International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:4, No:11, 2010

Multidimensional Data Mining by Means of Randomly Travelling Hyper-Ellipsoids

Pavel Y. Tabakov, Kevin Duffy

Abstract—The present study presents a new approach to automatic data clustering and classification problems in large and complex databases and, at the same time, derives specific types of explicit rules describing each cluster. The method works well in both sparse and dense multidimensional data spaces. The members of the data space can be of the same nature or represent different classes. A number of N-dimensional ellipsoids are used for enclosing the data clouds. Due to the geometry of an ellipsoid and its free rotation in space the detection of clusters becomes very efficient. The method is based on genetic algorithms that are used for the optimization of location, orientation and geometric characteristics of the hyper-ellipsoids. The proposed approach can serve as a basis for the development of general knowledge systems for discovering hidden knowledge and unexpected patterns and rules in various large databases.

Keywords—Classification, clustering, data minig, genetic algorithms.

I. Introduction

WITH the advent of information and database technologies in various applications of modern business, the need to digest large volumes of data is now crucial. Certain predictions and data analysis are hardly possible when using classical statistical techniques and, in these cases, data mining is generally useful. The number of areas in which there is a need for data mining, and where it can be usefully and successfully applied, is growing all the time. These techniques are used in a wide variety of applications: sales and marketing, insurance companies, finance, planning and scheduling, optimization, forecasting, network and system management, etc. Generally speaking, any technique that helps to extract more knowledge hidden in data is useful, so data mining incorporates such methods as query tools, statistical techniques, visualization, on-line analytical processing, case-based learning, decision trees, association rules, neural networks, genetic algorithms, etc. The reader is referred to references [2], [5] for further discussions on standard data mining techniques and applications.

Probably the first recorded scientific application of data clustering dates back to the early 1900s when astronomers tried to understand the relationship between the luminosity of stars and their temperatures. Astronomers Hertzsprung and Russell plotted a two-dimensional graph with the vertical scale representing luminosity in multiples of the brightness of our sun and temperature on the horizontal scale. As a result three main clusters were discovered- white dwarfs, red giants and the main sequence. Presently clustering is an efficient tool to work with a large, complex data set with many variables and intricate internal structure. Every so often clustering is the first technique to start with, before other tools can be applied. The most commonly used method of cluster detection is the K-means method (see for example [11]). It has many variations, which are simple and quite reliable. It is easy to use when the feature vectors (members) of a data space are of the same class or, in other words, have some kind of natural association. There are many other clustering techniques for use in various disciplines such as biology, geology, archeology, geography, marketing and many others. A comprehensive review of different clustering techniques and publications on the subject can be found in [14].

Notwithstanding that there is a number of different clustering techniques available, the main difficulty of their application is not to single out a cluster but rather to decipher its intrinsic internal nature (see for example [1]). Another difficult problem is detection of proper cluster boundaries. For example, the K-means method does not detect boundaries well and can fail completely in higher dimensions. The clustering technique proposed in this paper eliminates this problem. It is very flexible and can easily cluster the heterogeneous data in multi-dimensional space, separating one cluster or set from another because such clusters are in the shape of a hyper-ellipsoid and have proper boundaries. The ability of the ellipsoid to arbitrarily change its shape while freely rotating in the space makes the clustering process very efficient. Among other advantages is the ability to regulate the cluster density and its dimension and orientation relative to the global coordinate system, etc. However, the biggest advantage is the ability to represent each cluster in the form of an explicit rule. This rule can be either simple, when the boundary conditions and coordinates of the data are presented as one logical or fuzzy-logical expression, or compound, when the rule can be expressed in the form of a decision tree. In some cases the cluster can be subdivided into smaller segments, or even clusters resulting in a set of explicit rules.

We live in a three-dimensional world and therefore the algorithms for the derivation of N-dimensional rotation matrices and the calculation of volume of the hyper-ellipsoids are not easily found in the literature. The published material on the subject is mostly theoretical without any practical application. In the present paper the reader will find simple and easy to understand algorithms for both the derivation of rotation matrices and the calculation of N-dimensional geometrical characteristics of an ellipsoid.

The method allows us to obtain explicit rules describing

P.Y. Tabakov is with the Department of Mechanical Engineering, Durban University of Technology,South Africa; e-mail: pashat@dut.ac.za
K. Duffy is with the Institute of System Science, Durban University of Technology, South Africa; e-mail: kevind@dut.ac.za

each cluster. When new data are added to the space these rules can also be used for analysis without re-running the whole program. Data clustering is an optimization problem and a genetic algorithm is used.

