Clustering by Multi Objective Genetic Algorithm

Dipankar Dutta
Dept. of CSE and IT,
U.I.T., The University of Burdwan,
Burdwan, W.B., India.
e-mail: dipankar_dutta@rediffmail.com

Paramartha Dutta
Dept. of Computer and System Sciences,
Visva Bharati University,
Santiniketan, W.B., India.
e-mail: paramartha.dutta@gmail.com

Jaya Sil
Dept. of Computer Science and Technology,
Bengal Engineering and Science University,
Shibpur, W.B., India.
e-mail: js@cs.becs.ac.in

Abstract— **The aim of the paper is to study a real coded multi objective genetic algorithm based *K*-clustering, where *K* represents the number of clusters, may be known or unknown. If the value of *K* is known, it is called *K*-clustering algorithm. The searching power of Genetic Algorithm (*GA*) is exploited to get for proper clusters and centers of clusters in the feature space to optimize simultaneously intra-cluster distance (Homogeneity) (*H*) and inter-cluster distances (Separation) (*S*). Maximization of 1/H and *S* are the twin objectives of Multi Objective Genetic Algorithm (*MOGA*) achieved by measuring *H* and *S* using Euclidean distance metric, suitable for continuous features (attributes). We have selected 10 data sets from the UCI machine learning repository containing continuous features only to validate the proposed algorithms. All-important steps of algorithms are shown here. At the end, classification accuracies obtained by best chromosomes are shown.**

Keywords- Clustering; homogeneity and separation; real coded multi objective genetic algorithm; Pareto optimal front

I. INTRODUCTION

Data mining is the process of extracting previously unknown, comprehensible, and actionable information from large databases and using it to make crucial business decisions – Zekulin [10]. Major data mining tasks are classification, clustering, association rule discovery, sequential pattern discovery, regression, deviation detection, prediction, data visualization and many more. In course of machine learning, machine learns from training data using learning mechanism that can be supervised, unsupervised or reinforced. Clustering is an example of unsupervised learning where class labels are not available at the training phase. The term clustering refers to the identification of natural groups within data sets such that instances in the same groups are more similar than instances in different groups [1], [9], [14], [18], [20]. Objects within the same clusters are similar therefore intra-cluster distances (Homogeneity) (*H*) are low. Objects within the different clusters are different therefore inter-cluster distances (Separation) (*S*) are high.

GAs [4] are "search algorithms based on the dynamics of natural selection and natural genetics" [11]. Categories of *GAs* are - simple *GA* and multi objective *GA* (*MOGA*). When an optimization problem involves only one objective, the task of finding the best solution is a single objective optimization problem. However, in real world, life appears to be quite complex. Most of the problems are consisting of more than one interdependent objective, which are to be minimized or maximized simultaneously. These types of problems are multi objective optimization problem [6]. Like many other machine learning problems, clustering can therefore, be considered as an intrinsically multi objective optimization problem [14]. The use of *GAs* in clustering is an attempt to exploit effectively the large search space usually associated with cluster analysis and better interactions among the features for forming chromosomes. Almost all-conventional methods start searching from a single point and through subsequent iterations converge to the best solution. However, they are sensitive to the choice of initial solution. *GAs* always works on a whole population of points (strings) and perform a nearly global search rather than performing a local, hill-climbing search. This phenomenon improves the chance of reaching to a global optima, vis-à-vis, reduces the risk of being trapped at a local optima, thereby offering robustness.

