all the cluster centres do not change. Otherwise repeat the process from Step 2.

Even though the K-means algorithm is computationally efficient, there are a few critical weaknesses. The quality of the clustering is totally dependent on the initial cluster centres and is sensitive to outliers. Moreover, it can also be easily trapped in a local minimum [Lu et al., 2004b, Ordonez, 2003, Pham et al., 2004]. The other weakness of K-means is that it is time consuming when applied to a large data set [Sanjay Chakraborty, 2011].

Incremental K-means (IKM) [Ordonez, 2003, Pham et al., 2004, Sanjay et al., 2011, Sanjay Chakraborty, 2011] was proposed to improve the performance of the standard K-means with different objectives. Chakraborty and Nagwani [2011], Sanjay et al. [2011], Sanjay Chakraborty [2011] proposed an IKM where the new data are clustered by comparing their smallest distance from the means of the existing clusters. The result is same as if a standard K-means was run for the whole data set, but this new proposed method needed less computation time. The proposed IKM showed better performance when compared with the standard K-means. However, when the threshold value or the % of δ change in the database exceeded 57%, standard K-means outperformed IKM.

Pham et al. [2004] proposed an IKM variant that applied the device of allowing a cluster center to jump to a completely different point. This device may enable the cluster centres to move to reduce a cluster distortion. When the value of K becomes larger, the search process is slowed down because the algorithm needs to find all possible beneficial places to insert the new clusters.

The IKM method proposed by Ordonez [2003] deals with large binary data sets specifically. This method may be viewed as a compromise method between the online K-means and the standard K-means algorithms. Online K-means, most commonly known as sequential K-means, allows to update the model as new data is received. However, unlike these two methods, this IKM will not iterate until convergence, and only the values of the sufficient statistics N and M have to be updated at each step. Let n be the number of cases, then $N_j, j = 1, \ldots, k$ is the number of cases that are assigned to cluster j and M_{ij} is the sum of the instances that equal to 1 recorded for the factor i for all the cases that are assigned to cluster j. The performance of IKM outperformed standard K-means, Scalable K-means and On-line K-means in most cases. However, when this algorithm was applied to very sparse matrices and very high dimensional data set, it was found that the Scalable K-means had slightly better solutions than IKM.

Another approach to improve the performance of the standard K-means is by using Genetic Algorithms. Genetic Algorithms or GA was an idea from John Holland and his friend in the early of 1970's. It is a subclass of the Evolutionary Algorithms which use stochastic searching techniques [Goldberg, 1989]. These techniques are inspired by the process of biological evolution; selection, crossover and mutation processes. The continues process for searching the best population of the problem solutions, gives GA the ability to escape from stopping at the local optima. This can make GA more efficient compared to other local optimization methods.

Some of the researchers that use GA to improve the performances of the standard K-means, are Bandyopadhyay and Maulik [2002b] (Genetic Clustering for Unknown K), Krishna et al. [1999] (Genetic K-means Algorithm), Lu et al. [2004a] (Fast Genetic K-Means Algorithm), Lu et al. [2004b] Incremental Genetic K-means Algorithm , Arunprasath et al. [2010] (Genetic Algorithm for the K-means Initialization[Kwedlo and Iwanowicz, 2010], Rapid Genetic

Algorithm), and Guo et al. [2006], Li et al. [2010], Murthy and Chowdhury [1996], Tan and Lu [2009](An Improved K-means with Combined with the Genetic Algorithm). The details for the Genetic Clustering Algorithm will be explained in Chapter 3.

Based on these few examples, the idea of using GA for K-means improvement is not new. However, many of these studies focus on quantitative data (discrete and continuous). Therefore the development of clustering for categorical data has not grown as tremendously as that for numerical data. It may be due to the definition of clustering for categorical data being not as clear as the problem for numerical data [Duda and Hart, 1974]. For numerical data, the problem of high dimensionality [Ordonez and Omiecinski, 2002], data sparsity [Guha et al., 1998] and noise [Bradley et al., 1998] make clustering for this type of data sets more challenging compared to categorical data.

Even though the idea of using GA to improve the performance of K-means has proposed by many researchers, most of them used numerical data sets as their test data. Thus, the main idea in this study, is to cluster the binary data sets by exploiting the advantages in GA to provide a good population and then cluster it by using computing devices taken from the efficient and effective IKM.

1.2 Objectives of the Study

This thesis proposes a new GA based clustering algorithm that focuses on clustering binary data. In particular, the aims of this thesis are to:

1. exploit the most characteristics properties of binary data by using the existing GA based clustering methods.

2. to improve the performance of GCUK to cluster the binary data and imbalanced class data sets.

3. compare the performance of GCUK-binary with K-means clustering algorithm.

4. propose a new clustering algorithm from a combination of GA and IKM that specifically focusing on binary data and imbalanced class data sets.

5. to compare the performance of GAIKM with the existing clustering methods, IKM, SKM, GCUK and NSGAII.

1.3 Significance of the Study

GA is a well-known optimization method that has been shown by many researchers to be capable of successfully carrying out the clustering task. Unlike the other traditional clustering algorithms, GA provides populations that are filled with only the best problem solutions. Each of the populations have a chance to be picked as the best solution. The continuous process to search for the best solution gives an advantage to GA to escape from being trapped at the local optima.

One advantage of GA, is it uses the entire space as a solution to the objective questions. GA can be used either for single objective optimization or multi objective optimizations. The well-known GCUK [Bandyopadhyay and Maulik, 2002b] is one of the examples of GA that use the single objective optimization and one of the algorithms that improved the performance of K-means. It was proven as an efficient and effective algorithm to cluster large data sets.

In spite of the effectiveness of GCUK to cluster large data sets, this algorithm has a weakness as it is uses the length of the data point as a chromosome representation. This results in increased time to converge if it involves large high dimensional data sets. It should be noted that GA must iterate over every string in the chromosome and basically the longer data point (string) the longer it takes to converge. The GCUK also has the capability to provide the number of clusters during the process. However, this advantage has a drawback as GCUK can give empty clusters. We show this in the experiment in Chapter 4 (GCUK-binary).

The objective of the version of GCUK implemented in this study was to exploit the most characteristic properties of binary data by using the well-known GA based clustering algorithm. Some new improvements were added in this algorithm to suit the binary environment as GCUK has never been used to cluster these types of data sets. Based on the findings, advantages and weaknesses from the results were used to develop a new clustering method.

The other significance output in this study was to improve the performance of K-means and Incremental K-means algorithm which was proposed by [Ordonez, 2003]. Although K-means was already known as efficient and provided an easy way to interpret the results, this algorithm has its drawbacks such as: time consuming, easily trapped at the local optima, extremely dependent on the initial values and sensitive to outliers. To overcome these problems, some researchers proposed an Incremental K-means (IKM) to enhance the performance of the standard K-means. There are varieties of IKM based on the objectives of the studies. Ordonez [2003] for example, proposed an IKM that specifically improves the performance of the standard K-means to cluster the binary data streams. His study proved that this algorithm is efficient and effective when compared with the Scalable K-means and On-line K-means. However, in some cases, where the data sets have some issues like sparsity and high dimensionality, the Scalable K-means slightly outperformed the IKM.

Based on this weakness, this study used the advantages and the efficiency of the IKM from the idea of Ordonez [2003] and combine with the GA. Single objective optimization was chosen as from the findings and results, there are some gaps from this method that can be improve in this study.

The method used in this study has been used by other researchers before, and many of them have proposed a new method based on the improvement from the original method. Nevertheless, this study is still important and contributes something new. Apart from the different types of data sets that were used, the other contribution is the application to imbalanced class data sets in this study which have never been tested with GCUK. The findings and results from this study can provide a new reference to future research involving GA and imbalanced class data.

1.4 Scope and Limitation of the Study

There are many efficient GA clustering algorithms that have been proposed by researchers, but in this study a well-known existing GA based clustering algorithms was applied; Genetic Clustering for Unknown K (GCUK) proposed by Bandyopadhyay and Maulik [2002b]. This algorithm has been used for numerical data sets and has never been applied to binary data or imbalanced class data sets, except in an improvement of the GCUK method by Lin et al. [2005].

The data involved in this study are four data sets that were retrieved from the UCI Machine Learning database [Lichman, 2013] and two imbalanced data sets; Christchurch Road Traffic Accidents (CRTA) and United Kingdom Road Traffic Accidents (URTA). All the data sets from the UCI Machine Learning database contain categorical data which were then recoded into binary attributes.

This study does not focus on the natural clusters for each of the variables as the intention is to see the performance of the proposed method when compared to other clustering methods (standard K-means, GCUK, NSGAII, IKM and SKM). The time taken to converge by the proposed method was not recorded in this study and is suggested for future research.

1.5 Overview of Thesis

The aim of this thesis is to propose a new clustering method that can be applied specifically for binary and imbalanced class data sets. It is organised as follows:

Chapter 2 provides a brief explanations on GA.

Chapter 3 provides a literature reviews on GA based clustering.

Chapter 4 implementation of GCUK that reflect some modifications on the data representation to suit the binary environment. Here, the performance of GA to imbalanced data was tested too. These results, have been discussed at the conferences at the Otago University, University of Wellington and University of Auckland. A paper that explained some part of the results has been submitted to International Conference on Education, Mathematics and Sciences 2016 (ICEMS2016) and The Asian Mathematical Conference (AMC 2016 Bali), Bali, Indonesia.

Chapter 5 introduces the new proposed GA based clustering method. This is a combination of simple GA with the IKM and Scalable K-means. A part of the analysis of the proposed method was given in Saharan and Baragona [2013].

Chapter 6 concludes this thesis by summarizing all the results from the study and discussing all the possible extensions from the modified and proposed method.

Chapter 2

Overview of Genetic Algorithms

2.1 Introduction of Genetic Algorithms

GAs are one of the best ways to solve a problem for which little is known. These algorithms can solve both constrained and unconstrained optimization problems based on a natural selection process that mimics biological evolution. GAs work within a search space that contains all feasible solutions. Each point in the search space represent one feasible solution and each of these feasible solutions will be marked by its fitness value. The set of solutions is called a population. One solution is taken from the population (P) to form a new population (P'). A strong possible solution will survive to the next generation, but weak solutions will be eliminated from the population. The critical problem in the GAs process is to assign the fitness function and the initialization of the starting point in the search space [S.N. Sivanandam and S.N. Deepa, 2008].

Before we can further discuss GAs, there are some of the terms that are usually used in GAs literature. These terms are shown in Table 2.1:

Table 2.1: Terms in Genetic Algorithms

Term	**Definition**
String	The individual part to form the chromosome, e.g: Chromosome: 43571 : String: $4, 3, 5, 7, 1$
Chromosomes	A set of strings and one chromosome represent one solution
Population	The set of the chromosomes
Fitness	The value that measured how far or close the chromosomes from the solution.
Fitness Function	A particular type of objective function that is used to summarize and measure how close a given design solution is to achieving the set aims. This is the problem specific in the GA and very crucial.

The solutions in the search space are nothing without the process of the combination and crossover. The new population will hopefully contain better solutions from the current population and this is only done after the two solutions are combined. Furthermore, to make it more unique, some of the characters in the solutions will be changed or mutated with a very small probability. The objective of these two operators is to make sure that the new population will be fitter compared to the old population [Goldberg, 1989, S.N. Sivanandam and S.N. Deepa, 2008].

GA is an iterative process of three major operators of selection, crossover and mutation as illustrated in Figure 2.1. The process of GA iterations will be continued until the stopping criterion satisfied.

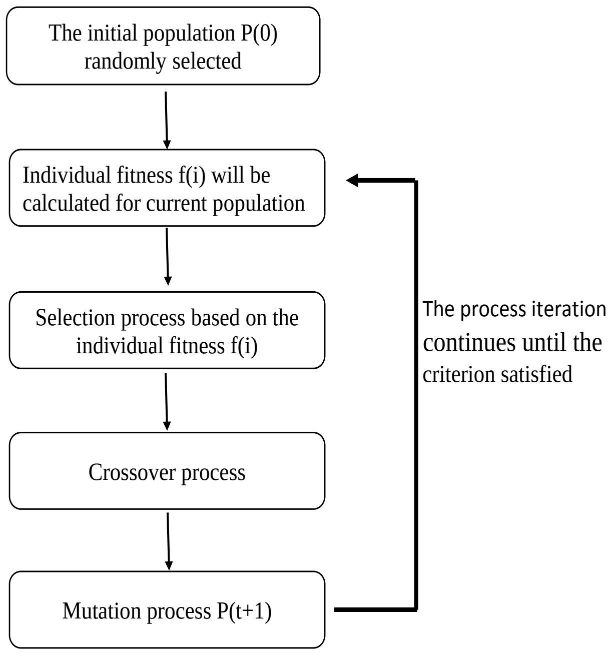

The initial population P(0) randomly selected
Individual fitness f(i) will be calculated for current population
Selection process based on the individual fitness f(i)
Crossover process
Mutation process P(t+1)
The process iteration continues until the criterion satisfied

Figure 2.1: Basic steps in GA

GA applies an artificial process of evolution to construct a robust search which requires minimal information. It starts with the initialization of the population or the potential solutions of the problems. This initialization is represented by the chromosomes which is a set of genes, with each gene carrying the characteristics of the data set. These chromosomes have their own fitness values depending on the objectives of the clustering, so it is very important to determine the right objective function(s). The process of evolution or building a new generation is a series of selection, crossover and mutation processes. The selection process, is based on the value of the fitness. If the chromosome is fit enough (has achieved the objective function), then it can be selected and the weakest chromosomes will be eliminated from the population. Then, these chromosomes will go into the mating pool and two chromosomes will be randomly selected to mate and combine some of their genes to produce a new offspring. A small change (mutation) of a gene in a child may happen with a lower probability. The idea is to make the child more unique or have a high quality compared with their parent. The mutation process also has a role in preventing the algorithm from drop into the local minimum. Thus, as new generations develop, the quality of the chromosomes or the possible solutions are also increased. This process is repeated until some criteria are met or some predefined number of generations is achieved.

2.2 Encoding

The first step in GA is to choose the suitable encoding to represent the chromosomes or the possible solutions. The users will have to encode their problem's variables into the chromosomes environment. Usually GA uses a binary encoding, but many researchers such as Bandyopadhyay and Pal [2007], Baragona et al. [2006], Jie et al. [2003], Maulik and Bandyopadhyay [2000], Mukhopadhyay et al. [2009] used real numbers for encoding in GA. The binary encoding only contains 0's (absent) and 1's (present) and it was first used because of its relative simplicity [Goldberg, 1989]. These chromosomes will have a length of l that represent solutions for every objective. The number of solutions that can be obtained is then clearly 2^l.

There are three methods of encoding: point-based encoding, centre-based encoding and locus-based adjacency. Of these point-based and centre-based encoding are the methods that widely used by researchers.

2.2.1 Point-based Encoding

Many early researchers such as Krishna et al. [1999], Krovi [1992], Murthy and Chowdhury [1996] used point-based encoding in their research. Point-based refers to the length of the data points or the chromosomes. Each of points is assigned to a cluster $(C_1, C_2, \ldots, C_K)$ based on their position in the chromosomes. For example, if the data point i is assigned a value of j, then this means the i^{th} point belongs to j^{th} cluster. Although this encoding is simple and very easy to use, when the chromosomes become very big, they will face the cost in computational time to converge because the longer the chromosomes, the longer it takes for the method to converge.

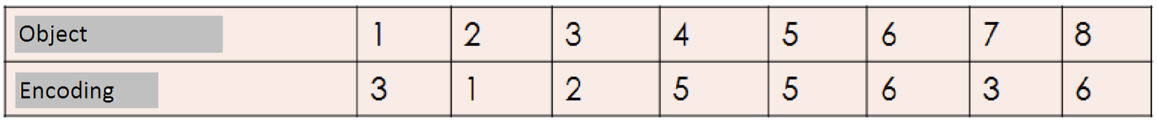
Lu et al. [2004b] used this encoding when they proposed extension research of genetic K-means algorithms (GKA) by Krishna et al. [1999]. Their proposed method, which used the concept of incremental process, was shown to perform well and converge to the global optima in comparison to GKA.

2.2.2 Centre-based Encoding

Centre-based encoding uses cluster centres as chromosomes and has a length of $K \times d$ where d is the dimension of the data set and K is the number of clusters. Hence, K and the length of the chromosomes can be varied. This encoding can make the process of convergence faster as the number of clusters usually is shorter than the length of the data points. Well-known researchers who use a centre-based encoding are Bandyopadhyay and Maulik [2002a]. They proved that the proposed genetic based clustering algorithms using centre-based encoding works efficiently and is much faster to converge.

2.2.3 Locus-based Adjacency Representation

Locus-based Adjacency Representation (LAR) uses a graph as a chromosomes representation [Handl and Knowles, 2007]. Each of the solutions consist of n genes or elements, where n is represents the number of data points. Each gene can have a values from $1, \ldots, n$. If gene i is assigned a value j that represents a connection link between the data points i and j, these two points will belong to the same group (see Figure 2.2).



Object	1	2	3	4	5	6	7	8
Encoding	3	1	2	5	5	6	3	6

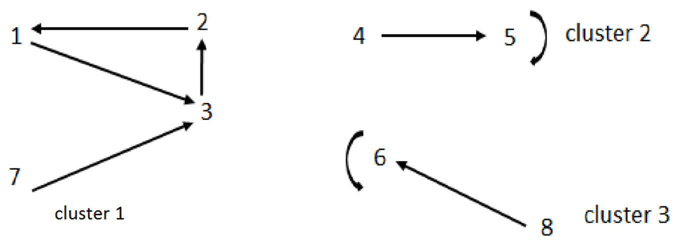


Figure 2.2: Locus-based Adjacency Representation illustration

In Figure 2.2, all the cliques become clusters. The advantage of LAR is that the number of clusters are automatically formed when all the genes are connected with each other [Handl and Knowles, 2007].

2.3 Fitness Function

GA works with the set of solutions and not the decision variables like the other statistical methods [Deepa and Punithavalli, 2010, Goldberg, 1989, Mitchell, 1995]. After creating the solutions which are represented by the chromosomes, each of these chromosomes will be evaluated for their fitness depending on the fitness function. Chromosomes that have the fittest value will survive to the next generation. The fitness function depends on the objective of the problem statement. Most of the fitness will be made equal to the objective function value. If the problem statement is to have a minimum cost of some product, then the optimization function here is to find the lowest of the fitness values. This indicates that the chromosome that has a lower fitness value will have a minimum cost of the product compared with the other chromosomes.