II. Problem Formulation

Every entry in a database can be considered as an ordered n-tuple $(x_1, x_2, \dots, x_n)$ of real numbers in dataspace $\Re^n$ as the point X or vector x representing the coordinates of the point. All the points corresponding to all the values of the co-ordinates are said to form an n-dimensional Euclidean space denoted by $\Re^n$. Several or all of the co-ordinates may be normalized in range to ensure a one-one correspondence between the points in the space and the sets of co-ordinates.

In the general case $\Re^n = \Re_1 \bigcup \Re_2 \bigcup, \dots, \bigcup \Re_k$, where $\Re_i$ represents a region that contains the points of set i while k is the number of sets in the space. One set represents the data of a similar nature, in other words, all the points in the region belong to the same class. A distinction is made between a space occupied by only one set of data and when there are two or more different sets in the space. In the latter case we might want to find not only explicit rules for each cluster but also to separate data of one set from another, i.e. solve a classification problem.

Clustering is considered to be the task of finding such regions which are characterized by high population density relatively to the surrounding regions. In clustering there are no predefined classes whereas in classification the population is subdivided by assigning each element or record to a predefined class.

The approach proposed here allows the performance both of tasks, clustering and classification (data separation) as well as the ability to combine them easily. Moreover, if necessary, it is possible to do classification within already detected clusters. Obviously, in this case the cluster is detected on the basis of density or compactness and some other geometric characteristics. Because of the flexibility of the method, with the help of the obtained rules, we can determine quite accurately the membership of any new record given once the problem of classification is solved. For the classification problem there are no effective methods that are efficient for arbitrary geometry to extract rules explicitly. Artificial neural networks of different types are used, see for example [16]. Unfortunately, extracting explicit rules from neural networks is not simple, see for example [13], and moreover, the number of these rules are usually large. Another approach is the application of evolving neural or fuzzy-neural networks, see [15]. Unfortunately, although these will provide explicit rules, their generalization is not good, as they are not globally optimized. One can employ fuzzy logic systems to extract the rules, but the number of rules will also generally be large. For example, consider the feature vectors (points) in Fig. 1 have three components and it should be possible to divide the whole region $\mathcal{V}$ into a number of parallelepipeds where each parallelepiped contains only data from one class. Having a complex distribution of feature vectors will result in a large number of rules and this will also impact on the generalization ability of these rules. Moreover,

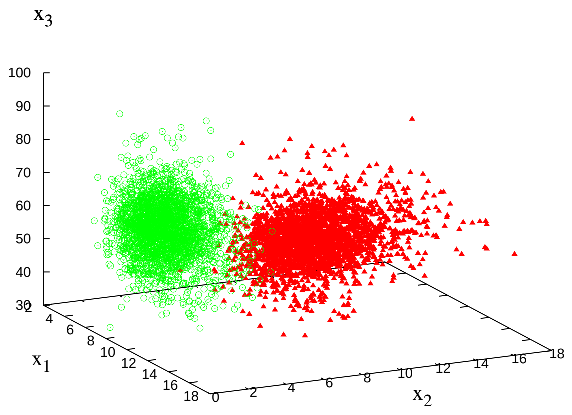

Fig. 1. Data distribution in a three-dimensional space

the approach is hard to implement if the space dimension is larger than three.

In the present study genetic algorithms (GAs) are used to construct the explicit rules for the membership recognition of each data type. This allows us to use the proper geometry of the solution, which cannot be easily implemented by the other methods. In our case a tournament selection is used together with a two-point crossover technique.

Genetic algorithms are search algorithms that simulate the mechanics of natural genetics for artificial systems based on natural selection, see for example [8], [10], [12]. These methods have been derived based on Darwin's theory of evolution, which can be considered as a process of continuous change from one state of an organism to another state of the evolved organism. The state of evolved organism can be better or worse. If the state of the organism degrades, the organism has less chance to survive. Thus, natural selection, where the fittest individuals have the highest probability of survival, is the fundamental concept in GA theory.

III. Clustering

Clustering can be considered as an optimization problem where the density of points is maximized subject to some boundary conditions. Both for clustering and data separation (the classification problem) the same approach is used. A geometric form (shape) that will ensure the highest possible density of feature points in the closed volume is selected. At the same time this geometric shape must be very flexible in space and should be represented mathematically as simply as possible. It is quite obvious that an ellipsoid can meet these requirements since it can be easily adjusted by changing the centre, orientation and dimension of its axes. This allows for the flexible adjustment of ellipsoids to cover the desired domain in the best way. Each such N-dimensional ellipsoid clearly describes one cluster via its location, density and number of enclosed points. A population of such hyper-ellipsoids randomly searches the space and guarantees the