II. PREVIOUS WORK

Initially *GAs* [17], [26] used to deal with single objective optimization problems such as minimizing or maximizing a variable of a function. Such *GAs* is simple *GAs*. David Schaffer proposed vector evaluated genetic algorithm [31], [32], which was the first inspiration towards *MOGA*. Several improvements on vector evaluated genetic algorithm are proposed in [5], [11]. Vector evaluated genetic algorithm dealt with multivariate single objective optimization. V. Pareto suggested Pareto approach [28] to cope with multi objective optimization problems. In this work chromosomes lying on the Pareto optimal front (dominant chromosomes) are selected by *MOGA*, which are giving near optimal solutions for clustering. According to the dominance relation between two candidate chromosomes, a candidate clustering chromosome Ch_1 dominates another candidate clustering chromosome Ch_2 if and only if: 1) Ch_1 is strictly better than Ch_2 in at least one of all the measures considered in the fitness function and 2) Ch_1 is not worse than Ch_2 in any of the measures considered in the fitness

function. Researchers developed different implementation of *MOGA* e.g. *SPGA* [33], *PAES* [21] and *NSGA-II* [7]. In this paper, we have proposed a novel *MOGA* to find out near optimal clusters by optimizing *H* and *S* values.

From the viewpoint of the chromosome encoding scheme *MOGA* can be categorized in binary coded *MOGA* and real coded *MOGA*. Michalewicz [26] indicated that for real valued numerical optimization problems, floating-point representations outperforms binary representation because they are more consistent, precise and lead to faster execution. For this and to keep the mapping between the actual cluster centers and the encoded centers effective we have carried out real coded *MOGA* in the work.

Traditional clustering algorithms often fail to detect meaningful clusters because most real-world data sets are characterized by a high-dimensional, inherently sparse data space [13]. Most of the clustering approaches, single objective or multi-objective, are converted to a single objective by weighed sum method [19] or by any other means. However only a few *MOGA* clustering approaches have been proposed so far and their experimental results have proven that *MOGA* clustering approaches significantly outperform existing simple *GA* clustering approaches [13]. Earlier work on clustering by simple *GAs* are [24], [25], [27] and by *MOGAs* are [2], [15], [22], [23], [30].

III. DEFINITIONS

A relation R and a list of features $A_1,A_2,...A_n$ defines a relational schema $R(A_1,A_2,...A_n)$, where n = total number of features.

The domain of A_i is $dom(A_i)$, where $1 \leq i \leq n$.

X represents a data set, which is a set of tuples that is $X = \{t_1,t_2,...t_m\}$, where m = total number of tuples or records. X can be complete set, training set or test set of any data set.

Each tuple t is a n-dimensional feature vector, which is an ordered list of n values that is $t = [v_1^t, v_2^t,...v_n^t]$, where $v_i^t \in dom(A_i)$, with $1 \leq i \leq n$. v_i^t is i^{th} value in tuple t, which corresponds to the feature A_i. Each tuple, t belongs to a predefined class represented by v_n^t where $v_n^t \in dom(A_n)$. v_n^t or class labels are unknown in clustering, so $t = [v_1^t, v_2^t,...v_{(n-1)}^t]$. After normalization (explained in section A.2) $t = [nv_1^t, nv_2^t,...nv_{(n-1)}^t]$, where $0 \leq nv_i^t \leq 1$.

The problem is to cluster every t_i of X ($1 \leq i \leq m$) in K numbers of non-overlapping groups $\{C_1,C_2,...C_K\}$, where $C_1 \cup C_2 \cup ...C_K = X$, $C_i \neq \varnothing$ and $C_i \cap C_j = \varnothing$ for $i \neq j$ and $j \leq K$.

A solution is a set of cluster centers (*CCs*) that is $\{CC_1,CC_2,...CC_K\}$. Each CC_i is $(n-1)$-dimensional feature vector, that is $CC_i = [c_1^i, c_2^i,...c_{(n-1)}^i]$.

IV. PROPOSED APPROACH

Figure 1 shows a flowchart of the process.

Before applying *MOGA K*-clustering algorithm data need to be preprocessed. Description of several preprocessing steps [12], [29] are given below.