Before the process of GA starts, the termination condition will be checked first to see whether the termination criteria has been met, depending on the fitness value calculated before. If the termination condition is not met, these chromosomes will then be put through all the process of GA again; selection, crossover and mutation. All these operators are to modify the chromosomes so they can meet the termination criteria after creating new and better solutions.

Specification in the fitness function is one of the crucial problems in GA because it will determine which chromosomes can survive to the next generation and which will be eliminated from the population. A good choice of the fitness function will give a good answer for the problem statement.

2.4 Elitism

Sometimes during the evolution process, the good chromosomes can be lost and eliminated from the current population. The function of elitism is to make sure that the good and strong chromosomes can be carried to the next generation by storing them outside the current population. If there are any weak chromosomes in the current population, these chromosomes will be taken out and replaced with the strong chromosomes that have been stored before to make sure that the current population will only contain the best chromosomes. This weak chromosome may be selected during the selection process. Although this solution may be weak, it may contain some useful components which could improve the solution during the process of crossover and recombination. Elitism can be applied in many ways. One way is to combine both parent and child to produce a new population with all competing to survive to the next generation. The use of elitism has been proven to produce better children and can help to converge at the global optimal solution [Rudolph, 1994]. Elitism is not a constrained for the algorithm, but this process will make sure that the next generation will only have the best chromosome.

Elites for single optimization problem can be easily identified. The solutions that meet the objective function(s) values indicate better solutions. However, for multi-objective optimization, the definition of elitism is rather tricky. An intermediate degree of elitism α has to be introduced first to choose the number of elites that can be carried to the next generation. The choice of the α value will influence the diversity of the new population. If the value of α is very big, then there will no diversity in the new population as it is only filled by the elites and chances of the children to get into the new population is very small. This stunts the progress of the new generation. However, if the choice of α value is very small, the advantages of the elites will not be exploited. Usually, the common value of α is from 1 to $0.1N$ from the size of the population that has been used [Deb and Kalyanmoy, 2001].

Elitism can be explained as the process that protect the best chromosomes from being eliminated during the process of evolution. It is not a constrained for the algorithm, but this process will make sure that the next generation will only have the best chromosome. Reeves and Rowe [2002] showed that GAs which do not use the elitism may not converge to the optimal solution.

2.5 Selection Operator

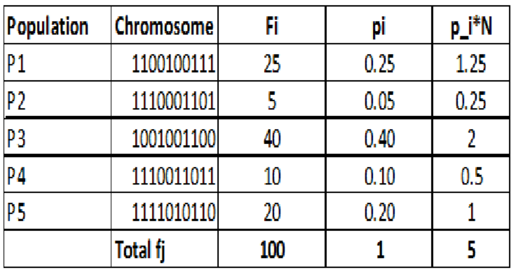
The selection process is the process to select the fittest chromosomes to be copied to the next generation. The higher of the fitness value, the higher the probability to be selected and copied many times. To maintain the size of the population, GA only selects the fittest chromosomes and eliminates the weak chromosomes from the population. The objective to have a good set of solutions [Goldberg, 1989] can be achieved by :

- Only selecting the fittest chromosomes
- Making many copies from the fittest chromosomes
- Eliminating the weak chromosomes, so the next generation only contains the good chromosomes.

The function of the fitness value is exclusively in the selection process. There are three common types of selection operator, Fitness proportional / Roulette Wheel Selection [Goldberg, 1989], Stochastic Universal Sampling [Baker, 1987] and Binary Tournament Selection Tournament [Goldberg, 1989]. The details of these selection operators are given below:

2.5.1 Fitness Proportional / Roulette Wheel Selection

Roulette Wheel Selection or RWS is another name for Fitness Proportional Selection. This selection process uses the proportion of the solutions which will affect the area in the wheel. The higher proportions will have a larger area in the wheel and vice versa. Let us say, F_i is the fitness value of all the chromosomes, the probability of the chromosome i can be calculated by $p_i = \frac{F_i}{\sum_{j=1}^{N} F_j}$.



Population	Chromosome	Fi	pi	p_i*N
P1	1100100111	25	0.25	1.25
P2	1110001101	5	0.05	0.25
P3	1001001100	40	0.40	2
P4	1110011011	10	0.10	0.5
P5	1111010110	20	0.20	1
	Total fj	100	1	5

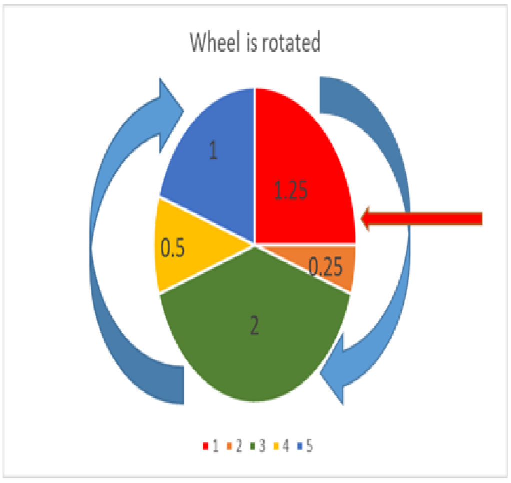
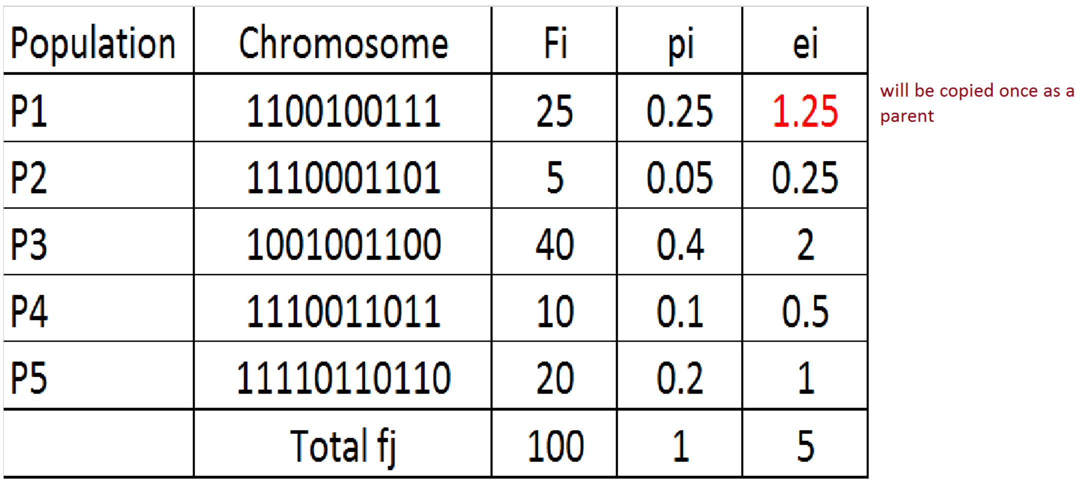


Figure 2.3: Roulette Wheel Selection Illustration

The wheel in RWS then will be partitioned according to this probability where the higher probability will have a bigger area and the lower probability will have the smaller area (Figure 2.3). When this wheel is spun N times, the pointer will stop at one of the area in the wheel. The pointer will choose the chromosomes that can be selected to the next generation. This means that the bigger area in the wheel will have a much higher chance of being selected compared to the smaller areas on the wheel. However, the process of RWS is slow because in order to find the average fitness, one must compute the fitness for all the chromosomes in the population. Furthermore, RWS is a biased selection because the chance of the small area being selected is very low [Goldberg, 1989]. Even though that there is a biased in the selection area, RWS still has an advantage where the weaker solutions have a chance to be selected and may survive in the next generation. Although this solution may be weak, it may contain some useful component which could improve the solution during the process of crossover and recombination.

2.5.2 Stochastic Remainder Roulette Wheel Selection, (SRWS)

The Stochastic Roulette Wheel Selection or SRWS uses a concept of removing or copying the strings based on the values of the reproduction counts. The process is done by computing the reproduction count associated with each string. At first, the probability of the selection $p_i = F_i / \sum F_i$ is calculated. The expected number of the individuals of each string is then $e_i = p_i \times P$, where P is the population size. For example, a string with $e_i = 1.25$ is listed once as a parent because the integer part is 1 (see Figure 2.4). This is continue until all the strings in the population are examined. The strings with non-zero counts get multiple copies in the population equal to the value of their counts, while the strings that have zero counts are eliminated from the population. The size of the population is kept constant and this completes the reproduction operation.



Population	Chromosome	Fi	pi	ei
P1	1100100111	25	0.25	1.25
P2	1110001101	5	0.05	0.25
P3	1001001100	40	0.4	2
P4	1110011011	10	0.1	0.5
P5	11110110110	20	0.2	1
	Total fj	100	1	5

will be copied once as a parent

Figure 2.4: Stochastic Roulette Wheel Selection Illustration

2.5.3 Stochastic Universal Sampling (SUS)

Baker [1987] proposed SUS to overcome the problem of $O(N)$ computational time in RWS and SRWS. SUS is more unbiased compared with RWS and SRWS because it uses weighted random sampling in order to compute the areas of the sections in the wheel. Even though the sections of the solutions still depend on the fitness of values, there is only one random number r needed to carry out the whole selection process. For example, we have R random number for S different solutions, where each number of R can be chosen by the cumulative of the probability before.

$$R = \{r, r + \frac{1}{S}, r + \frac{2}{S}, \ldots, r + \frac{(S-1)}{S}\}\text{mod}1$$

This means that the chances that the pointer will stop at any of the sections on the wheel now become equal (see Figure 2.5). However, because the area of the section on the wheel still depends on the fitness values, the fittest chromosomes still have better chances to be selected for the next generation compared to the weak chromosomes.

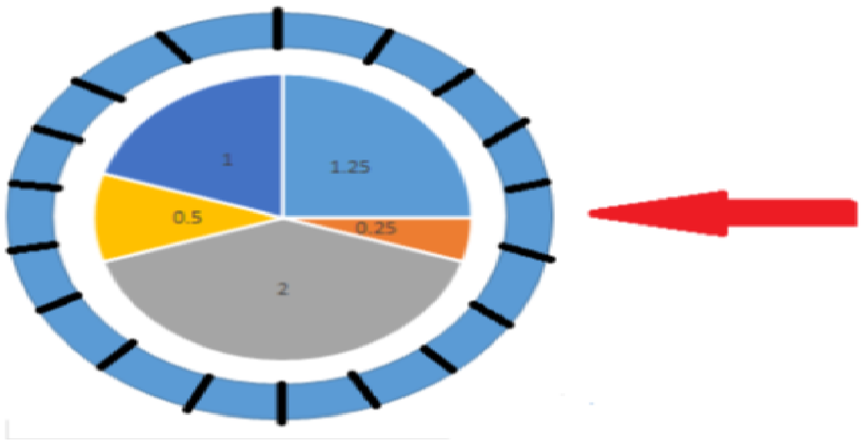


Figure 2.5: Stochastic Universal Sampling Illustration

2.5.4 Binary Tournament Selection

This selection method is straightforward to use, and easier to understand the process. The tournament is played with only two chromosomes (solutions) and the better chromosomes will be chosen and go to the mating pool. Two chromosomes are selected at random, and their fitness values compared. The winner (higher fitness) goes into the mating pool. At the end, in the mating pool, there will be a pair of chromosomes that has been chosen according to their fitness. These random pairs of chromosomes will be compared again and if the chromosomes win two times, it will have two copies for the next generation. On the other hand, the weakest pair of chromosomes will not have any copies at all.

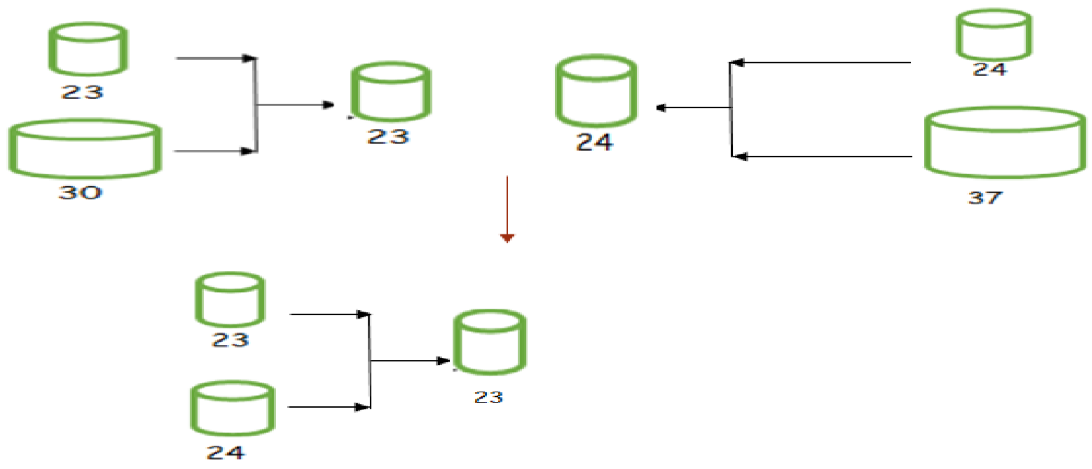


Figure 2.6: Binary Tournament Selection Illustration

Figure 2.6, is an example of the cost to produce a can drink. The objective is to find the minimum cost of the can production. Thus, the minimum cost of can production will be the winner during the selection.

2.6 Crossover Operator

After selecting only the fittest chromosomes in the population, two random chromosomes (parents) will be mated and will produce a child or offspring. The process is not creating new chromosomes but just exchanging some of the genes from the parents. The objective of this process is to copy as many good genes or characteristics from the parent to have better offspring for the next generation. There are many types of crossover, but most crossover operators will involve two random parents which will exchange some of their good genes [Goldberg, 1989]. There are three techniques of crossover that are widely used; single-point crossover, two-point crossover and uniform crossover.

2.6.1 Single-point Crossover

In this technique, along the length of the chromosomes, a cross site will be randomly chosen. The first side is called a “head” and after the cross line it will called a “tail”. The size of the tail is determined by the probability of crossover p_c. The genes or string of the chromosomes for these two chromosomes will be exchanged and will produce two new children.

There are 2^{n-1} possible string pairs produced by the crossover process, but for the single-point technique only n different string pairs are created. Since in the selection process, we have selected only the fittest chromosomes and eliminated all the weakest chromosomes, there is a high chance that the combination of the two randomly chromosomes will now produce better chromosomes that can compete with their parents. The process in the crossover does not change the character of the chromosomes because only some of the genes in the parent will be exchanged and the child produced will still have similar characteristics with its origins, but with some new and better characteristics [Deb and Kalyanmoy, 2001, Goldberg, 1989, S.N. Sivanandam and S.N. Deepa, 2008].

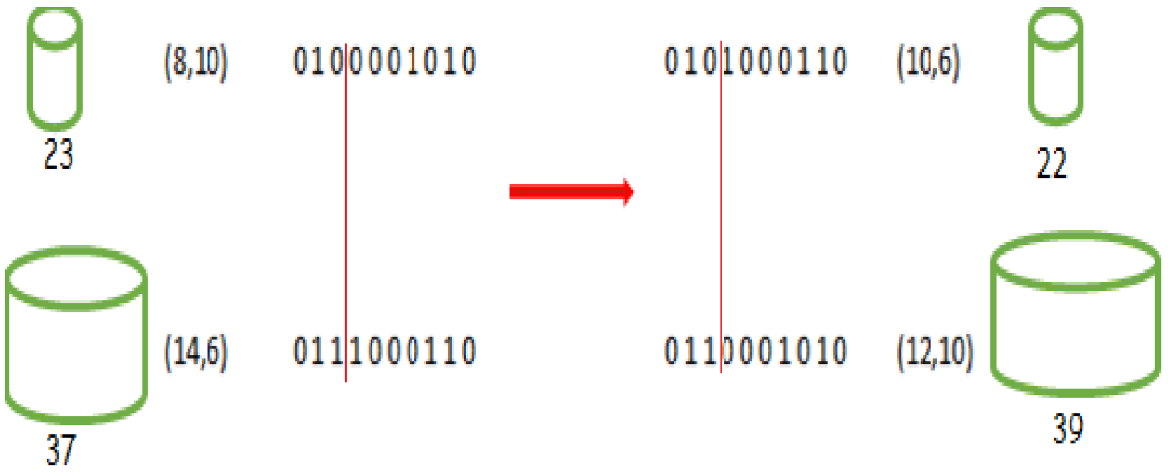


Figure 2.7: Single Crossover Illustration

In Figure 2.7, one single line was draw after the third gene for both chromosomes. These chromosomes will exchange their genes after this single line. As a results, the new chromosome has a better solution than before.

2.6.2 n-point Crossover

The n-point crossover technique will have n- cross sites along the length of the chromosomes and the parents will exchange their strings between these line and produce another two children. From this technique, GA users can expect to have n points in their chromosomes depending on their interests. If there is an even number of cross sites chosen, the process of exchanging the string will be applied alternately along the line. However, if there is an odd number of cross sites, the users should consider the last string as an additional cross site, and then the total will become an even number. [Deb and Kalyanmoy, 2001]

2.6.3 Uniform Crossover

Uniform crossover has a different approach compared to single-point and n-point crossovers. Here, a binary mask chromosome that has the same length as the parent will be created and the parents will change their strings according to the number appearing in the binary mask chromosome. If number 1 appears, the parents will exchange their genes, but if number 0 appears, then there will be no change in the chromosomes. For this technique, the probability of crossover, $p_c = 0.5$ will be usually used.

The probability of crossover can be defined as $100p_c\%$, which means that all the strings in the population are used for the process of crossover. However, only $100(1 - p_c)\%$ will be copied for the next generation. This is to preserve the good chromosomes from being eliminated during the process of exchanging the strings.

2.7 Mutation Operator

The last process in GA is the mutation process which prevents the new population from stopping at the local optima and keeps the diversity in the population. In other words, this process will create a unique and more fit offsprings in the population. With a very small probability (usually $p_m = 0.001$ suggested by Bandyopadhyay and Maulik [2002b]) not all the chromosomes will be mutated. The process is to change a string using a binary flip 1 to 0 and vice versa for the binary chromosomes.

Goldberg [1989] proposed a *mutation clock* in order to overcome the problem of the computational complexity in the original mutation process. He used the exponential distribution to find the next location to change the string by using the first changed string location. A random number of $r \in [0, 1]$ will be assigned first and then the mean of the distribution can be computed by $\mu = \frac{1}{p_m}$. The next string location can be find by skipping $\sigma = -p_m(ln(1-r)$. Here the computational time of the random number is less because now the average $O(1/p_m)$ of the computational time has been generated.

The mutation in this process is aiming to explore more searching space while the crossover tries to converge on some point. This is because the role of the mutation is to solve the local minimum problem. Thus, this process is one way to prevent local minimum solutions at the expense of exploring more areas.