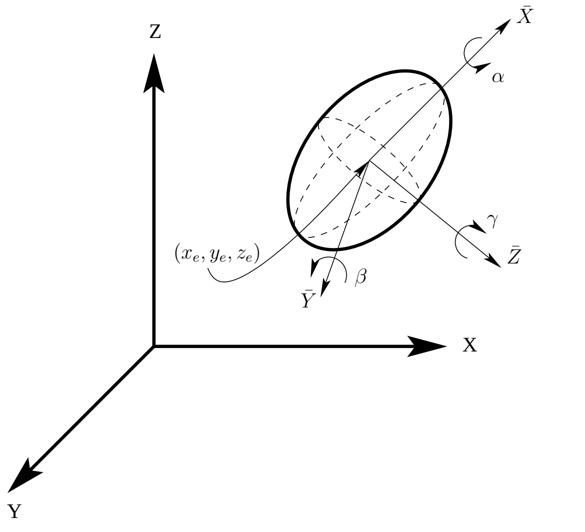

Fig. 2. Ellipsoid for data clustering/classification in a three-dimensional space

best coverage possible. Also, due to the nature of the fitness functions, 'the convergence' of the algorithm is pretty fast.

With the help of genetic algorithms the optimized parameters of all ellipsoids–clusters in the feature space can be obtained. For example, for the three-dimensional case, these parameters are the co-ordinates of the centre of the ellipsoid x_e, y_e, z_e, the lengths of semi-axes a, b, c and the orientation angles in the space α, β, γ (Fig. 2). As we are solving an optimization problem we need to determine the boundary conditions and formulate the cost function. The boundary conditions depend on the particular problem and include such parameters as the geometrical limitations of the ellipsoid and density of the data cloud. If the estimation of the required density presents some difficulties, then the data space can be analysed with spheres of fixed preselected dimensions. This procedure will give us a general understanding of the data distribution in the space. Now we maximise the volume of each ellipsoid until the density inside reaches some minimum predetermined value.

The foundation of a GA is its population, and the efficiency of the algorithm depends directly on how successfully the population is organised. Each population consists of individuals. One or more chromosomes may be required to specify an individual. An individual is coded by a string and the string is formed by a number of sub-strings or chromosomes. Each chromosome represents one optimising variable and consists of unchanged individual structures called genes. The location of the gene within the chromosome determines one particular characteristic. The initial population is generated using a random operator. A generation (or population) as an array of type *individual* is created. Using pseudo–code the type *individual* can be presented as a structure as follows:

```
Struct individual{
  int chromosome[];
  float VolumeFitness;
  float DensityFitness;
  float ParameterSet[];
  int BegXSite,
      EndXSite;//crossover points
  int IsPointIn;
                    };
individual OldPopulation[PopSize],
           SubPopulation[PopSize],
           Newpopulation[PopSize];
```

Here the array *chromosome* is a binary string representing one individual and the individual represents one rule.

The standard equation of a second order hyper-surface in Euclidean space can be represented as

$$\lambda_1 x_1^2 + \lambda_2 x_2^2 + \ldots + \lambda_n x_n^2 = H \tag{1}$$

where $\lambda_1, \ldots, \lambda_n$ are characteristic roots. If $\lambda_1, \ldots, \lambda_n$ and H are all of the same sign, then such a hyper-surface is called an N-dimensional ellipsoid or (for $n > 3$) hyper-ellipsoid and its equation can be rewritten in terms of the semi-axes a_i as

$$\frac{x_1^2}{a_1^2} + \frac{x_2^2}{a_2^2} + \ldots + \frac{x_n^2}{a_n^2} = 1 \tag{2}$$

In the special case when $a_1 = a_2 = \ldots = a_n = a$ the surface is called a hyper-sphere.

We define the density fitness for one ellipsoid-rule as follows:

$$f_d = \frac{p_{in}}{V_e} \tag{3}$$

where p_{in} is the number of feature points within the ellipsoid and V_e is the volume of the (hyper-)ellipsoid. In general, the volume can be represented as

$$V_e = \xi_n \prod_{i=1}^{n} a_i \tag{4}$$

The coefficients $\xi_1, \ldots, \xi_n$, depend on the space dimensionality, can be calculated with the help of the *gamma function*:

$$\Gamma(\mu) = \int_0^\infty x^{\mu-1} e^{-x} dx \text{ and } \xi_n = \frac{\sqrt{\pi^n}}{\Gamma\left(\frac{n+2}{2}\right)} \tag{5}$$

Integrating by parts, the value of the gamma function can be easily determined as

$$\Gamma\left(\frac{\mu}{2}\right) = \left(\frac{\mu}{2} - 1\right)! \tag{6}$$

if μ is even, otherwise, taking into account that $\Gamma\left(\frac{1}{2}\right) = \sqrt{\pi}$ and $\Gamma(\mu + 1) = \mu!$ the recursion formula should be used:

$$\Gamma(\mu) = (\mu - 1)\Gamma(\mu - 1) \tag{7}$$

By doing this the volume of the N-dimensional hyper-ellipsoid or hyper-sphere can easily be calculated. Thus, $\xi_2 = \pi$, $\xi_3 = (4/3)\pi$, $\xi_4 = (1/2)\pi^2$, $\xi_5 = (8/15)\pi^2$ and so on.