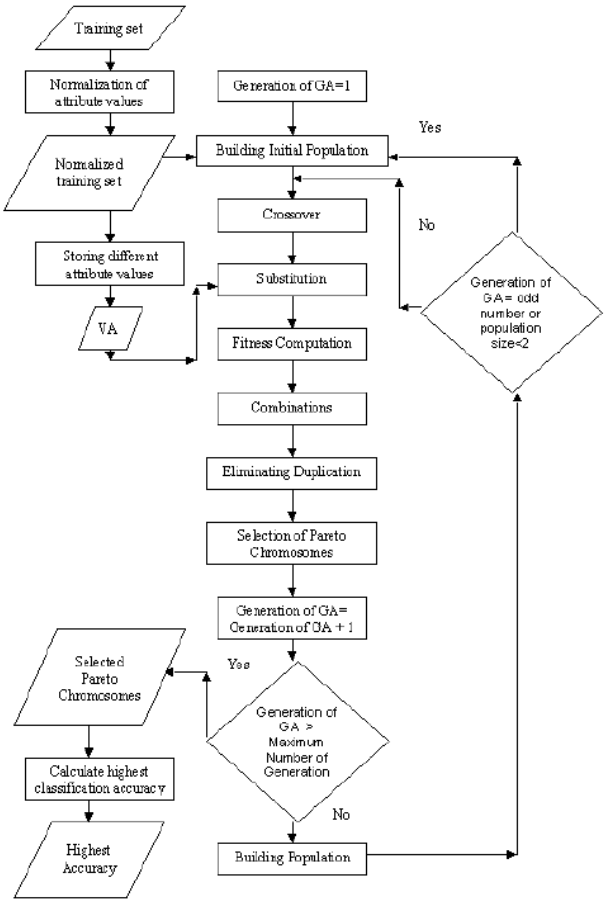

Figure 1. Flowchart of *MOGA*

A. Preprocessing:

1) Making class label as the last attribute:

Original data sets are changed to label the last feature as class label. After this step class label becomes n^{th} feature. In clustering class labels are unknown so we are considering first $(n-1)$ features of data sets.

2) Normalization of data:

A_i^{min} (minimum value of A_i) and A_i^{max} (maximum value of A_i) are calculated, where $1 \leq i \leq (n-1)$. As defined earlier $t = [v_1^t, v_2^t,...v_{(n-1)}^t]$, with $v_i^t \in dom(A_i)$, where $1 \leq i \leq (n-1)$. After normalization $t = [nv_1^t, nv_2^t,...nv_{(n-1)}^t]$ (where $0 \leq nv_i^t \leq 1$) using equation 1. It is a simple linear interpolation based conversion formula.

$$nv_i^t = (v_i^t - A_i^{min})/(A_i^{max} - A_i^{min}) \text{ for } i = 1, 2, ...(n-1). \quad (1)$$

3) Storing distinct values of each attribute:

All distinct normalized values of A_i are stored in VA_i (values of attributes) considering test data set where $1 \leq i \leq (n-1)$.

B. *Building initial population:*

In most of the literature on *GA* [24], [25], [27] fixed number of prospective solutions build initial population (*IP*). Here *IP* size is the nearest integer value of 10% of training set size. Population size guides searching power of *GA* and therefore its size should increase with the size of the data set. Here *IPS* means *IP* size and Ch_j represents the j^{th} chromosome in the population, where $1 \leq j \leq IPS$. Algorithm 1 shows the steps of building initial population.

Algorithm 1 Algorithm for building initial population

$m \leftarrow$ total number of tuples in training set
$K \leftarrow$ number of clusters
$IPS \leftarrow \lfloor (1/10 \times m) \rfloor$
for $j = 1$ to IPS **do**
  for $i = 1$ to K **do**
    $rn_i \leftarrow GenerateRandomNumber(1, m)$/* Generate integer random number between 1 to m */
  end for
  $Ch_j \leftarrow nv_1^{rn_1}, nv_2^{rn_1}, ...nv_{n-1}^{rn_1} | nv_1^{rn_2}, nv_2^{rn_2}, ...nv_{n-1}^{rn_2} | ... | nv_1^{rn_K}, nv_2^{rn_K}, ...nv_{n-1}^{rn_K} |$ /* $nv_l^{rn_i}$ represents normalized value of l^{th} feature of rn_i^{th} tuple. ',' separates dimensions of CC_i and '|' separates CC_i */
end for

As tuples of data set built cluster centers, it induces faster convergence of *MOGA*, compared to building chromosomes by random number generation.