2.8 Summary

In this chapter, the general concepts of Genetic Algorithms (GAs) were briefly discussed. GAs plays with the idea of the evolutionary process where the chromosomes will have to compete with each other to have a place in the next generation. A strong chromosome can survive, and usually the weak chromosomes are eliminated from the population. GAs is working with a search space that contains all feasible solutions. It means that each of the points in the search space is represent one feasible solutions that will be marked according to its fitness of the objectives. The core processes of GAs are selection, crossover and mutation. All the processes of GAs make this algorithm more unique compared to other conventional algorithms for the optimization. The selection process aims to select the good chromosomes which will be sent to the mating pool to combine with the other chromosome and produce two new children and copying a good strings/ characteristics from their parents. Meanwhile the mutation process aims to encourage diversity in the new population with a very small probability.

General differences between GAs with Simulated Annealing and Hill Climbing are described below:

Genetic Algorithms (GAs) : GAs are basically random and are based on natural evolution, such as selection, crossover and mutation. The randomness can be controlled in guided searches. The degree of randomness can be controlled by changing the setting of the parameters value. GAs are built from a set of chromosomes called a population. Each of the chromosomes will be tested for their fitness depending on the objective functions. If the chromosome is strong or fit enough, it can be selected for the next generation, while the weakest chromosome will be deleted from the population. This can be done by the process of selection, crossover and mutation.

Simulated Annealing (SA) : SA is a metaheuristics algorithm that can be used to approximate global optimization in a large search space. The inspiration behind the name of simulated annealing come from annealing in metallurgy. This is a technique that involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. The slow cooling means as a slow decrease in probability of accepting the worse solutions. This allows the algorithms to explore more solution space.

Hill Climbing (HC) : HC algorithm is an optimization technique that starts with an arbitrary solution to a problem. It attempts to find a better solutions by incrementally changing a single element of the solution. If the change produces a better solution, an incremental change is made to the new solution. This process will be repeated until no further improvements can be found.

In the next chapter, the brief explanation and past literature reviews on GA based clustering are discussed.

Chapter 3

Genetic Algorithm Clustering

3.1 Introduction

GAs is one of the most promising algorithms that have consistently performed well in solving clustering problems. The capability of GAs have been proven to give efficient and effective results and provide appropriate clustering. In the past several years, GAs has been extensively used as an optimization method in various domains such as ([Cole, 1998, Goldberg, 1989, Mitchell, 1995]), image processing [Arunprasath et al., 2010, Fatima Zohra Bellala Belahbib, 2011], clustering [Baragona et al., 2006, Li et al., 2010, Lin et al., 2005, Maulik et al., 2011, Murty et al., 2008, Paterlini and Minerva, 2010, Wang and Zhang, 2009] to name a few examples. Many of GAs's various applications can be found in journals such as IEEE, Information Sciences and the Journal of Evolutionary Process.

There are several literature reviews that focus on the application of GAs to cluster binary data. All of these methods showed good performance and better results when compared to other clustering methods. However, some of these methods have drawbacks and need to be improved to be better clustering algorithm.

3.2 Past literature on Genetic Clustering

Cluster analysis is a multivariate statistical method that is often used to discover the patterns of a given data set. The data set contains information on variables and usually one attempts to reorganize the useful variables that have the same characteristics into the same group or cluster. However, the main challenge with the clustering is that different clustering algorithms may provide different clusters for the same data set. A good clustering algorithm is the algorithm that can reflect the natural clusters in a data set and at the same time attain the lowest validity index value. The clustering validity indexes usually measure the compactness and the separability of the clusters.

Genetic K-means Algorithm, GKA is one of the examples of GAs that was proposed to improve the performance of K-means. The main objective of GKA is to find the global optimal of the given data set and partition the data into a specified number of clusters. In GKA, instead of using a common crossover operator, Krishna et al. [1999] used K-means as a search operator. The problem of minimizing the Total Within Cluster Variation (TWCV) was also handled successfully by GKA. Lu et al. [2004a] proposed a *Fast Genetic K-means Algorithm*, FGKA, which was inspired by GKA, by incorporating several improvements over GKA. Both FGKA and GKA achieved the objective of their studies which was to converge to the global optima, and the study found that FGKA runs faster than GKA. In FGKA Lu et al. [2004a] defined the number of non-empty clusters $e(S_z)$ in any solution S_z as a legal (accepted) if $e(S_z) = 1$, and illegal otherwise. These strings are permitted but are considered as the most undesirable solution and the lowest fitness value was defined as $+\infty$. These illegal strings were not permitted and were eliminated from GKA, which showed that FGKA was better. This process improves the performance of the time convergence. Another extension of GKA and FGKA is the *Incremental K-means Algorithm*, IGKA [Lu et al., 2004b]. This algorithm outperforms FGKA when the probability of mutation is small. IGKA was proposed where the objective of the study was to cluster the centres incrementally whenever the mutation probability was small. IGKA always performs well and converges to the global optima.

Maulik and Bandyopadhyay [2000] proposed a GAs based clustering where chromosomes were represented by the strings of the real numbers and encoded with a fixed number of cluster centres in R^N. This algorithm was then extended by Bandyopadhyay and Maulik [2002b] and named as *Genetic Clustering for Unknown K* (GCUK). In GCUK, the strings were comprised of the real numbers. Bandyopadhyay and Maulik [2002b] proposed the symbol of # or "do not care" symbol. This symbol was to represent any empty cluster in the strings. The algorithm automatically evolves with the number of clusters too. Both algorithms implement the elitism strategy whereby the best chromosomes are preserved at each generation outside of the population to prevent them from being eliminated during the iteration process. To check the performance of the algorithm, Maulik and Bandyopadhyay [2000] compared the minimum value of the objective function, M, in the K-means algorithm with the same K, and showed that the GCUK outperform the K-means. In GCUK, Bandyopadhyay and Maulik [2002b] used the Davies Bouldin (DB) index to measure the validity of the clusters. These two algorithms used the Euclidean distance to calculate the distance from a point to a cluster centre. GCUK became the most effective GAs clustering method but due to the real number representation, it took a longer time to converge [Lin et al., 2005].

In a paper by Lin et al. [2005], the cluster centres were selected directly from the data set and they constructed the look-up table to save the distances between all pairs of the data points. This process allowed the algorithm to speed up the evaluation of the fitness value. In GCUK [Bandyopadhyay and Maulik, 2002b] and GAs clustering by Maulik and Bandyopadhyay [2000], the string representation was used to encode the variable number of cluster centres, while Lin et al. [2005] used the binary representation. The new, and more effective versions of the GAs operators (selection, crossover and mutation) were introduced in this paper. This algorithm also used Euclidean distance and the DB index as the clustering metric to compare the validity of the clustering. Lin et al. [2005] clustering algorithm performed better than GCUK in terms of validity and time efficiency but some of the tested data produced unnatural clusters.

Validity Clustering

Validity clustering is done to measure the quality of the clustering methods based on the compactness and separateness of the clustering. There are two types of validity clustering:

External Index Used to measure the extent to which cluster labels match externally supplied class labels. Eg: Rand and Adjusted Rand index

Internal Index Used to measure the goodness of a clustering structure without respect to external information. Eg: Davies-Bouldin index, Dunn's index, Xie-Beni index, Silhouette index

Rand Index

[William M . Rand, 1971] proposed an index for measuring the agreement between two clusterings, named Rand Index. This index is a number between 0 and 1. When the value of Rand index is 1, the two partition agree perfectly. The formula for the Rand Index is given by;

$$Rand = \frac{\binom{n}{2} + 2\sum_{i=1}^{c_1}\sum_{j=1}^{c_2}\binom{n_{ij}}{2} - [\sum_{i=1}^{c_1}\binom{n_i}{2} + \sum_{j=1}^{c_2}\binom{n_{.j}}{2}]}{\binom{n}{2}}$$

Here n_{ij} is the number of points that are in cluster i for Method A and cluster j for Method B. c_1 is the number of clusters for Method A and c_2 the number of clusters for Method B. Also, $n_{.j} = \sum_{i=1}^{c_1} n_{ij}, n_{i.} = \sum_{j=1}^{c_2} n_{ij}$ and $n = \sum_{i=1}^{c_1}\sum_{j=1}^{c_2} n_{ij}$.

Adjusted Rand

The Rand Index tends to give quite large values even when clustering methods are in substantial disagreement. [Hubert, 1985] proposed an adjustment to the Rand Index by considering a distribution for assigning points to clusters under the condition that cluster sizes remained unchanged. The Adjusted Rand Index is calculated as follows;

$$\text{Adjusted} = \frac{\binom{n}{2}\sum_{i=1}^{c_1}\sum_{j=1}^{c_2}\binom{n_{ij}}{2} - \sum_{i=1}^{c_1}\binom{n_{i.}}{2}\sum_{j=1}^{c_2}\binom{n_{j.}}{2}}{\frac{1}{2}\binom{n}{2}[\sum_{i=1}^{c_1}\binom{n_{i.}}{2} + \sum_{j=1}^{c_2}\binom{n_{.j}}{2}] - \sum_{i=1}^{c_1}\binom{n_{i.}}{2}\sum_{j=1}^{c_2}\binom{n_{.j}}{2}}$$

Jaccard Index

Jaccard index was used purposely for binary data. The definitions of binary similarity can be expressed by Operational Taxonomic Units (OTU) (Table 3.1).

Table 3.1: Operational Taxonomic Units

i \ j	Present	Absence	Total
Present (1)	$a = (1, 1)$	$b = (1, 0)$	$a + b$
Absence (0)	$c = (0, 1)$	$d = (0, 0)$	$c + d$
Total	$a + c$	$b + d$	$P = a + b + c + d$

where;

- a = both the event i and j are present (1,1)
- b = event i is present but absent in event j (1,0)
- c = event i is absent but present in event j (0,1)
- d = both event i and j are absent (0,0)
- n = is the total sum of the variables

Jaccard Index can be calculated as following:

$$J = \frac{a}{a + b + c}$$

Misclassification Error Rate

The error rate is a prediction error metrics for a binary classification problem. It can be calculated by using a Confusion Matrix:

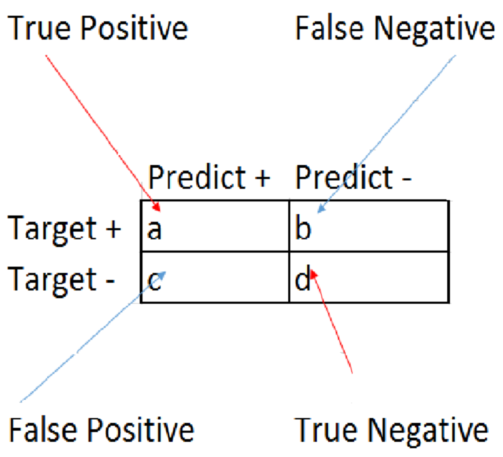


Figure 3.1: Confusion Matrix

The confusion matrix (Figure 3.1) shows the results for two class classification problem where:

- **a + d = good prediction**
- **b + c = bad prediction**

3.2.1 Multi Objective Genetic Algorithms, MOGAS

Most of existing clustering methods can only optimize a single objective function. However, in real world data, the types of data are not limited to solely numeric attributes, but many consist of categorical, binary and ordinal data. This in turn necessitates multiple objective functions. For example, in a clustering for categorical data, it is necessary to optimize the compactness and separation of the clusters separately and not just combine it into a single measure. GAs is one of the clustering techniques that provide the ability to optimize two or more objective functions simultaneously.

Mukhopadhyay et al. [2015, 2014a] and Mukhopadhyay et al. [2014b] have provided a detail explanation of the multi objective evolutionary algorithms. They have clearly explained various of application of multi objective optimization in data mining tasks such as association analysis, clustering, classification, ensemble learning and etc. All these techniques are reviewed followed by some real life applications of multi objective clustering in the various domains. Many of the multi objective clustering techniques are based on two most prominent algorithms: (*i*) the *Non-dominated Sorting Genetic Algorithm II*, NSGAII [Deb et al., 2002] and (*ii*) the *Pareto Envelope-Based Selection Algorithm II*, PESA-II [Corne et al., 2001]. NSGAII was proposed to eliminate the weaknesses of NSGA [Srinivas and Deb, 1995] in terms of the high computational complexity, lack of elitism and need for the sharing parameters to ensure diversity in the population. The objective functions of NSGAII are to minimize simultaneously the Xie-Beni index (XB) [Xie and Beni, 1991] and the Fuzzy C-means (FCM) index [Bezdek, 1981]. The selection techniques in NSGAII have become highly efficient and perform favourably in comparison to other clustering methods.

PESA-II empoyed by a new region-based selection, as an alternative of the selection operator that was used in GAs. The mating selection was calculated by using the region-based (hyperbox) and not individual-based like the other evolutionary multi objective (EMO) algorithms. In region-based selection, a hyperbox was selected and then the individuals of the genetic operations were randomly chosen from this selected hyperbox. For individual-selection process, the candidate individuals in the population are inserted into the archive set one by one . However, this selection process needs an incremental update mode which leads to extra time consumption. This algorithm also has poorer distribution range. Li et al. [2013] introduced the IPESAII- *Improved Pareto Envelope-Based Selection Algorithm II* as a solution for all of these problems. In IPESAII, the effective improvements were made in the environmental selection by keeping the boundary of all the individuals in the population and extending the distribution range of the solution set.

Another well-known MOGAS is *Multi Objective Clustering with automatic K-determination* (MOCK) by Handl and Knowles [2007]. The basic algorithm in the clustering phase of MOCK is the selection algorithm which is taken from PESAII [Corne et al., 2001]. This algorithm uses the evolutionary algorithm to enable the whole Pareto front to be approximated as a single algorithm run. MOCK was proposed from the idea of *Voronoi Initialized Evolutionary Nearest-Neighbor Algorithm* (VIENNA [Handl and Knowles, 2004]), the first multi objective GAs that which was proposed earlier. The improvement from the VIENNA method includes a new method of selecting the Pareto front from a null model and the ability to determine the number of clusters automatically. MOCK's advantages are providing automatic estimate of the quality of the individual solutions and keeping the number of clusters dynamic. Despite the advantages in MOCK, it still has some inherent limitations. There is no guarantee of the best clustering solution for the new generation as MOCK only provides an approximation of the unknown Pareto front. MOCK also requires expensive computation for some applications such as real time document clustering.

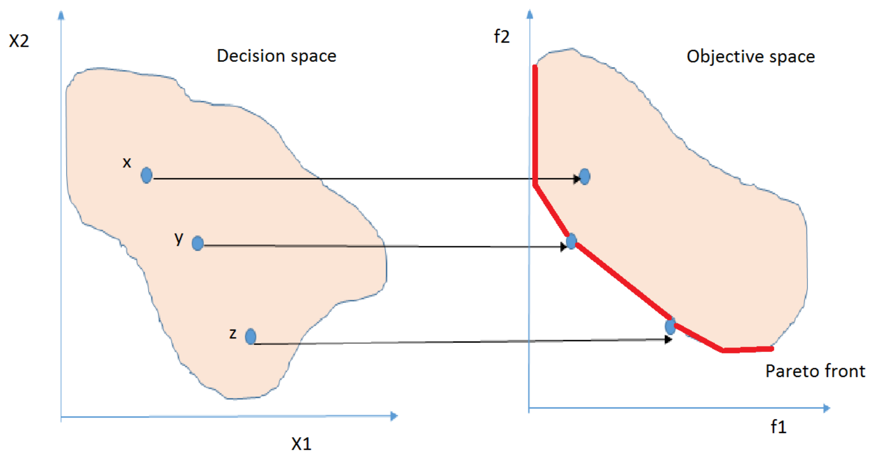


Figure 3.2: Illustration of Pareto-front

In Figure 3.2, the red line is called the Pareto front and solutions on it are called Pareto-optimal. All Pareto-optimal solutions are non-dominated. Here, x is dominated solution, while y and z are non-dominated solutions.

3.2.2 GAs for Categorical Data Sets

Most researchers focus on clustering for numeric data in which the dissimilarity between two points in the data set can be computed by using standard distance metric such as Euclidean distance. However, in real life data sets, it is not as clear cut. There are many types of data, like categorical and ordinal data, which the numbers do not have mathematical meaning. There are a specific distance measurements for categorical data such as Overlap, Eskin, IOF and etc [Boriah et al., 2008].

Mukhopadhyay et al. [2009] proposed a MOGA for categorical data that was designed based on NSGAII and adapted a novel technique for the selection process from fuzzy clustering approach. The final promising clustering solution was selected based on the majority voting among the Pareto optimal front of non-dominated and then followed by K-nn classification. Due to the categorical data, the chromosomes were represented by the cluster modes. This algorithm gave a better performance compared to the other well-known categorical data clustering algorithms. For further research, they suggested the use of more than two objective functions optimization and other classification tools.

Jie et al. [2003] proposed the used of GAs for mixed variables (numeric and categorical) but the same problems of predicting the number of clusters still required substantive attention in further research.

3.2.3 GAs clustering for Binary Data

The development of GAs clustering for binary data sets was not as wide as numeric data, but these types of data sets still received considerable attention from researchers. Some of the earliest researchers who used binary data sets in the implementation of GAs for clustering were Kaukoranta and Nevalainen [1997]. In this paper, several data sets of non-binary and binary data sets were used. In their findings, Kaukoranta and Nevalainen [1997] successfully clustered the non-binary data sets using GAs, but found that the method had room for improvement with respect to clustering binary data.

The number of researchers who use binary encoding has been on the increase since early 2000. Binary encoding was used rather than a string representation in the studies by A. Casillas et al. [2003], Lin et al. [2005], Nasraoui and Krishnapuram [2000], Pan and Cheng

[2007], Tseng and Yang [1997]. Most of this research, showed that binary encoding produces better results. Leon et al. [2006] proposed *Evolutionary Clustering with Self Adaptive Genetic Operators* which extended research by Nasraoui and Krishnapuram [2000]. In this research Leon et al. [2006] found that when they used the Euclidean distance, the real encoding was much easier to define for the special genetic operators compared to the binary representation that was used by Nasraoui and Krishnapuram [2000].

Research by Tseng and Yang [1997] and Pan and Cheng [2007], used binary encoding with two crossover points and one crossover point respectively. The difference between these groups was, that Pan and Cheng [2007] adopted this encoding based on a number of clusters that was pre-determined while Tseng and Yang [1997] based their encoding on the building blocks for the crossover operator.