C. *Crossover:*

Single point crossover is applied with crossover probability (P_{co}), chosen as 0.5. As two chromosomes are taking part in one crossover, 100% chromosomes undergo with this crossover.

The numbers of possible crossover points in a chromosome are $K \times (n\text{-}1)$, where K is the number of clusters and n is the number of features including class labels. As class labels are not taking part in chromosome formation so (n-1) number of features are taking part in chromosome building. *PCh* and *CCh* stands for parent and child chromosomes, N_{pch} stands for the number of parent chromosomes, at first $N_{pch} = IPS$. Algorithm 2 explains crossover.

D. *Substitution:*

Substitution probability (P_{sb}) of *MOGA* is 0.01. Produced child chromosomes by crossover are parent chromosomes of this step. Here $dom(A_i)$ is continuous. In conventional mutation, any random number in the range of 0 to 1 can replace nv_i resulting many chromosomes. Number of substitution points $N_{sb} = \lfloor (N_{pch} \times K \times (n\text{-}1) \times P_{sb}) \rfloor$ or N_{pch} whichever is less. Algorithm 3 explains substitution.

E. *Fitness Computation:*

As stated earlier, intra-cluster distance (Homogeneity) (H) and inter-cluster distances (Separation) (S) are two measures for optimization. Maximization of $1/H$ and S are the twin objectives of *MOGA*. It converts the optimization problem befitting into max-max framework. System is computing the values of H and S by using the Euclidean distance per feature measures.

Algorithm 2 Algorithm for crossover

/* All Chs produced by building population methods are $PChs$ of crossover */
$K \leftarrow$ number of clusters
$n \leftarrow$ number of features including class labels
$N_{pch} \leftarrow$ Number of $PChs$ in population
$P_{co} \leftarrow 0.5$
$N_{co} \leftarrow \lfloor N_{pch} \times P_{co} \rfloor$
for $counter = 1$ to N_{co} **do**
  repeat
    $rn1 \leftarrow GenerateRandomNumber(1, N_{pch})$/* Generate integer random number between 1 to N_{pch} */
    $rn2 \leftarrow GenerateRandomNumber(1, N_{pch})$/* Generate integer random number between 1 to N_{pch} */
  until $rn1 \neq rn2$ **AND** $PCh_{rn1} \neq null$ **AND** $PCh_{rn2} \neq null$
  $N_{sp} \leftarrow K \times (n-1)$
  $rn3 \leftarrow GenerateRandomNumber(1, N_{sp})$/* Generate integer random number between 1 to N_{sp}. Position of $rn3^{rd}$ special character ('|' or ',') is CP */
  $CCh_{rn1} \leftarrow TakePartOf(PCh_{rn1}, START, CP) + TakePartOf(PCh_{rn2}, CP + 1, END)$/* Combine START to CP part of PCh_{rn1} with CP+1 to END part of PCh_{rn2} */
  $CCh_{rn2} \leftarrow TakePartOf(PCh_{rn2}, START, CP) + TakePartOf(PCh_{rn1}, CP + 1, END)$/* Combine START to CP part of PCh_{rn2} with CP+1 to END part of PCh_{rn1} */
end for
/* For PChs not taking part in crossover */
for $counter = 1$ to N_{pch} **do**
  if $PCh_{counter} \neq Null$ **then**
    $CCh_{counter} \leftarrow PCh_{counter}$
  end if
end for

Equation 2 is calculating the Euclidean distance per feature between one cluster center and one tuple.