Another group of researchers who used a one point crossover and adopted the binary encoding was A. Casillas et al. [2003]. In their research, they used binary vectors with $N - 1$ elements to encode a Minimum Spanning Tree (MST). MST is a spanning tree of a connected, undirected graph. It connects all the edges together with the shortest distance to its edges. There were two conditions to eliminate the edges by looking at the vector elements. A. Casillas et al. [2003] defined that if the vector elements have the value of “0”, then the edges will be retained. Whereas if the vector elements showed “1”, the edges are eliminated. Thus, this research requires considerable computation ability because of the use of tree-based encoding which needs a lot of calculation to build the perfect tree.

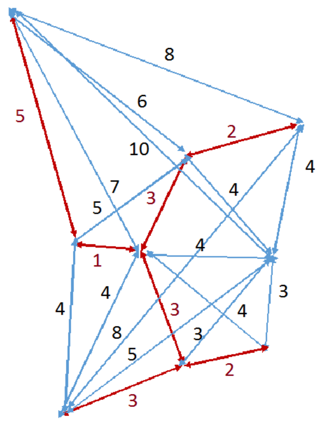


Figure 3.3: Illustration of Minimum Spanning Tree

In Figure 3.3, the minimum distance between the edges are connected together.

Even though GCUK [Bandyopadhyay and Maulik, 2002b] is well-known and the most effective method in GAs based clustering, it has a problem of high computational time to iterate as this algorithm uses real encoding. To overcome on this drawback, Lin et al. [2005] proposed a new efficient GAs clustering approach that replaced the real encoding with a binary representation as a way to encode a variable number of cluster centres. The new GAs operators and the look-up table provided in this algorithm led to improved performance over the original GCUK in terms of time efficiency and the validity of the clustering.

As well as the algorithm proposed by Lin et al. [2005], there are other GAS clustering that use binary encoding. These are the *Binary Small World Optimization Algorithm* (BSWOA) by Wu et al. [2012] and *Genetic Algorithm for the K-Means Initialization* (GAKMI) by Kwedlo and Iwanowicz [2010]. BSWOA is an optimization algorithm based on the searching mechanism in social networks. BSWOA emphasized the local search to find the solutions for optimization problems by using binary encoding. The purpose of GAKMI is focused on improving the performance of the K-means algorithm which uses binary encoding and genetic algorithms. This representation outperformed the K-means and GKA, but in some cases BSWOA performed worse than the earlier two algorithms due to the dependency on some parameter settings and the number of clusters.

In GAKMI, the binary string with length N is represented as the set of initial cluster centre, where N is the size of the learning set X. The feature vector x_i is used as a cluster centre if and only if the ith bit is a set of $U = x_i$. There are two version of GAKMI; Random Repair (RR) and Distance-Based Repair (DBR). In RR, the bits in infeasible chromosomes are fliped randomly in order to convert it into feasible ones. While for DBR, when there are only a few centers, the new feature vectors are added to the set of centers. This result in the possibility of discovering a new clusters which are not covered by the existing one.

From the results, GAKMI with Distance-Based Repair is able to outperformed the K-means initialization methods. Kwedlo and Iwanowicz [2010] planned to modify specialized crossover and mutation operators in order to produce better feasible chromosome without needing any repair of algorithm in GAKMI.

3.2.4 GAS clustering for imbalanced class data sets

Another topic that is most important in this research is clustering for imbalanced class data sets. A data set is imbalanced if the classification labels or categories are not approximately equally represented. Typically, they are composed by two classes; the majority (negative) class and the minority (positive) class. Standard classification algorithms usually consider a balanced training set and this problem affects the bias towards the majority class. A literature review search found that there have only been a few studies done using genetic algorithms to solve this issue.

As mentioned in Chapter 1, standard clustering algorithm cannot be simply applied to this problem as it leads to misclassified results. This is due to the fact that the minority instances may be ignored by the standard clustering algorithm and may not be counted during the iteration process. Usually, the interest of the researchers is to find the relationships or the effect from the minority instances to the whole data set. Therefore, this study will use the advantages of GAs to help with the imbalance issue.

A study by Soler et al. [2009] used membership functions to derive fuzzy rules, while Villar et al. [2012] used linguistic fuzzy rules to perform a feature selection and granularity learning to solve the problem of imbalanced classification. Both methods use the real valued variables (numeric) that was taken from a Down's syndrome problem data set [Soler et al., 2009] and imbalanced data sets from the database [Villar et al., 2012]. The results of these algorithms showed that the implementation or the combination with GAS produced better results as compared to other classical imbalanced classification methods.

3.3 Summary

GAs is one of the well-known optimization methods which can successfully cluster data sets. The main reason to use GAs in this study is because GAs search parallel from a population of points. Therefore, it has the ability to avoid being trapped in local optimal solution like traditional methods, which search from a single point. GAs also uses a probabilistic selection rules, not deterministic ones. Further, GAs work with chromosomes, which are an encoded version of potential solutions' parameters, rather than the parameters themselves. Last but not least, GAs uses fitness scores which are obtained from the objective function without order derivatives or auxiliary information.

GCUK is a well-known and efficient GAs based clustering method that is explicitly design to handle an unknown number of clusters, K. This algorithm produces good results and shows the abilities of GAs for clustering. However, due to the chromosome representation, the drawback from this algorithm is that it is time consuming when applied to large data set.

Based on the literature reviews, GCUK was chosen for this study. This algorithm was shown to be efficient, effective and exhibited good performance in the clustering task. This method has never been applied to cluster binary and imbalanced class data sets. The results of this study are a contribution to this field study.

This method used in this study had been used by the other researchers before, and many of them have proposed a new methods based on the improvement from the original method. Nevertheless, this study is still important and contributes something new. Apart from the different types of data sets that were used, the other interesting contribution is the presence of the imbalanced class data sets in this study.

Chapter 4

GCUK-binary

4.1 Introduction

This chapter presents a GAs based clustering method which was used in this research. The chosen method is based on its performance for single objective optimization. The main objective for this chapter is to exploit the characteristic properties of binary data by using the existing GAs based clustering methods and comparing the performance with the K-means algorithm. In the following section, the most well-known single objective optimization GAs method, GCUK (*Genetic Clustering for Unknown K*) is explained, and discussed. The application to the real life data sets and the results of GCUK-binary are explained in Section 4.3 and Section 4.4 respectively. Finally, the discussion and summary of this chapter can be found in Section 4.5.

4.2 Genetic Clustering for Unknown K, GCUK

Among GAs based clustering algorithm, GCUK is the most effective method for single objective optimization [Lin et al., 2005]. GCUK was proposed by Maulik and Bandyopadhyay

[2000] and Bandyopadhyay and Maulik [2002b] to use GAs to carry out clustering tasks. In this research, Bandyopadhyay and Maulik [2002b] tested the GCUK method with some artificial data and large real data sets from the SPOT image study in Calcutta. They showed that this GA is an effective and efficient clustering method which can be applied to large data sets.

However, due to the real number encoding (string representation), to encode the cluster centres, it has a high cost of computational time for floating-point computation. Inspired by this drawback, Lin et al. [2005] proposed an improvement to GCUK by using a "winner-replacing steps" approach for all three operators in GAs. This proposed method had the ability to outperform the original GCUK. This study did not focus on the research by Lin et al. [2005] as the interest was to test the credibility of GCUK to cluster the binary data specifically.

4.2.1 Encoding

In GCUK, any possible solution (chromosome) was coded as a string of maximum number of clusters, K; k_{max} characters. These characters were either coordinates of the data points or used special character of # or "don't care" symbol to represent any unassigned genes. For example, let $k_{min} = 2$ and $k_{max} = 8$ and the chosen number of clusters be $k = 3$. The chromosome i will encode the centre of 3 clusters which were chosen randomly from the data set. One random distribution, the chromosome for i may look like the following:

$$(12.2, 10.3), \#, \#, (15.3, 3.8), \#, \#, (16.1, 6.7), \#$$

Because GCUK uses a string from a real number encoding to encode the cluster, it incurs a high cost of computational time especially when the data is large. The cluster means should be recomputed each time to compute the fitness (the distance between every data to its closest cluster centre) and this process requires significant computational time.

In GCUK-binary, a set of non-overlapping exhaustive clusters i.e., a partition, is encoded in a chromosome as a sequence of centres. The coordinates of a centre, i.e., the values of the factors associated to the centres, constitute a subsequence of length d, the dimension of the data. So if k is the number of clusters, the length of a chromosome is equal to kd. In this study, the original data is in binary form, so the chromosomes may be assembled direct from the binary data points themselves and it can save time to compute the distance between the data points to their closest centres. In this study, the Hamming distance was used to calculate the distance between a data point and its cluster center. Hamming distance is a number used to denote the difference between two binary strings. More formally, the distance between two strings A and B is $\sum |A_i - B_i|$.

The GCUK determines simultaneously the number of clusters K and the searching for k^{th} is constrained to a suitable interval $[k_{min}, k_{max}]$ where $k_{min} > 1$ and $k_{max} \leq n$. The chromosome has a fixed length of K_{max}. While, in GCUK-binary, the number of K is held to $k_{min} = k_{max}$. The reason to hold the K constant is to make sure that the best number of clusters has been pre-specified and, and its validity tested by the Silhouette index.

The Silhouette index [Rousseeuw, 1987] is one of many cluster indexes that can be used to find the number of clusters. The highest value of the average of the Silhouette index s_i indicates a suitable number of clusters.

This index is another internal cluster index that can be used as a tool to find the suitable numbers of K with a graphical aid to show the performance of the clustering algorithm.

For each datum i, the Silhouette index can be defined as follows:

$$s_i = \frac{b(i) - a(i)}{\max \quad a(i), b(i)}$$

which is

$$s_i = \begin{cases} 1 - a(i)/b(i), \quad \text{if} \quad a(i) < b(i) \\ 0, \quad \text{if} \quad a(i) = b(i) \\ b(i)/a(i) - 1, \quad \text{if} \quad a(i) > b(i) \end{cases}$$

where

$a(i)$ the average dissimilarity of i with all data in the same cluster

$b(i)$ the average dissimilarity of i between the other neighbouring cluster

The smallest values of $a(i)$ will indicate the better cluster while the largest values of $b(i)$ will represent a cluster badly matched to its neighbour.

If the value of s_i is close to 1, then we can say that it is well-clustered. However, if s_i is near to 0, then is not clear in which cluster i belongs to. The larger the average value of the Silhouette index (s_i), the better the performance of the results.

4.2.2 Population Initialization

Let s be the size of the population. For initializing the i^{th} chromosomes, Ch_i in the population ($i = 1, 2, \ldots, s$), the cluster centres are randomly chosen from the data set. In the binary encoding, the gene corresponds to the index of each of the cluster centres chosen is represented as "1"; while each of the remaining gene is represented as "0". For example, if the length of the data id $l = 10$ and $k_i = 4$ for the chromosome Ch_i and four cluster centres randomly chosen from the data set have indices 2, 5, 7, and 8 respectively, then the chromosome Ch_i is 0100101100. In GCUK-binary, no genes are left unassigned and therefore the # symbol is not needed.

4.2.3 Fitness Function

GCUK is a single objective optimization method that use only one fitness function. Bandyopadhyay and Maulik [2002b], Lin et al. [2005] used the DB index [Davies and Bouldin, 1979] as their fitness function to measure the validity of the clustering algorithm. This index measured the similarity between the clusters (how separated and compact are the clusters). The lowest values of the index indicate the better clustering and how well the clusters are separated. For these reasons, this study uses the DB index.

The DB index measures the similarity of the clusters by calculating the function of the ratio of the sum of *within cluster scatter* to the *between cluster separation*. The scatter within the C_i for the i^{th} can be computed as:

$$S_{i,q} = (\frac{1}{|C_i|} \sum_{x \in C_i} \| x - c_i \|_2^q)^{1/q} \tag{4.1}$$

where c_i is the centre of the cluster C_i. Usually the choice of q is 2, where a Euclidean distance measured between the centre of the cluster and the individual data points. The $R_{i,qt}$ in Equation 4.2 denotes the similarity of C_i to the other clusters. In this study, the Hamming distance is denotes in $d_{ij,t} = d(C_i, C_j)$.

$$R_{i,qt} = Max_{j,j\neq i}\{\frac{S_{i,q} + S_{j,q}}{d_{ij,t}}\} \tag{4.2}$$

where $d_{ij,t} = d(C_i, C_j) = \parallel c_i - c_j \parallel_t$

Then, the fitness value can be evaluate by:

$$DB_i = \frac{1}{k_i}\sum_{i=1}^{k_i} R_{i,qt} \tag{4.3}$$

From Equation 4.1, the measure of dispersion of a cluster $C_i, i = 1, \ldots, K_i$ is represented by $S_{i,q}$. The evaluation of the DB index for the chromosome Ch_i is defined in Equation 4.3 where the lowest value is indicated the better clustering.

The following operators are the essential components of GA as implemented in the GCUK-binary to deal with binary data.

4.2.4 Selection Operator

Conventional proportion or Roulette Wheel Selection (RWS) was used in the GCUK-binary which is same as in GCUK. The largest portion on the wheel represents the largest fitness value. The wheel will be spun N times and the pointer will stop at any portion that is assigned to the chromosomes. The largest portion will have the highest chances to be selected more than once compared to the smaller portions. Next, the selected chromosomes replace the whole population P and became known as P' (new population). It has to be noticed that the chromosome may be chosen more than once if its fitness value is large or it may not be chosen at all (very small portion).

4.2.5 Crossover Operator

The crossover operator is used to exchange parts of two chromosomes to explore further regions of the solution space and possibly to obtain better chromosomes. In this study, a single point crossover was used, following Bandyopadhyay and Maulik [2002b] in the original GCUK. In order to get the best results, several rounds of testing were done and each of the data sets had a different probability of p_c.

Parents	1	0	1	0	1	$\|\|$	0	0	0	0	1	Offspring	1	0	1	0	1	$\|\|$	1	1	0	1	1
	0	1	0	0	0	$\|\|$	1	1	0	1	0		1	0	0	0	0	$\|\|$	0	0	0	1	

For example, two chromosomes will exchange all the alleles at their 6^{th} genes. Thus, all the alleles after 5^{th} change each other and produce two new offspring.

4.2.6 Mutation Operator

Each and every centre coordinate mutates with probability p_m. Let v denote a coordinate of the chormosome. Then, if a mutation occurs, v becomes either $v \pm 2\delta v$ if $v \neq 0$ or $\pm 2\delta$ if $v = 0$. δ is a number generated from the uniform distribution in [0,1]. The + or − sign occurs with equal probability. If the result of the mutation exceeds the extremes of the [0,1] interval it is reset either to 0 if less than zero or to one if greater than 1. Here, the uniform random mutation in which each gene in each chromosome is flipped with a pre-specified probability,

p_m. Then, new population P'' entirely replace the past population P'.

4.2.7 Termination Process

The process of the GCUK-binary is executed after achieving the maximum number of iterations. The final results represent the final generation with the smallest fitness value. However, investigation regarding how many generations the GA has to run to produce an optimal or at least near optimal solution has not been fully conclusive till now. Research by Safe et al. [2004] suggested that the iterations can be stopped when the upper limit on the number of generations has been reached . The process can be used when the chance of the algorithm achieving significant changes in the next generations are excessively low .
In this study, the parameter settings, including the maximum generations needed, were set according to the power of the computing engine and the time that was thought of to be reasonable for the algorithm to run.

All the computer programs for this study were written in MATLAB software R2015a 64-bit on a Windows 7 platform.

4.3 Application to Real World Data Sets

The original data sets are from the categorical data which have been changed into the binary form. Three different real life binary data sets were used to show the effectiveness of the GCUK-binary. These data were the Nursery [Lichman, 2013], the Car Evaluation [Lichman, 2013] and the Bank Marketing [Sérgio Moro and Paulo Cortez and Paulo Rita, 2014] data sets. For the imbalanced class data sets: the CRTA, and URTA, were taken from the New Zealand Transport Agency (NZTA) and UK Department for Transport website respectively. CRTA2 is an extracted data from the CRTA data set. All the explanations about these data sets are given below:

Nursery data: Nursery data is a categorical data set that was changed into binary data. The original data had $n = 12960$ observations with eight groups of categorical variables explaining the criteria for choosing the children before they can enter primary school. These eight groups of variables represent the parents' occupation, the history of the nursery, completeness of the form, number of children, housing condition, family financial standing, family social environment and children's health condition. Each of these criteria that are contained in these eight groups were the variables to be clustered and there was a total of 27 variables for this data.

Car Evaluation data: This data set contains information about the performance evaluation of the cars. There were six groups of variables that represent the car's price, the car's maintenance, the number of doors, the number of passengers, the size of luggage boot and the car safety rating. The criteria for these groups include the nominal attributes such as very high, high, medium and low. All these 21 criteria were the variables to be clustered and there was a total of 1728 observations in this data.

Bank Marketing data: This data set has 45211 observations about the marketing campaign done by the Portuguese Banking Institution. Phone calls were made more than once to the same clients to find out whether they were interested in subscribing to their bank's term deposit campaign. There were 17 groups of the characteristics of the clients in categorical and numerical form. However, only 46 categorical data were selected as the variables for this study. The group variables recorded bank client data such as their job, marital status, education, personal loan and housing loan.

Christchurch Road Traffic Accidents data (CRTA): This data represented the road traffic accidents in Christchurch from year 2000 - 2009. The data set had 26440 accidents (n) with 50 identified causes of road accidents provided by New Zealand Transport Agency (NZTA). The raw data was an accidents form data explaining how the accidents happened. The distribution of the types of injuries were; fatality (5%), serious (5.9%), minor (25%) and non-injured (67.6%).

United Kingdom Road Traffic Accidents, (URTA): This data is similar to the CRTA data, which is a record of road traffic accidents in United Kingdom (UK). The total accidents for 2014 in UK was 146323, however because of the limitation of the capability of the computing engine used in this research, only the records for December were used. Thus only $n = 11,652$ were used which was from the December's record, with three level of injuries; fatality (1.49%), serious (13.32% and minor / non-injured (85.19%).

For the types of injuries, if two or more injuries had been found for the same accident, the worst one was chosen. For example two persons, **A** and **B** were involved in an accident, with minor and serious injuries respectively. Then, the serious injury was chosen as this was the worst one for that accident's case. Both data sets (CRTA and URTA) had an imbalanced class problem where the class are not evenly distributed. Usually the intention of the researchers is to look at the impacts from the minority attributes to the whole data set rather than the majority instances. This makes the regular clustering algorithms unable to simply be applied to this data as it will treat all the instances with the same degree of importance. This could cause the misclassification results where some of the important variables were placed into the wrong group. The regular classification can handle this issue by assigning a weight or a cost to the classification of object i to clustering.

To see the difference results, the majority instances (minor and non-injured) were excluded from the CRTA data. This data was referred as CRTA2.

4.4 Results

In this section, the results are divided into three subsections. The first subsection is about the results of finding the number of clusters, K. The second subsection is the parameter settings for each of the data sets. The last subsection is the results of GCUK-binary. The first process of GCUK-binary was to chose the best number of K for each of the data sets.

4.4.1 Number of clusters, K

The GCUK algorithm has an ability to give automatically the number of clusters K. At first, the original GCUK algorithm was tested with the CRTA data to test the effectiveness of this algorithm to find the number of clusters for this data set.

For the encoding, any possible solution is coded as a string of k_{max} characters. These characters are follow the original GCUK where it can use either coordinates of the cluster centres; $v^i = (v_1^{(i)}, \ldots, v_p^{(i)})'$ and $i = 1, \ldots, k$ or the special character # (do not care). Each vector $v^{(i)}$ represents a cluster while # represents an empty cluster. The GCUK algorithm was run by adopting the parameters size of population, $s = 30, p_c = 0.8, p_m = 0.001$ and assumed $k_{min} = 2$ and $k_{max} = 9$. The results are shown below:

Table 4.1: Clusters of variables obtained from the GCUK algorithm

Cluster	**Factors**
1	Failed to give way
2	**Empty cluster**
3	Alcohol or drugs
4	Inattentive: Failed to notice
5	Other factors
6	Did not see until too late
7	Illness and disability
8	Speed more than 60 km/h
9	Sudden action

It seems very interesting that the clustering procedure provides us with 7 singletons (clusters with only one element) because the variables included in each of them are naturally the most relevant in connection with a major causes of the road accidents. The rest of the other factors of road traffic accidents are put into the same cluster, Ch_5. While for the Ch_2 there is an empty cluster. It has to be noticed that there are 50 factors of road traffic accident.

This is the other reason why in this study, the number of clusters K had to be controlled and determined by using the Silhouette index. From this experiment, the description of the encoding from the original GCUK was modified where the chromosomes are all filled with the cluster centres and there is no use of the special character of # or " do not care" symbol allowed in the chromosomes.

Table 4.2 below shows the average values of the Silhouette index for Nursery, Car Evaluation, Bank Marketing, CRTA, CRTA2 and URTA data sets to find the best number of clusters, K.

Table 4.2: Silhouette index for GCUK-binary for all the data sets

Nursery	Car	Bank	CRTA	CRTA2	URTA
$k = 4, s_i = 0.0678$	$k = 3, s_i = 0.0465$	$k = 3, s_i = 0.4260$	$k = 5, s_i = 0.5354$	$k = 4, s_i = 0.5830$	$k = 5, s_i = 0.8229$
$k = 5, s_i = 0.1296$	$k = 4, s_i = 0.0299$	$k = 4, s_i = 0.4752$	$k = 6, s_i = 0.6697$	$k = 5, s_i = 0.6626$	$k = 6, s_i = 0.8379$
$k = 6, s_i = 0.0921$	$k = 5, s_i = 0.0079$	$k = 7, s_i = 0.6117$	$k = 5, s_i = 0.3625$	$k = 6, s_i = 0.5830$	$k = 7, s_i = 0.7514$

From Table 4.2, the Silhouette index suggested that for the nursery data set, the best choices to find K were when $k = 4, s_i = 0.0678$, $k = 5, s_i = 0.1296$ or $k = 6, s_i = 0.0921$. These results were supported by graphical figures (see Figure 4.1, Figure 4.2 and Figure 4.3). It showed that the best choice was when $K = 5$ as it gave the highest index value, $s_i = 0.1296$.

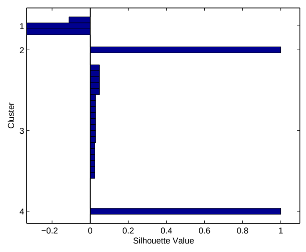


Figure 4.1: Silhouette index for Nursery data, k=4

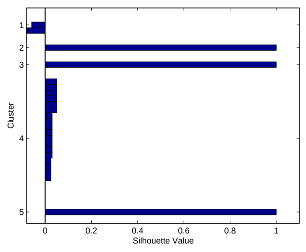


Figure 4.2: Silhouette index for Nursery data, k=5

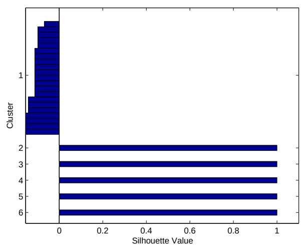


Figure 4.3: Silhouette index for Nursery data, k=6

The best choice for the number of clusters for the Car Evaluation data was $k = 3$ with the value of $s_i = 0.0465$, where as the other options were when $k = 4, s_i = 0.0299$ and $k = 5, s_i = 0.0079$. The graphical results for car evaluation are shown in Figure 4.4, Figure 4.5 and Figure 4.6.

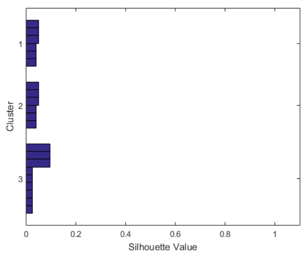


Figure 4.4: Silhouette index for Car Evaluation data, k=3

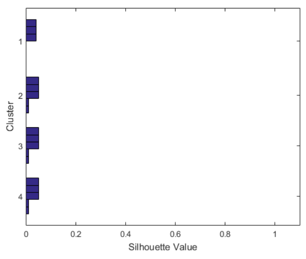


Figure 4.5: Silhouette index for Car Evaluation data, k=4

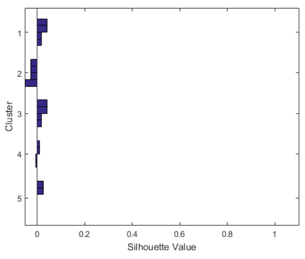


Figure 4.6: Silhouette index for Car Evaluation data, k=5

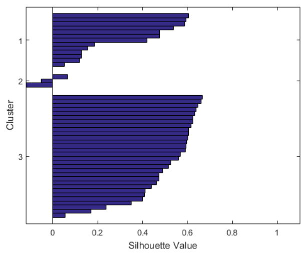


Figure 4.7: Silhouette index for Bank Marketing data, k=3

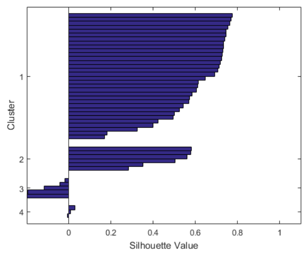


Figure 4.8: Silhouette index for Bank Marketing data, k=4

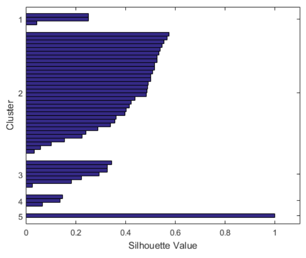


Figure 4.9: Silhouette index for Bank Marketing data, k=5

The best choice K for Bank Marketing data set was $k = 5$ with the average index value is $s_i = 0.6116$. The graphical results for this data set are show in Figure 4.7, Figure 4.8 and Figure 4.9.

The suitable number of clusters for CRTA was $k = 6$ where it gave the highest value of the Silhouette index; $s_i = 0.6697$. Since the average value of $k = 7$ ($s_i = 0.6117$) was lower than the value of $k = 6$, the process was stopped and by assuming that the average value for $k = 8$ had a lower value than $k = 7$. The graphical results for CRTA data set are show in Figure 4.10, Figure 4.11 and Figure 4.12.

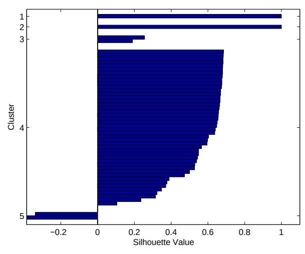


Figure 4.10: Silhouette index for CRTA data, k=5

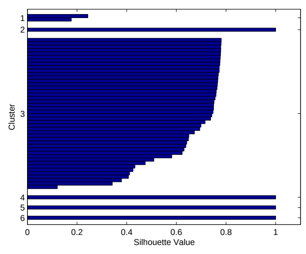


Figure 4.11: Silhouette index for CRTA data, k=6

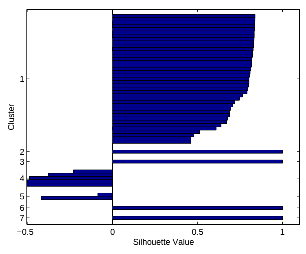


Figure 4.12: Silhouette index for CRTA data, k=7

While for the CRTA2, the number of clusters was different from the CRTA as the majority instances (minor injury and non-injury data points) were excluded from the original data set. It suggested that for CRTA2, the $k = 5$ was the best choice as the average Silhouette index is 0.6626 which was the highest compared to $k = 4$ and $k = 6$.

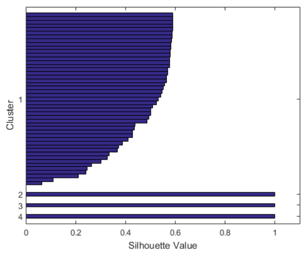


Figure 4.13: Silhouette index for CRTA2 data, k=4

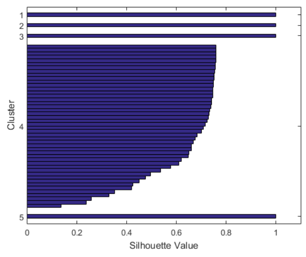


Figure 4.14: Silhouette index for CRTA2 data, k=5

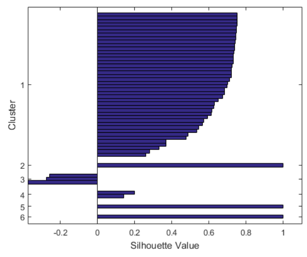


Figure 4.15: Silhouette index for CRTA2 data, k=6

The graphical results for URTA data set were as in Figure 4.16, Figure 4.17 and Figure 4.18 which show that there is a misclassification results when $k = 5$ and $k = 7$. This indicated that the best choice for URTA data was when $k = 6$ with the Silhouette index $s_i = 0.8379$.

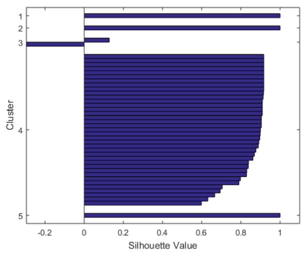


Figure 4.16: Silhouette index for URTA data, k=5

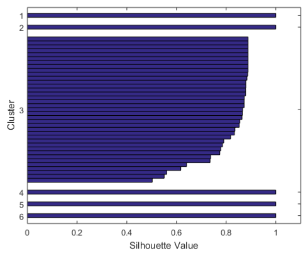


Figure 4.17: Silhouette index for URTA data, k=6

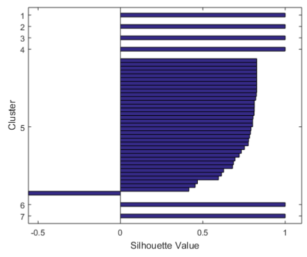


Figure 4.18: Silhouette index for URTA data, k=7

After the number of clusters, K, for each of the data sets were selected, the next process is to set the parameter settings. It has to be notice that GA was built with the different number of generation, number of population, probability crossover and mutation to produce a good population.

4.4.2 Parameter Settings

Due to the differences of the total observations for these data sets, the parameter settings for each of the parameter settings are different too. The following parameters were used in the GCUK-binary:

- G: ***number of generations***, which is to show how many iterations that need to run the algorithm before the process terminated.
- P: ***number of populations that are needed***
- p_c: ***probability of crossover***
- p_m: ***probability of mutation***

The algorithm needs to run several times before the final number of generations can be determined. The lowest value of DB index indicated the best and final result. However, this study did not focus on the natural clusters as the objective was to check the performance of the GCUK-binary to cluster the binary data and compared with the K-means algorithm.

The results of the different parameter settings are shows in Table 4.3, Table 4.4, Table 4.5, Table 4.6, Table 4.7 and Table 4.8.

Table 4.3: Different parameter settings and the DB value for Nursery data

Parameter settings	**DB**
$G = 50, P = 30, p_c = 0.8, p_m = 0.01$	0.2966
$G = 50, P = 50, p_c = 0.8, p_m = 0.01$	0.3079
$G = 70, P = 50, p_c = 0.9, p_m = 0.01$	0.3300
$G = 100, P = 50, p_c = 0.8, p_m = 0.01$	**0.2964**

Table 4.4: Different parameter settings and the DB value for Car Evaluation data

Parameter settings	**DB**
$G = 50, P = 30, p_c = 0.8, p_m = 0.01$	0.4373
$G = 50, P = 50, p_c = 0.8, p_m = 0.01$	1.0458
$G = 50, P = 30, p_c = 0.9, p_m = 0.01$	0.5753
$G = 50, P = 30, p_c = 0.9, p_m = 0.02$	**0.3565**

Table 4.5: Different parameter settings and the DB value for Bank Marketing data

Parameter settings	**DB**
$G = 50, P = 10, p_c = 0.8, p_m = 0.02$	0.6797
$G = 50, P = 10, p_c = 0.8, p_m = 0.01$	0.7327
$G = 50, P = 10, p_c = 0.9, p_m = 0.01$	0.5466
$G = 50, P = 10, p_c = 0.9, p_m = 0.02$	**0.3937**

After several round of testing, the best choice parameters for the Nursery data were $G = 100, P = 50, p_c = 0.8, p_m = 0.01$ and $DB = 0.2964$, which was the lowest index value among the others. For Car Evaluation data set, the best parameter settings were $G = 50, P = 50, p_c = 0.9, p_m = 0.02$ and gave the value of DB is equal to 0.3565. The best parameter settings for Bank Marketing data set were when $G = 50, P = 10, p_c = 0.9$ and $p_m = 0.02$ with the value of the DB index equal to 0.3937.

As mentioned before, CRTA data set was special data due to the imbalanced issue. For this data, first the whole data set was used and the reason was to see the performance of the GCUK-binary with the issue of imbalanced class. Then CRTA2 data was used to look at the different results (if any) as CRTA2 do not have this problem. The results are shows in Table 4.6 for full data set (CRTA) and Table 4.7 is for the fatality and major injury cases (CRTA2).

Table 4.6: Different parameter settings and the DB value for CRTA

Parameter settings	**DB**
$G = 50, P = 30, p_c = 0.8, p_m = 0.01$	0.9371
$G = 50, P = 30, p_c = 0.9, p_m = 0.01$	0.8398
$G = 50, P = 10, p_c = 0.9, p_m = 0.01$	**0.3838**

Table 4.7: Different parameter settings and the DB value for CRTA2

Parameter settings	**DB**
$G = 20, P = 10, p_c = 0.9, p_m = 0.01$	0.8897
$G = 20, P = 10, p_c = 0.8, p_m = 0.02$	**0.3580**
$G = 30, P = 10, p_c = 0.9, p_m = 0.01$	1.1903

From the results in Table 4.6, the lowest value of the DB index for CRTA data set was from the parameter settings of $G = 50, P = 10, p_c = 0.9, p_m = 0.01$ with DB is equal to 0.3838. While for CRTA2 the parameter settings were $G = 20, P = 10, p_c = 0.8, p_m = 0.02$ and the DB index value is 0.3580.

The parameter settings the other imbalanced class data set: URTA, is show in Table 4.8 below. For this study, the parameter settings of $G = 50, P = 10, p_c = 0.8, p_m = 0.02$ was chosen as it gave the lowest fitness value ($DB = 0.0850$).

Table 4.8: Different parameter settings and the DB value for URTA

Parameter settings	**DB**
$G = 30, P = 10, p_c = 0.9, p_m = 0.01$	0.6718
$G = 30, P = 10, p_c = 0.9, p_m = 0.02$	0.1220
$G = 50, P = 10, p_c = 0.9, p_m = 0.01$	**0.0850**
$G = 50, P = 10, p_c = 0.8, p_m = 0.02$	0.9099

By assuming that all these index values were enough to compete with K-means, the process of searching the parameter settings was stopped due to the power of the computing engine to do the other iterations.

4.4.3 GCUK-binary Results

After the process of searching the number of clusters, K and the parameter settings for each of the data sets, the comparison with K-means algorithm was done. To check the performance of GCUK-binary, the K-means algorithm was run with the same K and compared the DB index value.

The results of *DB* index values for GCUK-binary for all the data sets are shown in Table 4.9 below.

Table 4.9: Result for GCUK-binary

Data set	**GCUK-binary**	**K-means**
Nursery	0.2964	1.5870
Car Evaluation	0.3565	1.8160
Bank Marketing	0.3937	1.4913
CRTA	0.3838	0.8524
CRTA2	0.3580	1.2019
URTA	0.0850	1.8864

From Table 4.9, it shows that the lowest value for all the data sets were attained by GCUK-binary. It proved that the GCUK-binary is competent to be an efficient and effective clustering algorithm. It also showed that this algorithm gave good performance when clustered the imbalanced class data sets.

4.4.4 Summary

In this study, the well-known GA based clustering for single objective function was studied and implemented to cluster the binary and imbalanced class data sets. Even though that the nature of the GA was using the binary chromosomes, there are only a few researchers specifically used it with these data sets.

The word "GCUK-binary" was used to differentiate it from the original GCUK and even though there is only some improvements were added, the GCUK-binary still have a contributions in this study.

In this study, the center-based encoding was used where it can hasten the converging time compared with the original GCUK. This chromosome length has a faster convergence rate [Maulik et al., 2011] compared to the point-based encoding that was used in the original GCUK. Further, the use of binary data point in this study also make it easier to calculate the centre as the centre can taken directly from the data points themselves. In this study, Hamming distance was used to denote the difference between two binary data points.

For the first experiment with GCUK-binary, the coordinates of the cluster centre can be filled with the special character of # or "do not care" symbol similar as the original GCUK. The number of the clusters, K also was automatically defined by the algorithm. The implementation showed that GCUK gave a maximum number of cluster with one empty cluster. Thus for this study, the # symbol was not allowed and the number of K was held at $K_{min} = K_{max}$. This was to avoid any empty cluster results. To validate the choice of the K, the Silhouette index was used to produce coherent results.

The other contribution is this efficiency of the single objective optimization GA clustering to cluster the imbalanced class issue. The standard clustering algorithm can not be simply applied to this data as it can cause misclassification. The results showed that the lowest index value was attained by GCUK-binary. It means that the implementation of GCUK-binary is successful and efficient compared to the K-means.