$$d(CC_i, t_j) = [\sum_{l=1}^{l=(n-1)} (c_l^i - nv_l^j)^2]^{1/2} \quad (2)$$

If $d_{lowest}(CC_i, t_j)$ is the lowest distance for any value of i (where $1 \leq i \leq K$), then t_j is assigned to C_i and H is defined by equation 3.

$$H = [\sum_{j=1}^{j=m} d_{lowest}(CC_i, t_j)/(n-1)]/m \quad (3)$$

Equation 4 calculates the Euclidean distance between two tuples of training data.

$$d(t_i, t_j) = [\sum_{l=1}^{l=(n-1)} (nv_l^i - nv_l^j)^2]^{1/2} \quad (4)$$

If t_j and t_p are two distinct tuples from training data, t_j is assigned to C_x and t_p is assigned to C_y based on $d_{lowest}(CC_i,t_j)$ and S is defined by equation 5.

$$S = [\sum_{j,p=1}^{j,p=m} d(t_j, t_p)/(n-1)] \quad (5)$$

where $j \neq p$, $C_x \neq C_y$ and $x \neq y$.

For every chromosome in the population, algorithm 4 calculates *1/H* and *S*.

Algorithm 3 Algorithm for substitution

/* All $CChs$ produced by crossover are $PChs$ of substitution */
$K \leftarrow$ number of clusters
$n \leftarrow$ number of features including class labels
$N_{pch} \leftarrow$ Number of $PChs$ in population
$P_{sb} \leftarrow 0.01$
$N_{sb} \leftarrow \min(\lfloor (N_{pch}) \times K \times (n-1) \times P_{sb} \rfloor$ **OR** $N_{pch})$.
for $counter = 1$ to N_{sb} **do**
  repeat
    $rn1 \leftarrow GenerateRandomNumber(1, (N_{pch} \times K \times (n-1)))$/* Generate integer random number between 1 to $(N_{pch} \times K \times (n-1))$ */
    $rn2 \leftarrow \lfloor (rn1/(K \times (n-1))) \rfloor$
    $rn3 \leftarrow |(rn1/(K \times (n-1)))|$
    $rn4 \leftarrow |(rn3/(n-1))|$
  until $PCh_{rn2} \neq null$
  $rn3^{rd}$ value of PCh_{rn2} is substituted by any value of VA_{rn4} to produce CCh_{rn2}
  $PCh_{rn2} \leftarrow null$
end for
/* For $PChs$ not taking part in substitution */
for $counter = 1$ to N_{pch} **do**
  if $PCh_{counter} \neq Null$ **then**
    $CCh_{counter} \leftarrow PCh_{counter}$
  end if
end for

F. Combination:

At the end of every generation of *MOGA* some chromosomes are lying on the Pareto optimal front. These chromosomes have survived and known as Pareto chromosomes. Chromosomes are obtained from previous method (section E), say NC_{pre} and previous generation Pareto chromosomes (section H) of *MOGA*, say PC_{pre} are considered to perform in the next generation. For the first generation of proposed *MOGA*, PC_{pre} is zero because of the nonexistence of the previous generation.

In general, for i^{th} generation of *MOGA*, if NC_{pre} is *m* and PC_{pre} at the end of $(i\text{-}1)^{th}$ generation of *MOGA* is *n* then after combination number of chromosomes are *n* + *m*.

Algorithm 4 Algorithm for calculating $1/H$ and S

$m \leftarrow$ total number of tuples in the training set
$K \leftarrow$ number of clusters
$n \leftarrow$ number of features
/* **********Building M********** */
for $j = 1$ to m **do**
  for $i = 1$ to K **do**
    $a_{ij} \leftarrow 0$ /* a_{ij} is any element of the matrix M */
  end for
end for
for $j = 1$ to m **do**
  for $i = 1$ to K **do**
    Calculate $d(CC_i, t_j)$ using equation 2
    if $a_{j1} = 0$ **then**
      $a_{j1} \leftarrow d(CC_i, t_j)/(n-1)$
      $a_{j2} \leftarrow i$
    else if $a_{j1} > d(CC_i, t_j)/(n-1)$ **then**
      $a_{j1} \leftarrow d(CC_i, t_j)/(n-1)$
      $a_{j2} \leftarrow i$
    end if
  end for
end for
/* **********Calculating $1/H$********** */
$TotalDistance \leftarrow 0$
for $j = 1$ to m **do**
  $TotalDistance \leftarrow (TotalDistance + a_{j1})$
end for
$H \leftarrow TotalDistance/m$
Calculate $1/H$
/* **********Calculating S********** */
$TotalDistance \leftarrow 0$
for $j = 1$ to m **do**
  for $p = (j+1)$ to m **do**
    $Dist \leftarrow 0$
    if $a_{j2} \neq a_{p2}$ **then**
      $Dist = d(t_j, t_p)/(n-1)$
      $TotalDistance \leftarrow (TotalDistance + Dist)$
    else
      do nothing
    end if
  end for
end for
$S \leftarrow TotalDistance$

G. Eliminating duplication:

Chromosomes lying on the Pareto optimal front of the previous generation may be generated again in the present generation of *MOGA*. As a result of that combined chromosome set will have the two copies of same

chromosome. So in this process only unique chromosomes are selected from a combined chromosome set.

H. Selection of Pareto chromosomes:

All chromosomes lying on the front of *max(1/H, S)* are selected as Pareto chromosomes. In other words maximizing *1/H* and *S* are two objectives of *MOGA*. As this process is applied after combination, elitism [6], [7] is adhered to.

I. Building intermediate population:

From 2nd generation onwards for every even generation, selected Pareto chromosomes of previous generation build population. This helps to build a population of chromosomes close to the Pareto optimal front resulting fast convergence of *MOGA*. For every odd generation of *MOGA* or if for any generation previous generation of *MOGA* does not produce Pareto chromosomes greater than 2 then population is created using the procedure illustrated in section B. This introduces new chromosomes in population and induces diversity in the population. Population size also varies from generation to generation.

V. TESTING

Performance of *MOGA K*-clustering is given for 10 popular data sets from UCI machine learning repository [16]. While testing, tuples of the test set are provided with class labels. In a chromosome, every cluster is assigned a class label based on majority voting method. Predicted class label and the actual class label are used to build confusion matrix or matching matrix [8] for each chromosome in Pareto population. Each tuple t_j where $1 \leq j \leq m$ and m = total number of tuples in test set are assigned to a cluster based on $d_{lowest}(CC_i,t_j)$ where $1 \leq i \leq K$ and K = number of clusters to get predicted class label. From confusion matrix, performance of classifying all t_j is measured. Among all chromosomes in Pareto population, the one giving highest performance in classification is recorded after each run of proposed *MOGA*. For every data set, performances of 10 runs are recorded.

TABLE I. CLUSTERING PERFORMANCES IN TERMS OF CLASSIFICATION ACCURACIES OF *MOGA K*-CLUSTERING

Data set	***Accuracies***	***Generation of MOGA***
Blood Transfusion	55.68%	500
Diabates(Pima)	68.17%	500
Glass	42.9%	1000
Heart	78.70%	1000
Ionosphere	71.48%	1000
Iris	93.13%	1000
Sonar	55.82%	1000
Vehicle	38.13%	500
Wine	94.27%	1000
Zoo	69.21%	1000
Average	66.75%	

Average values of performances are shown in the Table I. Required time to run *MOGA* depends on the number of generations of *MOGA*. To keep the time required within the limit, in case of some data sets we set it to 500. For other data sets it is 1000.