The objective function for this study was to see the performance of the single objective optimization GA clustering when clustering binary data and imbalanced class data sets. All the contributions have made GCUK more efficient when dealing with both data and it confirmed that this algorithm can be further explored to deal with these data sets with added the missing data and the outliers.

Chapter 5

Genetic Algorithm with Incremental K-Means

5.1 Introduction

In the previous chapter; the results and findings from the GCUK-binary showed the capabilities and the effectiveness of this single objective of GA to cluster binary and imbalanced class data sets. This method attained the lowest value of the fitness function and outperformed the K-means algorithm.

In the previous chapter (1,2,3) the literature on K-means improvement and GA based clustering research was discussed thoroughly. The Incremental K-Means method proposed by Ordonez [2003] had been chosen in this study due to its capability to speed up the calculations and accelerate convergence to the solutions.

Thus, in this study, the new proposed clustering algorithm was proposed by combining the simple GA with the IKM which was then referred to as GAIKM ([Saharan and Baragona, 2013]). The combination of this algorithm and GA was believed to enhance the effectiveness and efficiency of the algorithms to cluster the binary and imbalanced class data sets.

This chapter is divided into five sections: introduction of the GAIKM, the procedure of GAIKM, a simulation study, application to real world data sets, results and summary.

5.1.1 Incremental K-Means, IKM

The IKM proposed by Ordonez [2003] specifically deals with large binary data sets. This method may be viewed as a compromise method between the online K-means and the plain K-means algorithms. However, unlike these two methods, IKM will not iterate until convergence and only the values of the sufficient statistics N and M have to be updated at each step. Let n be the number of cases, which ordinarily is the number of steps of the algorithm as well, and k be the number of clusters. Then $N_j, j = 1, \ldots, k$ is the number of cases that are assigned to cluster j and M_{ij} is the sum of the instances equal to 1 recorded for the factor i for all the cases that are assigned to cluster j. The centres $C = \{C_j, j = 1, \ldots, k\}$, the variance $R = \{R_j, j = 1, \ldots, k\}$ of each factor in each cluster, where $R_j = (v_1 j, \ldots, v_d j)'$, and the centre weights $W = \{W_j, j = 1, \ldots, k\}$ need to be updated only every n/m steps, i.e., only if n/m is an integer. Namely, m is a parameter that controls the updating frequency. Ordonez [2003] suggested on empirical basis that the value of $m = \sqrt{n}$ is likely to ensure the best results in most problems encountered in practice. Such choice, however, is viable only if a batch computation where the number of records in the database is known in advance to be equal to n. For the complexity computation, the variants of k-means designed for clustering binary data require a computation time at least of order approximately $O(Tkn)$, where T is the average over all cases of how many factors out of d have recorded value 1. The space required for storing the matrices M and C is of order $O(dk)$ while it is $O(k)$ for the vector N and W. The space for R can be saved because it can be straightforwardly computed from C. However, an additional space is required to hold temporarily a certain number of cases in a buffer.

5.1.2 Data Representation

Let D represent a set of clustering data organized as $n \times d$ matrix of binary attributes, where n is the number of cases and d is the numbers of variables. If $t_i; i = 1, \ldots, n$ is assumed as a d-dimensional column vector whose elements are the factor values recorded at the occurrence of the case i, then

$$\mathcal{D} = \begin{pmatrix} t_1' \\ t_2' \\ \vdots \\ t_n' \end{pmatrix}, \quad t_i = (a_{i1}, a_{i2} \ldots, a_{id})', \quad a_{il} \in \{0, 1\}.$$

where $'$ means a transpose of a vector or matrix. An example of such a matrix is as follows:

$$\mathcal{D} = \begin{pmatrix} 0 & 1 & 1 & \ldots & 1 & 1 & 1 \\ 1 & 0 & 0 & \ldots & 1 & 1 & 0 \\ & & & \ldots & & & \\ 0 & 1 & 0 & \ldots & 1 & 0 & 1 \end{pmatrix},$$

The number at the intersection of row (case) i and column (factor) l is 1, if the factor l has been recorded concomitant to this factor i and 0 otherwise.

5.1.3 Fitness Function

[Ordonez, 2003] suggested using the Euclidean distance in his proposed IKM. In this study, the Hamming distance was used, i.e., $\delta(t_i, C_j)$ is set equal to the number of coordinates in which the vectors t_i and C_j differ.

The fitness function for GAIKM is to minimize the sum of the intra-cluster variances, for which only the sufficient statistics $N_j = \sum_{t_i \in C_j} 1$ i.e., the number of cases in cluster $j, j = 1, \ldots, k$, and M_j, is the vector whose entries are the sums of 1 for each factor of the data points in cluster C_j, are needed. For each cluster the variance of the proportion $C_j = M_j/N_j$ is equal to $R_j = C_j(1 - C_j)$, and the intra-cluster variance can be written as follows:

$$q(R, W) = \sum_{j=1}^{k} W_j \sum_{l=1}^{d} R_{lj} \tag{5.1}$$

where $W_j = N_j/n$. Equation 5.1 was the objective function in this study which the minimum value of the qRW indicated the better result.

5.1.4 Selection Process for GAIKM

The Roulette Wheel Selection (RWS) was used in GCUK-binary as the selection method, while in NSGAII-binary, the binary tournament technique was used. Here in the GAIKM, the stochastic universal sampling (SUS) is chosen as this method was shown to be unbiased and has a minimum spread compared with RWS [Baker, 1987]. Let $f_1, \ldots, f_s$ be the fitness function values computed for each chromosome in the population $\mathcal{P}$. Then define $f_i^* = f_i/(f_1 + \ldots + f_s)$. In a segment of length 1 each chromosome i is assigned a subsegment of length f_i^*. These subsegments are non-overlapping and their sequence fits the interval [0,1] exactly. A single uniform random number in [0,1] is generated which serves as starting point for choosing the first chromosome. The subsequent chromosomes are chosen by pointing at evenly spaced intervals to the subsegments assigned to the chromosomes. It should be noted that a chromosome may be chosen more than once if its fitness is large or conversely may not be chosen at all. The selected chromosomes replace the whole population $\mathcal{P}$ and we let $\mathcal{P}'$ denote the new population.

5.1.5 Crossover Process for GAIKM

The one point crossover is applied in the present algorithm, and two chromosomes (parents) chosen uniformly at random are assumed to produce two chromosomes (children) that replace their parents. The new population $\mathcal{P}''$ has the same size s. The parents have a pre-specified p_c (the crossover probability) to produce new chromosomes, otherwise they are directly included in the new population $\mathcal{P}''$. The one point crossover requires that a cutting point q be chosen uniformly at random in $[1, \ell - 1]$, where $\ell = dk$ is the chromosome length. Then, the genes from $q + 1$ to ℓ from the first chromosome replace the corresponding genes in the second chromosome and yields the first child. At the same time the genes of the second chromosome from $q + 1$ to ℓ go to substitute the corresponding genes in the first chromosome so forming the second child. For example, let $k = 3$ and $d = 6$ and the following two chromosomes in $\mathcal{P}'$ undergo the crossover with cutting point $q = 12$:

Parents (P):

0 1 1 0 1 1 1 1 1 0 1 1 | 0 1 1 1 1 1

1 0 1 1 0 0 0 1 0 0 0 0 | 1 0 0 0 1 0

Offsprings (P'):

0 1 1 0 1 1 1 1 1 0 1 1 | 1 0 0 0 1 0

1 0 1 1 0 0 0 1 0 0 0 0 | 0 1 1 1 1 1

5.1.6 Mutation Process for GAIKM

The mutation operator applies to the population $\mathcal{P}''$. It aims specifically at maintaining the diversity in the population and possibly at improving the chromosomes. In the present algorithm, uniform random mutation is used, i.e., each gene in each chromosome in the population flips with a pre specified probability p_m (the mutation probability). The new population $\mathcal{P}'''$ entirely replaces the past population $\mathcal{P}''$. Then $\mathcal{P}''' \rightarrow \mathcal{P}$ and the algorithm repeats its main cycle.

In GA Toolbox manual, the default for the mutation value is $P_m = 0.7/Lind$ where $Lind$ is the length of an individual. However, in GAIKM, the algorithm was run for several times with a different mutation value that ranged from 0.7 to 0.9. This is to make sure that the best choices were selected to be the population. It has to be noted that the probability of mutation should not be too small or too large to maintain the good diversity in the population.

5.1.7 Termination Criteria

For the termination criteria, the number of generation G was set according to the power of the computing engine and the time that was thought of to be reasonable for the algorithm to run. Each of the data sets had a different number of generations as each had a different number of observations.

5.1.8 The Process of GAIKM

One of the objectives in this study was to examine the effectiveness of applying the GA method to binary and imbalanced class data sets. The next step was to cluster them by using methods taken from IKM.

The procedure of the GAIKM was simple. First, s sets of cluster centres were generated at random and were encoded as a population of s chromosomes. Then, the stream of the cases $t_1, t_2, \ldots, t_n$ was inserted to the algorithm in a groups of h cases at a time. If $h = 1$ was chosen, then each case was input to the algorithm one at a time in the sequence. If $h = n$, then a batch execution was performed. These h cases were assigned to the closest centre using the distance $\delta(t_i, C_j)$. In GAIKM, the distance of the data points, t_i to the closest centre was calculated by using the Hamming distance which is same as in GCUK-binary.

However, unlike the IKM, in the GAIKM each case had a chance to be assigned in s different cluster systems. This makes GA differ from the other clustering methods that consider only a single one as IKM. As the new cases were input into the algorithm, the steps of the genetic algorithm were developed according to the three operators of selection, crossover and mutation processes.

Then a new population of s chromosomes, each one corresponding to a partition of the data points in the current database, replaced the old population. At the end, s different partitions of all n data points in the population were available. The pseudo-code for GAIKM is in Appendix A.

Compared with the proposed algorithm by Saha et al. [2015], GAIKM is using the effectiveness and efficiency of Incremental K-means in the clustering steps. The set for the population was selected and its fitness function tested by using the process of GAs. This differs from Saha et al. [2015], who used Random Forest in an incremental way to classify the semi and pure rough points.

5.2 A Simulation Study

Artificial binary data set were generated according to three models. For this simulation study all data sets have $n = 1000$ cases, on $d = 9$ variables, and a structure of $k = 5$ clusters.

The first one was a hand-made data set where each cluster was associated to one or more variables so that there was no overlapping between variables and clusters. For example, let cluster 1 be characterized by variables 1 and 7, then all records in cluster 1 will have 1 in fields 1 and 7 and zero elsewhere. The chosen “cluster prototypes” are displayed in Table 5.1.

Table 5.1: Cluster prototypes for the "hand-made" clustering structure of the first data set

Variables \ Cluster	1	2	3	4	5	6	7	8	9
1	1	0	0	0	0	0	1	0	0
2	0	1	1	0	1	0	0	1	0
3	0	0	0	1	0	1	0	0	0
4	0	0	0	0	0	0	0	0	1
5	0	1	0	1	0	0	1	0	1

The records displayed in Table 5.1 are replicated n_1 times to form cluster 1, n_2 times to form cluster 2, ... and n_k times to form cluster k. The integers n_j may be any number, with the obvious constraints that $n_j > 0 \quad \forall j$ and $\sum_{j=1}^{k} n_j = n$. The data set is made more realistic by adding a 25% noise to the data. That is 25% of the data were chosen uniformly random and their values flipped from 0 to 1 and vice versa.

Ten data sets were generated according to the preceding procedure and on each of them the algorithms GAIKM, IKM, Scalable K-means (SKM), GCUK and NSGA-II were run with the different size of clusters, n: $n_1 = 350$, $n_2 = 50$, $n_3 = 200$, $n_4 = 150$ and $n_5 = 250$. The Hamming distance was used for all algorithms by modifying, as a consequence, the objective function (fitness function in the GA framework) accordingly. All algorithms were intended to minimize the within-cluster sum of distances.

Let this experiment be called Experiment 1. The results are displayed in Table 5.2. The performance of each of the five algorithms were evaluated using 5 cluster external validity indexes, i.e. the misclassification rate, the corrected Rand index, the Jaccard index, the efficiency index and the purity index, and 2 cluster internal validity indexes, the Silhouette index and the within-cluster sum of distances.

The average and the standard deviation of each index computed across ten replications was reported. The results showed that GAIKM attained the lowest index value of the within-cluster sum of distances compared to standard deviation of the other clustering algorithms. It also reported that the misclassification rate for GAIKM was the lowest which indicates a good performance.

Table 5.2: Results from Experiment 1

Index \ Algorithm	GAIKM	IKM	SKM	GCUK	NSGA-II
misclassif.	0.01	0.18	0.15	0.23	0.06
	(0.03)	(0.07)	(0.06)	(0.07)	(0.21)
Rand	0.98	0.80	0.80	0.70	0.90
	(0.05)	(0.10)	(0.11)	(0.12)	(0.32)
Jaccard	0.98	0.75	0.76	0.66	0.92
	(0.07)	(0.10)	(0.12)	(0.11)	(0.24)
efficiency	0.99	1.00	1.00	0.99	1.00
	(0.04)	(0.00)	(0.00)	(0.01)	(0.00)
purity	0.99	0.75	0.76	0.66	0.92
	(0.04)	(0.10)	(0.12)	(0.11)	(0.24)
silhouette	0.92	0.77	0.80	0.73	0.74
	(0.05)	(0.10)	(0.08)	(0.07)	(0.07)
within-sum	0.32	0.538	0.34	0.47	0.34
	(0.054)	(0.181)	(0.144)	(0.177)	(0.164)

The standard deviations are enclosed in parentheses

The second artificial data set was generated by using the Matlab file *binornd* for the simulation of Bernoulli random numbers. Let this be called Experiment 2. As before, clusters were built by assuming that certain variables had more chance to take the value 1 than others. In general variables have little chance to take the value 1 in a cluster except for one or two variables which are given rather high probability to take the value 1. Table 5.3 reports for each of 5 clusters the probabilities that the variables take the value 1. The size of the clusters were chosen as in Experiment 1. For instance, in cluster one the column 2 was obtained as a sequence of Bernoulli variables with probability p_{11}.

Table 5.3: Probabilities that variables assume value 1 according to a Bernoulli scheme

Cluster \ Variables	1	2	3	4	5	6	7	8	9
1	0.85	0.01	0.01	0.01	0.01	0.01	0.75	0.01	0.01
2	0.01	0.75	0.01	0.01	0.80	0.01	0.01	0.01	0.01
3	0.01	0.01	0.01	0.90	0.01	0.75	0.01	0.01	0.01
4	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.95
5	0.01	0.01	0.75	0.01	0.01	0.01	0.01	0.80	0.01

The five algorithms were run on ten artificial data sets generated in Experiment 2. The same indexes were computed as in Experiment 1 and are reported in Table 5.4.

Table 5.4: Results from Experiment 2

Index \ Algorithm	GAIKM	IKM	SKM	GCUK	NSGA-II
misclassif.	0.09	0.17	0.21	0.19	0.12
	(0.03)	(0.07)	(0.14)	(0.06)	(0.04)
Rand	0.84	0.74	0.66	0.65	0.80
	(0.06)	(0.11)	(0.22)	(0.10)	(0.06)
Jaccard	0.79	0.70	0.62	0.60	0.74
	(0.07)	(0.11)	(0.19)	(0.09)	(0.07)
efficiency	0.83	0.92	0.79	0.78	0.80
	(0.06)	(0.03)	(0.11)	(0.06)	(0.06)
purity	0.93	0.75	0.74	0.72	0.91
	(0.02)	(0.12)	(0.19)	(0.10)	(0.03)
silhouette	0.69	0.61	0.56	0.54	0.67
	(0.02)	(0.08)	(0.15)	(0.07)	(0.03)
within-sum	0.32	0.63	0.44	0.45	0.32925
	(0.008)	(0.122)	(0.177)	(0.076)	(0.022)

The standard deviations are enclosed in parentheses

Experiment 3 is concerned with data sets with cluster structure generated by correlation of groups of variables. The data in a cluster are characterized by the fact that some variables are strictly correlated while other variables are uncorrelated each other and with respect to the correlated variables. The method followed, was proposed by Leisch et al. [1998] and may be run by using the R file *bindata*. A Matlab version of the R code was used in Experiment

3. Sets of variables may overlap, i.e. a variable which is correlated with other ones in a cluster may be correlated as well in another cluster, but with different variables. Table 5.5 reports for each cluster the variables which are correlated in such a cluster. The correlation coefficient has been chosen the same in each cluster, so each group of variables is followed by a single correlation coefficient.

Table 5.5: Correlated variables in each cluster of each data set for Experiment 3

Cluster	Correlated variables	Correlation Coefficient
1	4 5 6	0.95
2	1 7 8	0.87
3	2 4 7 9	0.97
4	2 3 8 9	0.89
5	1 3 5 8	0.91

The results from Experiment 3 are displayed in Table 5.6. In this case the algorithm NSGA-II gives results very close to those provided by GAIKM. It may be argued that the correlation structure may be correctly recovered by a multi-objective GA as well as a combination of GA with IKM.

Table 5.6: Results from Experiment 3

Index	Algorithm				
	GAIKM	IKM	SKM	GCUK	NSGA-II
misclassif.	0.06	0.23	0.17	0.21	0.06
	(0.01)	(0.12)	(0.08)	(0.05)	(0.01)
Rand	0.90	0.66	0.77	0.75	0.90
	(0.01)	(0.16)	(0.09)	(0.05)	(0.01)
Jaccard	0.85	0.63	0.72	0.69	0.86
	(0.01)	(0.14)	(0.09)	(0.06)	(0.02)
efficiency	0.89	0.92	0.89	0.88	0.89
	(0.01)	(0.02)	(0.02)	(0.02)	(0.01)
purity	0.96	0.67	0.79	0.76	0.96
	(0.01)	(0.15)	(0.10)	(0.07)	(0.01)
silhouette	0.95	0.72	0.81	0.65	0.95
	(0.00)	(0.14)	(0.10)	(0.13)	(0.00)
within-sum	0.28	0.36	0.36	0.41	0.29
	(0.017)	(0.402)	(0.195)	(0.134)	(0.004)

The standard deviations are enclosed in parentheses

5.3 Application to Real World Data Sets

Six binary data sets that had been used in GCUK-binary were tested in the GAIKM algorithm. Then, to compare the performance of the proposed algorithm, the other clustering methods; GCUK-binary, NSGAII, Incremental K-Means (IKM), Scalable K-Means (SKM) and K-Means algorithm were used with the same number of K.