VI. CONCLUSIONS

In this work we have carried out a novel real coded elitist *MOGA* for *K*-clustering as it is known that elitist model of *GAs* provide the optimal string as the number of iterations goes to infinity [3]. As infinite numbers of *GA* generations are not possible, obtained solutions are near optimal. We have achieved one important data mining task of clustering by finding cluster centers. Only continuous features are considered in this work. Euclidean distance measures are not suitable for categorical features and need other encoding. Dealing with missing features values and unseen data are other problem areas. It may be interesting to modify *MOGA* based *K*-clustering for categorical features and an unknown number of clusters.

REFERENCES

[1] P. Arabie, L. J. Hubert, and G. D. Soete, Eds. Clustering and Classification. Singapore: World Scientific, 1996.

[2] S. Bandyopadhyay, U. Maulik, and A. Mukhopadhyay, "Multiobjective Genetic Clustering for Pixel Classification in Remote Sensing Imagery," IEEE Trans. Geosci. Remote Sensing, vol. 45, no. 5, May. 2007, pp. 1506–511, doi:10.1109/TGRS.2007.892604.

[3] D. Bhandari, and C. A. Murthy, "Genetic algorithm with elitist model and its convergence," Int. J. Patt. Recog. Art. Intel., vol. 10, no. 6, Jul. 1996, pp. 731-747, doi:10.1142/S0218001496000438.

[4] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, 2nd Edition, New York: Springer-Verlag, 2007.

[5] D. Cvetković, I. C. Parmee, and E. Webb, "Multi-objective optimisation and preliminary airframe design," in Adaptive Computing in Design and Manufacture: The Integration of Evolutionary and Adaptive Computing Technologies with Product/System Design and Realisation, I. C. Parmee, Ed. New York: Springer-Verlag, 1998, pp. 255–267.doi: 10.1.1.52.7144.

[6] K. Deb, Multi-Objective Optimization Using Evolutionary Algorithms. New York: Wiley, 2001.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput., vol. 6, no. 2, Apr. 2002, pp. 182–197. doi:10.1109/4235/996017

[8] S. K. Dogra, "Confusion Matrix," QSARWorld-A Strand Life Sciences Web Resource. http://www.qsarworld.com/qsar-ml-confusion-matrix.php.

[9] B. Everitt, Cluster Analysis. New York: Halsted Press, 1980.

[10] J. H. Friedman, "Data mining and statistics: What's the connection?," in Proc. of the 29th Symposium on the Interface: Computing Science and Statistics, Houston, Texas, 1997, pp. 3–9. doi:10.1.1.81.6162.

[11] D. E. Goldberg, Genetic Algorithms for Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley, 1989.

[12] J. Han, and M. Kamber, Data Mining: Concepts and Techniques. San Francisco, CA: Morgan Kaufmann, 2006.

[13] J. Handl, and J. Knowles, "Exploiting the Trade-off - The Benefits of Multiple Objectives in Data Clustering," in Proc. of 3rd Int. Conf. on Evolutionary Multi-Criterion Optimization (EMO 2005), Guanajuato, Mexico, 2005, pp. 547–560. C. A. Coello Coello, A. H. Aguirre, and E. Zitzler Eds., in Lecture Notes in Computer Science, vol. 3410, chap. 38, pp. 547–560, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. doi: 10.1007/978-3-540-31880-4_38.

[14] J. Handl, and J. Knowles, “Multi-Objective Clustering and Cluster Validation,” in Multi Objective Machine Learning, Y. Jin, Ed., in Studies in Computational Intelligence, vol. 16, chap. 2, pp. 21–47, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, doi:10.1007/3-540-33019-4_2.

[15] J. Handl, and J. Knowles, “An evolutionary approach to multiobjective clustering,” IEEE Trans. Evol. Computat., vol. 11, no. 1, Feb. 2007, pp. 56–76, doi:10.1109/TEVC.2006.877146.

[16] S. Hettich, C. Blake, and C. Merz, “UCI repository of machine learning databases,” 1998. [Online]. Available: http://www.ics.uci.edu/ ~mlearn/MLRepository.html.