5.3.1 Results of Parameter Settings

Before the GAIKM process started, several tests for choosing the best parameters were carried at for each of the data sets. This process was to make sure only the best parameters were chosen because GAIKM is run with the choices of many generations and each of the generations gives different fitness values. This was to ensure that the fitness value of the GAIKM was good enough to compare with the other clustering algorithms. It should be noted that, the probability of the mutation is $P_m = x/Lind$ where the x value ranged from $0.7 - 0.9$.

The results for all the data sets are in Table 5.7, Table 5.8, Table 5.9, Table 5.10 and Table 5.11.

Table 5.7: Different parameter settings and the qRW value for Nursery data

Parameter settings	qRW
$G = 30, P = 30, p_c = 0.9, p_m = 0.9/Lind$	2183.00
$G = 30, P = 30, p_c = 0.8, p_m = 0.8/Lind$	2171.00
$G = 50, P = 30, p_c = 0.8, p_m = 0.8/Lind$	**2166.76**
$G = 50, P = 50, p_c = 0.8, p_m = 0.8/Lind$	2176.00

Table 5.8: Different parameter settings and the qRW value for Car Evaluation data

Parameter settings	qRW
$G = 30, P = 30, p_c = 0.8, p_m = 0.8/Lind$	**305.9048**
$G = 50, P = 30, p_c = 0.9, p_m = 0.8/Lind$	305.9048
$G = 50, P = 50, p_c = 0.9, p_m = 0.8/Lind$	305.9048
$G = 100, P = 50, p_c = 0.9, p_m = 0.8/Lind$	305.9048

Table 5.9: Different parameter settings and the qRW value for Bank Marketing data

Parameter settings	qRW
$G = 50, P = 50, p_c = 0.9, p_m = 0.8/Lind$	**4576.58**
$G = 50, P = 30, p_c = 0.8, p_m = 0.8/Lind$	4999.52
$G = 100, P = 50, p_c = 0.9, p_m = 0.8/Lind$	4638.09

The lowest index value for Nursery data set was attained when the parameters were $G = 50, P = 30, p_c = 0.9$ and $p_m = 0.8/Lind$ with $qRW = 2166.7$ (Refer Table 5.7). Surprisingly, for Car Evaluation data the parameter settings gave the same value of the fitness function; $qRW = 305.9048$. For this data set, the parameter settings were $G = 30, P = 30, p_c = 0.8, p_m = 0.8/Lind$ with qRW is equal to 305.9048 (Refer Table 5.8). For Bank Marketing data set, the best parameter settings were $G = 50, P = 50, p_c = 0.9$ and $p_m = 0.8/Lind$ with the value of qRW is equal to 4576.58.

Table 5.10: Different parameter settings and the qRW value for CRTA

Parameter settings	qRW
$G = 50, P = 50, p_c = 0.9, p_m = 0.8/Lind$	613.9947
$G = 100, P = 100, p_c = 0.9, p_m = 0.8/Lind$	579.8974
$G = 200, P = 100, p_c = 0.9, p_m = 0.8/Lind$	566.9113
$G = 300, P = 100, p_c = 0.9, p_m = 0.8/Lind$	**562.8963**
$G = 50, P = 50, p_c = 0.8, p_m = 0.8/Lind$	566.6752

Table 5.11: Different parameter settings and the qRW value for CRTA2

Parameter settings	qRW
$G = 50, P = 50, p_c = 0.9, p_m = 0.8/Lind$	38.4439
$G = 100, P = 100, p_c = 0.9, p_m = 0.8/Lind$	38.7209
$G = 200, P = 100, p_c = 0.9, p_m = 0.8/Lind$	39.0227
$G = 200, P = 200, p_c = 0.9, p_m = 0.8/Lind$	36.7214
$G = 300, P = 100, p_c = 0.9, p_m = 0.8/Lind$	37.2580
$G = 50, P = 50, p_c = 0.8, p_m = 0.8/Lind$	**36.7139**

Results showed that the best parameter settings for CRTA with the lowest value of qRW is equal to 562.8963 was given from the parameters $G = 300, P = 100, p_c = 0.9, p_m = 0.8/Lind$ (Table 5.10). However, because of the limitations of speed on the computer and the time taken for the algorithm to converge, the process of iteration process stopped with this setting. For the CRTA2 data set, the best setting were when $G = 50, P = 50, p_c = 0.8$ with the qRW is equal to 36.7139 (Refer Table 5.11).

Table 5.12: Different parameter settings and the qRW value URTA

Parameter settings	qRW
$G = 50, P = 30, p_c = 0.9, p_m = 0.9/Lind$	165.3707
$G = 100, P = 50, p_c = 0.9, p_m = 0.9/Lind$	169.7493
$G = 150, P = 50, p_c = 0.9, p_m = 0.9/Lind$	165.3707
$G = 100, P = 50, p_c = 0.9, p_m = 0.8/Lind$	165.3707
$G = 100, P = 50, p_c = 0.9, p_m = 0.8/Lind$	**161.8224**

The lowest qRW value for URTA data set was when the parameter settings were $G = 100, P = 50, p_c = 0.9, p_m = 0.8/Lind$ with a value of $qRW = 161.8224$. By assuming that the choices of the parameter settings were quite enough to compete with the other clustering algorithm, the searching process was stopped.

5.3.2 Results of GAIKM

After finalizing the parameter settings for each of the data sets, the process of GAIKM proceeded. Table 5.13 below contains the results of GAIKM for all of the data sets.

Table 5.13: Results for GAIKM

Data set	GAIKM	IKM	SKM	K-means	GCUK-binary	NSGAII-binary
Nursery	**2166.76**	2256.1	2258.6	2209.4	2328.4	2153.9
Car	**305.95**	312.49	320.91	322.89	311.64	310.86
Bank	**4638.09**	5475.9	4885.8	5130.7	5118.7	5219.1
CRTA	**562.89**	1040.1	981.42	719.2	630.09	951.19
CRTA2	**38.58**	57.12	60.78	56.47	49.58	64.09
URTA	**161.82**	188.73	290.09	259.71	237.61	457.35

From the results in Table 5.13, all the lowest values of the objective function were attained by the proposed method, GAIKM. Even for the imbalanced data CRTA and URTA, the GAIKM gave the lowest objective value compared to the other clustering methods. These values are far lower than the IKM and SKM, showing that the GAIKM is an efficient clustering algorithm especially to cluster binary data and imbalanced class data sets.

5.4 Summary

In this chapter, the new clustering algorithm that specifically made for binary data set is proposed. The IKM that was specially designed for binary data which use useful features in order to speed the calculations and accelerate convergence to the solution was used.

In this study, GA was chosen to combine with IKM to enhance the performance of IKM. The chromosome representation used in this proposed algorithm similar as in GCUK-binary.

This implementation was proven to be efficient in the previous chapter. The GA was acted as a tool to provide a group of the best populations (solutions) in the entire population. On the other hand, IKM acted as tools for the clustering process.

However, unlike the IKM, in the GAIKM, each case has a chance to be assign in a different cluster system. This process is different than IKM which only gave a single one solution. Elitism used in GA also enhance the performance of GAIKM as it ensures that the best solutions are not eliminated during the process of iteration.

This proposed method, GAIKM, was applied to six binary data sets; Nursery, Car Evaluation, Bank Marketing, CRTA, CRTA2 and URTA data sets. For CRTA and URTA, both data sets have an imbalanced class issue. To support the results, the majority of instances (minor and non-injury) from the CRTA data were excluded to balance the instances in the data set. This data set was then referred to as the CRTA2. Thus, the applications of GAIKM and the findings provided new contributions to the field.

As for the algorithmic performance, the results showed that GAIKM algorithm greatly improved the IKM algorithm itself. The comparison of GAIKM with SKM, IKM, GCUK, NSGAII and K-means showed that GAIKM attained the lowest of the fitness value. The findings from the imbalanced class issue also gave good performance and proved that this proposed algorithm is competent to be an efficient and effective new clustering algorithm.

Chapter 6

Conclusion

In this chapter, the findings and results from the GCUK-binary and GAIKM are discussed thoroughly. The recommendations for future research can be found at the end of this chapter.

6.1 GCUK-binary

A widespread genetic algorithm for clustering, namely the *Genetic Clustering for Unknown K* (GCUK) [Bandyopadhyay and Maulik, 2002b, Maulik and Bandyopadhyay, 2000] for single objective optimization was considered in this study.

Four aspects were considered in this study. First, the implementation of single objective optimization GA, on real data sets, where some new improvements were made to deal efficiently with binary data. GCUK has never been tested with binary data sets before except in some extended research done by Lin et al. [2005]. Thus, some improvements were needed to cope with the binary environment, including the distance measurement used. All the chromosomes in the GCUK-binary were filled with the strings and no #s or empty clusters were allowed.

The second contribution was the implementation of the cluster centre as a chromosome representation used in this study. GCUK, uses a real encoding as a chromosome representation which leads to a longer convergence time when the data set size increased. As mentioned previously, each process iteration in GA utilizes every point / string in each chromosome for each generation. Thus, the larger the data point, the more time is needed to reach the global optima. In this study, the cluster centre replaced the previous encoding representation which is much shorter than the length of the data points themselves, and it is believed that it can save more time in the process of converging. However, the comparison time was not taken into consideration in this study, but put aside for future research that focuses on computational complexity.

Third, even though GA is capable to give automatic number of clusters, to save time, the Silhouette index was used to produce coherent results and to be effective for choosing an optimal K. The first implementation of the original GCUK with the imbalanced class data showed that, GCUK gave the maximum K with one empty cluster. Thus, in this study, the number of clusters, K was held to $k_{min} = k_{max}$, to avoid the algorithm returning empty cluster(s).

The last aspect was the adaption of this algorithm to cluster imbalanced class data sets. This algorithm had never been tested with this problem before and results showed that it outperformed the K-means algorithm.

All the findings suggest that GA is an efficient, effective and competitive clustering algorithm to cluster various types of data set. The advantages and the findings should become a starting point for building new clustering algorithms .

6.2 GAIKM

The proposed clustering method, Genetic Algorithms-Incremental K-means (GAIKM) is a combination of the Incremental K-means and Genetic Algorithms. This is the first time that a thorough study which is based on extensive simulation experiments and several real world data sets have been performed on binary data and imbalanced class data sets.

The reason for choosing the IKM was that this clustering algorithm has proven to be a successful method for clustering binary data. It also has a good computational time to converge compared to the other methods such as On-line K-means, Scalable K-means and K-means algorithm. Thus, this combination was a perfect hybridization as both had many advantages to cluster binary data and imbalanced class data sets.

The process of GAIKM is different with IKM where there is a chance for all the data points to be assigned in s different cluster systems. For every new case that is input into an algorithm, the process of GA is developed and produces a new population, and replaces the old one. Finally, at the end, s different partitions of all n data points are available. This process makes GAIKM better than IKM in providing the best solution.

According to the results from the numerical applications, it appears to be very promising method. In all experiments the proposed hybrid that combines genetic algorithms with IKM outperformed all other binary clustering algorithms used for comparison. Guidelines for choosing the algorithm parameters were given and were found to be effective in enhancing the performance of the binary clustering algorithms.

6.3 Recommended Future Studies

This research found that GA has a good potential to cluster binary data and successfully handle imbalanced class data sets. However, the present study did not take into account any missing data. The observations that had the missing data were eliminated from the population. A new GA clustering method that is less influence by the missing data could be developed as real data sets commonly face this issue. As for the proposed method GAIKM, it could be interesting and beneficial if the future research could cope with the missing data as well.

Another potential avenue for future research is to take into account the time taken for the proposed method, GAIKM, to converge. This study did not take this into account, as the main objective was to examine the performance of the proposed method when compared with the other clustering methods. This future plan could yield significant results as the IKM had proved as the fastest clustering algorithm compared to Scalable K-means, Online K-means and the standard K-means.

Bibliography

A. Casillas, M. T. González de Lena, and R. Martínez (2003). Document Clustering into an Unknown Number of Clusters Using a Genetic Algorithm. *Springer-Verlag Berlin Heidelberg*, 2807(1):43–49.

Arunprasath, S., Chandrasekar, S., Venkatalakshmi, K., Shalinie, S. M., and Arunprasath, S. and Chandrasekar, S. and Venkatalakshmi, K. and Shalinie, S. (2010). Classification of remote sensed image using Rapid Genetic k-Means algorithm. In *Communication Control and Computing Technologies (ICCCCT), 2010 IEEE International Conference on*, pages 677–682.

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In *Proc of the Second International Conference of Genetic Algorithms and their application*, pages 14–21.

Bandyopadhyay, S. and Maulik, U. (2002a). An Evolutionary technique based on K-means algorithm for optimal clustering in RN. *Information Sciences*, 146:221–237.

Bandyopadhyay, S. and Maulik, U. (2002b). Genetic Clustering For Automatic Evolution Of Clusters And Application To Image Classification. *Pattern Recognition*, 35(6):1197–1208.

Bandyopadhyay, S. and Pal, S. K. (2007). Genetic Algorithms in Clustering. In *Classifica-*

tion and Learning Using Genetic Algorithms, Natural Computing Series, pages 181–212. Springer Berlin Heidelberg.

Baragona, R., Bocci, L., and Medaglia, C. M. (2006). Genetic Clustering Algorithms: a Comparison Simulation Study. *International Journal of Modelling & Simulation*, 26(3):190–200.

Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms.* Springer US, Boston, MA.

Boriah, S., Chandola, V., and Kumar, V. (2008). Similarity Measures for Categorical Data: A Comparative Evaluation. *Proceedings of 2008 {SIAM} Data Mining Conference*, pages 243–254.

Bradley, P. S., Fayyad, U., Reina, C., and Paul S. Bradley, Usama M. Fayyad, C. A. R. (1998). Scaling Clustering Algorithms to Large Databases. *American Association for Artificial Intelligence*, pages 9–15.

Chakraborty, S. and Nagwani, N. (2011). Performance Evaluation of Incremental K-means Clustering Algorithm. *IFRSA International Journal of Data Warehousing & Mining*, 1:54–59.

Cole, R. M. (1998). *Clustering With Genetic Algorithms.* PhD thesis, Dept. of Computer Science,University of Western Australia.

Corne, D. W., Jerram, N. R., Knowles, J. D., Oates, M. J., and J, M. (2001). PESA-II: Region-based selection in evolutionary multiobjective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 283–290. Morgan Kaufman Publisher.

Davies, D. L. and Bouldin, D. W. (1979). A Cluster Separation Measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-1(2):224–227.

Deb, K. and Kalyanmoy, D. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., New York, NY, USA.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

Deepa, T. and Punithavalli, M. (2010). An Analysis for Mining Imbalanced Datasets. *International Journal of Computer Science and Information Security*, 8(1):132–137.

Duda, R. O. and Hart, P. (1974). Pattern Classification and Scene Analysis. *IEEE Transaction on Automatic Control*, 19(4):462–463.

Fatima Zohra Bellala Belahbib, F. S. (2011). Genetic algorithm clustering for color image quantization. *Visual Information Processing (EUVIP), 2011 3rd European Workshop on.*

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition.

Guha, S., Rastogi, R., and Shim, K. (1998). CURE: an efficient clustering algorithm for large databases. *ACM SIGMOD Record*, 27(2):73–84.

Guo, H.-x., Zhu, K.-j., Gao, S.-w., and Liu, T. (2006). An Improved Genetic k-means Algorithm for Optimal Clustering. In *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*, pages 793–797. Ieee.

Handl, J. and Knowles, J. (2004). Evolutionary Multiobjective Clustering. *Lecture Notes in Computer Science*, 3242:1081–1091.

Handl, J. and Knowles, J. (2007). An Evolutionary Approach to Multiobjective Clustering. *IEEE Transactions on Evolutionary Computation*, 11.

Hruschka, E. R., Campello, R., Freitas, A. A., and de Carvalho, A. (2009). A Survey of Evolutionary Algorithms for Clustering. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 39(2):133–155.

Hubert, L. (1985). Classification ˜1985. *Journal of Classification*, 218:193–218.

Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8):651–666.

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323.

Jie, L. I., Xinbo, G. A. O., and Li-cheng, J. (2003). A GA-Based Clustering Algorithm for Large Data Sets With Mixed Numeric and Categorical Values. In *Computational Intelligence and Multimedia Applications, 2003. ICCIMA 2003. Proceedings. Fifth International Conference on*, pages 102–107.

Kaukoranta, T. and Nevalainen, O. (1997). Genetic algorithms for large scale clustering problems. *The Computer Journal*, 40(9):547–554.

Krishna, K., Narasimha Murty, M., and Murty, M. N. (1999). Genetic K-means algorithm. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 29(3):433–439.

Krovi, R. (1992). Genetic algorithms for clustering: a preliminary investigation. In *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, pages 540–544.

Kwedlo, W. and Iwanowicz, P. (2010). Using genetic algorithm for selection of initial cluster centers for the K-means method. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6114 LNAI(PART 2):165–172.

Leisch, F., Weingessel, A., and Dimitriadou, E. (1998). Competitive Learning for Binary Valued Data. In *Proceedings of the 8th International Conference on Arti Neural Networks (ICANN 98*, pages 779–784. Springer.

Leon, E., Nasraoui, O., and Gomez, J. (2006). ECSAGO: Evolutionary Clustering with Self Adaptive Genetic Operators. *2006 IEEE International Conference on Evolutionary Computation*, pages 1768–1775.

Li, M., Yang, S., Liu, X., and Wang, K. (2013). IPESA-II : Improved Pareto Envelope-Based Selection Algorithm II Miqing IPESA-II Problems of PESA-II. In Purshouse, R., Fleming, P., Fonseca, C., Greco, S., and Shaw, J., editors, *Evolutionary Multi-Criterion Optimization*, pages 143–155. Springer Berlin Heidelberg.

Li, X., Zhang, L., Li, Y., and Wang, Z. (2010). A improved k-means clustering algorithm combined with the genetic algorithm. In *Digital Content, Multimedia Technology and its Applications (IDC), 2010 6th International Conference on*, pages 121–124.

Lichman, M. (2013). UCI Machine Learning Repository.

Lin, H.-J., Yang, F.-W., and Kao, Y.-T. (2005). An Efficient GA-based Clustering Technique. *Tamkang Journal of Science and Engineering*, 8(2):113–122.

Lu, Y., Lu, S., Fotouhi, F., Deng, Y., and Brown, S. J. (2004a). FGKA: A Fast Genetic

K-means Clustering Algorithm. In *Proceedings of the 2004 ACM symposium on Applied computing*, SAC '04, pages 622–623, New York, NY, USA. ACM.

Lu, Y., Lu, S. Y., Fotouhi, F., Deng, Y. P., and Brown, S. J. (2004b). Incremental genetic K-means algorithm and its application in gene expression data analysis. In *BMC BIOINFORMATICS*, volume 5 of *SAC '04*, pages 622–623, New York, NY, USA. ACM.

Macqueen, J. B. (1967). Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.

Maulik, U. and Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. *Pattern Recognition*, 33(9):1455–1465.

Maulik, U., Bandyopadhyay, S., and Mukhopadhyay, A. (2011). *Multiobjective Genetic Algorithms for Clustering: Applications in Data Mining and Bioinformatics*. Springer.

Mitchell, M. (1995). *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA.

Mukhopadhyay, A., Maulik, U., and Bandyopadhyay, S. (2009). Multiobjective Genetic Algorithm-Based Fuzzy Clustering of Categorical Attributes. *Evolutionary Computation, IEEE Transactions on*, 13(5):991–1005.

Mukhopadhyay, A., Maulik, U., and Bandyopadhyay, S. (2015). A Survey of Multiobjective Evolutionary Clustering. *ACM Computing Surveys*, 47(4):1–47.

Mukhopadhyay, A., Maulik, U., Bandyopadhyay, S., and Coello, C. A. C. (2014a). A Survey of Multiobjective Evolutionary Algorithms for Data Mining: Part II. *IEEE Transactions on Evolutionary Computation*, 18(1):20–35.

Mukhopadhyay, A., Member, S., Maulik, U., and Member, S. (2014b). Survey of Multi-Objective Evolutionary Algorithms for Data Mining : Part-II. *IEEE Transactions on Evolutionary Computation*, 18(1):20–35.

Murthy, C. and Chowdhury, N. (1996). In search of optimal clusters using genetic algorithms. *Pattern Recognition Letters*, 17(8):825–832.

Murty, M., Babaria, R., and Chiranjib, B. (2008). Clustering Based on Genetic Algorithms. In *Multi-Objective Evolutionary Algorithms for Knowledge Discovery from Databases*, volume 98 of *Studies in Computational Intelligence*, pages 137–159. Springer Berlin / Heidelberg.

Nasraoui, O. and Krishnapuram, R. (2000). A novel approach to unsupervised robust clustering using genetic niching. *Ninth IEEE International Conference on Fuzzy Systems. FUZZ-IEEE 2000 (Cat. No.00CH37063)*, 1:170–175.

Ordonez, C. (2003). Clustering binary data streams with K-means. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, DMKD '03, pages 12–19, New York, NY, USA. ACM.

Ordonez, C. and Omiecinski, E. (2002). FREM: fast and robust EM clustering for large data sets. In *Proceedings of the eleventh international conference on Information and knowledge management*, CIKM '02, pages 590–599, New York, NY, USA. ACM.

Pan, S.-M. P. S.-M. and Cheng, K.-S. C. K.-S. (2007). Evolution-Based Tabu Search Approach to Automatic Clustering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(5):827–838.

Paterlini, S. and Minerva, T. (2010). Genetic algorithms in partitional clustering: a comparison. In *Proceedings of the 11th WSEAS international conference on neural networks and 11th WSEAS international conference on evolutionary computing and 11th WSEAS international conference on Fuzzy systems*, NN'10/EC'10/FS'10, pages 28–36, Stevens Point, Wisconsin, USA. World Scientific and Engineering Academy and Society (WSEAS).

Pham, D. T., Dimov, S. S., and Nguyen, C. D. (2004). An incremental K-means algorithm.

Reeves, C. R. and Rowe, J. E. (2002). Basic Principles. In Sharda, R. and Vo, S., editors, *Genetic AlgorithmsPrinciples and Perspectives*, volume 20 of *Operations Research/Computer Science Interfaces Series*, pages 19–63. Springer US.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(null):53–65.

Rudolph, G. (1994). Convergence analysis of canonical genetic algorithms. *Neural Networks, IEEE Transactions on*, 5(Jan 1994):96–101.

Safe, M., Carballido, J., Ponzoni, I., and Brignole, N. (2004). On Stopping Criteria for Genetic Algorithms.

Saha, I., Sarkar, J. P., and Maulik, U. (2015). Ensemble based rough fuzzy clustering for categorical data.

Saharan, S. and Baragona, R. (2013). A new Genetic Algorithm for clustering binary data with application to traffic road accidents in Christchurch. *Far East Journal of Theoritical Statistics*, 45(1):67–89.

Sanjay, C., N.K., N., and Lopamudra, D. (2011). Performance Comparison of Incremental K-means and Incremental DBSCAN Algorithms. *International Journal of Computer Applications*, 27(11):14–18.

Sanjay Chakraborty, N. N. (2011). Analysis and Study of Incremental K-Means Clustering Algorithm. In *High Performance Architecture and Grid Computing Communications in Computer and Information Science*, pages 338–341. Springer Berlin Heidelberg.

Sérgio Moro and Paulo Cortez and Paulo Rita (2014). A Data-Driven Approach to Predict the Success of Bank Telemarketing. *Decision Support Systems*, 62:22–31.

S.N. Sivanandam and S.N. Deepa (2008). *Introduction to Genetic Algorithms*. Springer Berlin Heidelberg, Berlin, Heidelberg.

Soler, Vicen, Prim, and Marta (2009). Extracting a Fuzzy System by Using Genetic Algorithms for Imbalanced Datasets Classification: Application on Downs Syndrome Detection. In Zighed, D., Tsumoto, S., Ras, Z., and Hacid, H., editors, *Mining Complex Data*, volume 165 of *Studies in Computational Intelligence*, pages 23–39. Springer Berlin / Heidelberg.

Srinivas, N. and Deb, K. (1995). Muiltiobj ective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221—-248.

Tan, Z. and Lu, R. (2009). Application of Improved Genetic K-Means Clustering Algorithm in Image Segmentation. In *Education Technology and Computer Science, 2009. ETCS '09. First International Workshop on*, volume 2, pages 625–628.

Tseng, L. Y. and Yang, S. B. (1997). Genetic algorithms for clustering, feature selection and classification. In *Neural Networks,1997., International Conference on*, volume 3, pages 1612–1616.

Villar, P., Fernández, A., Carrasco, R. a., and Herrera, F. (2012). Feature Selection and Granularity Learning in Genetic Fuzzy Rule-Based Classification Systems for Highly Imbalanced Data-Sets. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 20(03):369–397.

Wang, Y. and Zhang, W. (2009). Clustering analysis based on genetic algorithms. *2009 2nd IEEE International Conference on Computer Science and Information Technology*, pages 325–328.

William M . Rand (1971). Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850.

Wu, S., Yin, S., and Li, M. (2012). A new approach for clustering problem based on binary small world optimization algorithms. *Computer Science and Automation Engineering (CSAE), 2012 IEEE International Conference on*, 3:412 – 416.

Xie, X. and Beni, G. (1991). A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):841–847.

.1 Appendix A

The pseudo-code for GAIKM.

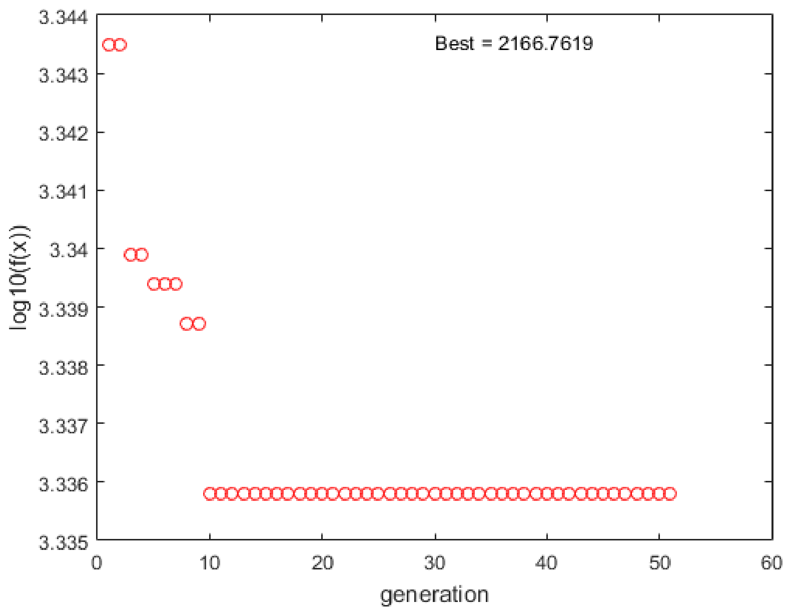
Algorithm 1 The Genetic Algorithm with Incremental K-means

```
Input: $\mathcal{D}$ $k$
Output: $C, R, W$ and $q(R, W)$
  $ga - parameters \leftarrow s\ p_c\ p_m\ g$
  for $\ell = 1$ TO $s$ do
    $C^{(\ell)} \leftarrow binornd(1, \rho, s, dk)$
    for $j = 1$ TO $k$ do
      $M_j^{(\ell)} \leftarrow \bar{0}$
      $N_j^{(\ell)} \leftarrow \bar{0}$
      $W_j^{(\ell)} \leftarrow 1/k$
    end for
  end for
  for $v = 1$ TO $g$ do
    if $hg < n$ then
      $\mathcal{D} \leftarrow \mathcal{D}, \oplus\{t_{(i-1)h+1}, \ldots, t_{ih}\}$,
      $m = ih$
    else
      $m = n$
    end if
    DO selection $\mathcal{P}' \leftarrow \mathcal{P}$
    DO crossover $\mathcal{P}'' \leftarrow \mathcal{P}'$
    DO mutation $\mathcal{P}''' \leftarrow \mathcal{P}''$
    $\mathcal{P} \leftarrow \mathcal{P}'''$
    for $\ell = 1$ TO $s$ do
      for $i = 1$ TO $m$ do
        $j =$ argmin Hamming distance $(t_i, C_j^{(\ell)})$
        $M_j^{(\ell)} \leftarrow M_j^{(\ell)} + t_i$
        $N_j^{(\ell)} \leftarrow N_j^{(\ell)} + 1$
      end for
      $W_j^{(\ell)} \leftarrow N_j^{(\ell)}/m$
      $R_j^{(\ell)} \leftarrow C_j^{(\ell)}(1 - C_j^{(\ell)})$
      $q(R^{(\ell)}, W^{(\ell)}) \leftarrow$ Eqn. (5.1)
    end for
    if $(v \bmod h) == 0$ then
      for $j = 1$ TO $k$ do
        $C_j \leftarrow M_j/N_j$
      end for
    end if
  end for
  $f_{\text{optimal}} = \overset{\min}{\ell}\{f_\ell, \ell = 1, \ldots, s\}$
```

The number of generations of the best individual for GAIKM.

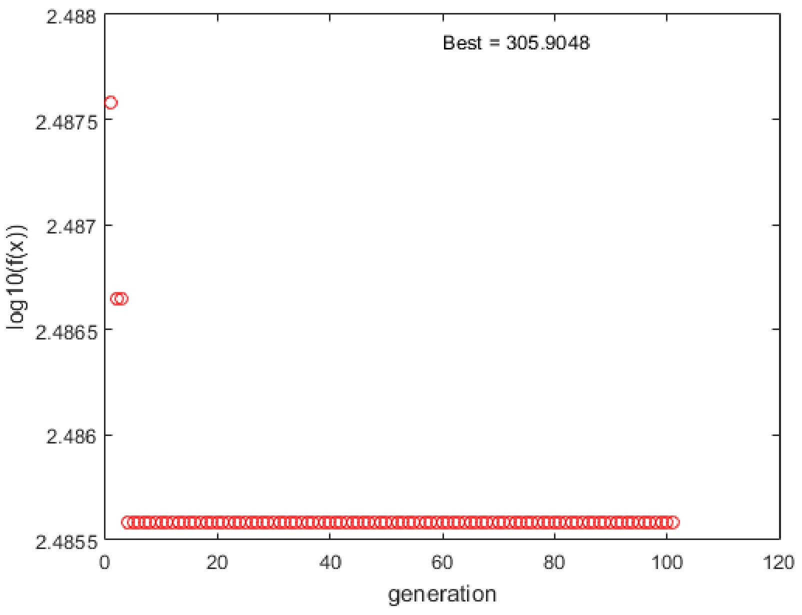
Nursery data set

Figure 1: Number of generations of the best individual for Nursery data set



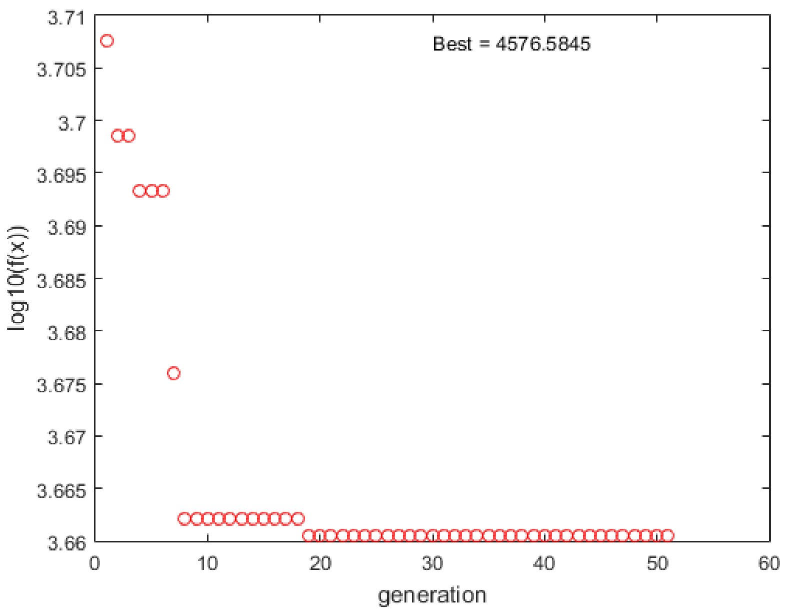
Car Evaluation data set

Figure 2: Number of generations of the best individual for Car Evaluation data set



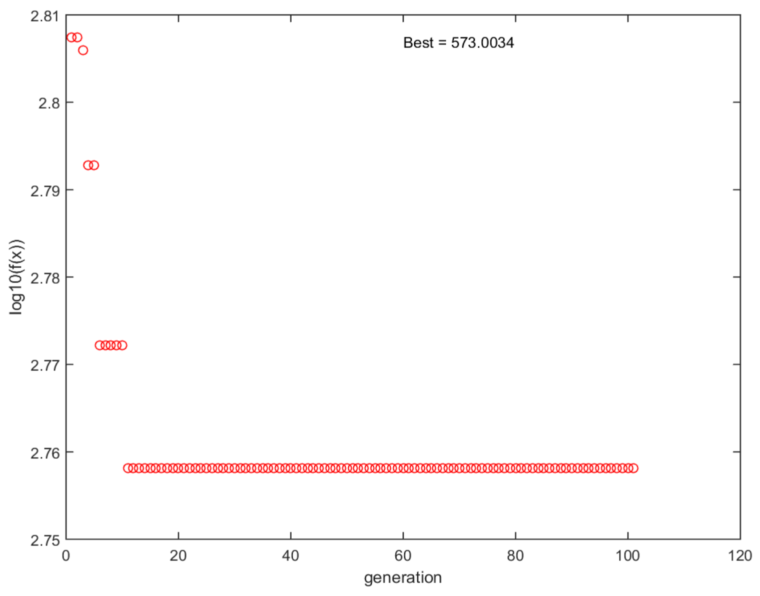
Bank Marketing data set

Figure 3: Number of generations of the best individual for Bank Marketing data set



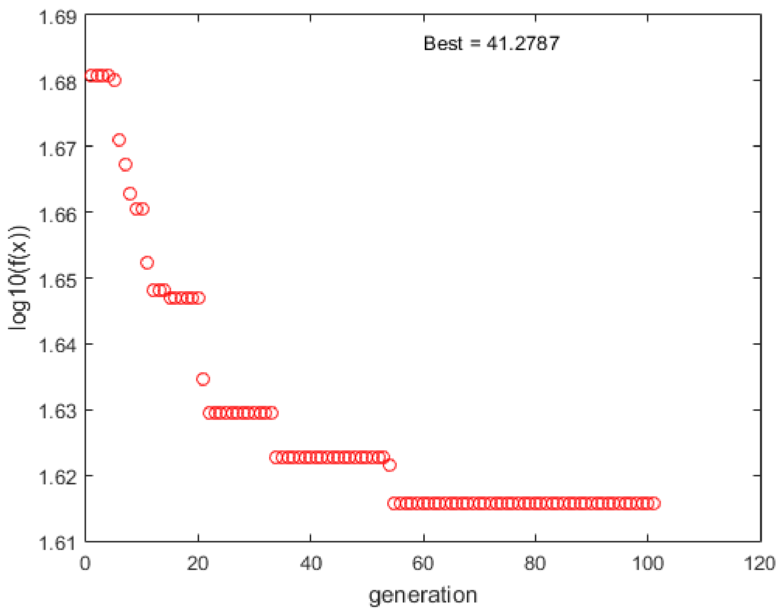
CRTA data set

Figure 4: Number of generations of the best individual for CRTA data set



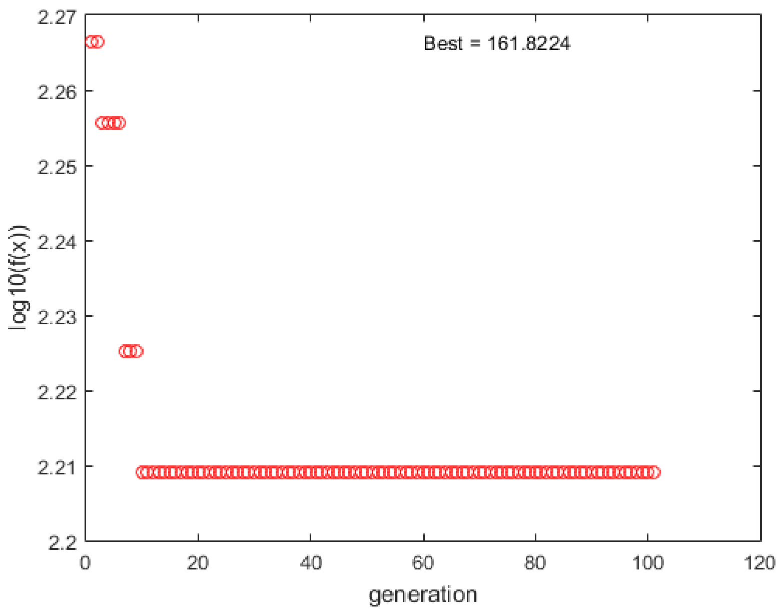
CRTA2 data set

Figure 5: Number of generations of the best individual for CRTA2 data set



URTA data set

Figure 6: Number of generations of the best individual for URTA data set


Best = 161.8224
log10(f(x))
generation