[17] J. Holland, Adaptation in Natural and Artificial Systems. Cambridge, MA: MIT Press, 1992.

[18] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering: A review,” ACM Comput. Surveys, vol. 31, no. 3, Sep. 1999, pp. 264–323. doi:10.1145/331499.331504.

[19] H. Jutler, “Liniejnaja modiel z nieskolmini celevymi funkcjami (linear model with several objective functions),” Ekonomika I Matematiceckije Metody, (In Polish), vol. 3, no. 3, pp. 397–406, 1967.

[20] L. Kaufman, and P. J. Rousseeuw, Finding Groups in Data. New York: Wiley, 1990.

[21] J. D. Knowles, and D. W. Corne, “Approximating the nondominated front using the Pareto archived evolution strategy,” Evol. Comput., vol. 8, no. 2, Summer, 2000, pp. 149–172. doi:10.1162/106365600568167.

[22] E. E. Korkmaz, J. Du, R. Alhajj, and K. Barker, “Combining advantages of new chromosome representation scheme and multi-objective genetic algorithms for better clustering,” Intell. Data Anal., vol. 10, no. 2, Mar. 2006, pp. 163–182.

[23] M. H. Law, A. P. Topchy, and A. K. Jain, “Multiobjective data clustering,” in Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit., Washington, DC, 2004, pp. 424–430. doi:10.1.1.114.2994.

[24] U. Maulik, and S. Bandyopadhyay, “Genetic algorithm-based clustering technique,” Pattern Recognit., vol. 33, no. 9, Sep. 2000, pp. 1455–1465. doi:10.1.1.19.1878.

[25] P. Merz, and A. Zell, “Clustering gene expression profiles with memetic algorithms,” in Proc. of the 7th Int. Conf. on Parallel Problem Solving from Nature (PPSN VII), Granada, Spain, pp. 811–820. J. J. Merelo Guervós, P. Adamidis, H. G. Beyer, J. L. F. Martín, and H. P. Schwefel Eds., in Lecture Notes in Computer Science, vol. 2439, pp. 811–820, Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. doi: 10.1.1.9.5447.

[26] Z. Michalewicz, Genetic Algorithms Data Structures Evolution Programs. Berlin, Germany: Springer-Verlag, 1992.

[27] H. Pan, J. Zhu, and D. Han, “Genetic algorithms applied to multi-class clustering for gene expression data,” Genomics, Proteomics, Bioinformatics, vol. 1, no. 4, Nov. 2003, pp. 279–287.

[28] V. Pareto, Manuale di Economia Politica. Milan: Piccola Biblioteca Scientifica, 1906. Translated into English by A. S. Schwier, and A. N. Page, Manual of Political Economy, London: Kelley Publishers, 1971.

[29] D. Pyle, Data Preparation for Data Mining. San Mateo CA: Morgan Kaufmann, 1999.

[30] K. S. N. Ripon, and M. N. H. Siddique, “Evolutionary multi-objective clustering for overlapping clusters detection,” in Proc. of the 11th Conf. on Congress on Evolutionary Computation (CEC 2009), Trondheim, Norway, 2009, pp. 976–982. doi:10.1109/CEC.2009.4983051.

[31] J. D. Schaffer, “Some experiments in machine learning using vector evaluated genetic algorithms (artificial intelligence, optimization, adaptation, pattern recognition),” Ph.D. dissertation, Dept. Elect. Eng., Vanderbilt Univ., Nashville, TN, 1984.

[32] J. D. Schaffer, “Multiple objective optimization with vector evaluated genetic algorithms,” in Proc. Int. Conf. Genetic Algorithms and their Applications, Pittsburgh, PA, 1985, pp. 93–100.

[33] E. Zitzler, “Evolutionary algorithms for multiobjective optimization: Methods and applications,” Ph.D. dissertation, Swiss Federal Institute of Technology (ETH), Zurich, 1999. doi:10.1.1.39.9023.