The *selection* operation of the genetic algorithm is based on the density of the cluster, but only after the volume of the ellipsoid is optimized. However, for classification this is not as important, which will be shown later. The algorithm to calculate the number of points inside the ellipsoid is as follows:

a) Determine the local co-ordinate system of the ellipsoid. Let $x_1, x_2, \ldots, x_n$ and $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n$ be the coordinates of the feature point in the global and local coordinate systems, respectively:

$$x_i = \sum_{k=1}^{n} t_{ik}\bar{x}_k, \qquad (i = 1, 2, \dots, n) \tag{8}$$

The above formula determines the transformation from one basis to another. The matrix

$$T = ||t_{ik}||_1^n \tag{9}$$

is called the matrix of the coordinate transformation and the inverse transformation can be written as

$$\bar{x} = T^{-1}x \tag{10}$$

Anyone interested in more detail on the subject is referred to [9].

While two- and three-dimensional rotation matrices are well described in the literature, higher dimensions are not considered often and mostly analyzed only from a theoretical point of view. The theory involved is very complex and difficult to understand. Nevertheless, there are real applications of multidimensional spaces, for example in such fields as chemistry and physics, and more recently in data mining. Next, without going into theoretical details, we shall show how easy an N-dimensional rotation matrices can be derived, though the interpretation of them might not be as easy.

Every linear homogeneous transformation of N-dimensional Euclidean space can be obtained by carrying out a succession of rotations in any order [9]. In three-dimensional space, rotations at angles θ_1, θ_2 and θ_3 about the x_1, x_2, x_3 axes in a clockwise direction when looking towards the origin produce the following matrices (trigonometric functions sin and cos abbreviated to s and c, respectively):

$$T_{x_1}^{(3)}(\theta_1) = \begin{vmatrix} 1 & 0 & 0 \\ 0 & c & s \\ 0 & -s & c \end{vmatrix}; \quad T_{x_2}^{(3)}(\theta_2) = \begin{vmatrix} c & 0 & s \\ 0 & 1 & 0 \\ -s & 0 & c \end{vmatrix}$$

$$T_{x_3}^{(3)}(\theta_3) = \begin{vmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{vmatrix} \tag{11}$$

It is obvious that with the rotation of only one axis the rest of the axes change their position. In the case of three dimensions two axes simultaneously change their position and this is why the two-dimensional rotation matrix is embedded here. This approach is used for higher dimensions, - the three-dimensional matrix is embedded in every four-dimensional matrix, the four-dimensional matrix is used for the derivation of the five-dimensional, and so on.

The composite rotation matrix $T^{(3)}$ can now be calculated as a product of the above three matrices, and thus a new co-ordinate system can be obtained:

$$T^{(3)} = T_{x_1}^{(3)} \times T_{x_2}^{(3)} \times T_{x_3}^{(3)} = \begin{vmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{vmatrix} \tag{12}$$

Now we extend the three-dimensional rotation to the four-dimensional rotation by adding unity as a diagonal element:

$$\begin{aligned} T_{x_1}^{(4)}(\theta_1) &= \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & t_{11} & t_{12} & t_{13} \\ 0 & t_{21} & t_{22} & t_{23} \\ 0 & t_{31} & t_{32} & t_{33} \end{vmatrix} \\ T_{x_2}^{(4)}(\theta_2) &= \begin{vmatrix} t_{11} & 0 & t_{12} & t_{13} \\ 0 & 1 & 0 & 0 \\ t_{21} & 0 & t_{22} & t_{23} \\ t_{31} & 0 & t_{32} & t_{33} \end{vmatrix} \\ T_{x_3}^{(4)}(\theta_3) &= \begin{vmatrix} t_{11} & t_{12} & 0 & t_{13} \\ t_{21} & t_{22} & 0 & t_{23} \\ 0 & 0 & 1 & 0 \\ t_{31} & t_{32} & 0 & t_{33} \end{vmatrix} \\ T_{x_4}^{(4)}(\theta_4) &= \begin{vmatrix} t_{11} & t_{12} & t_{13} & 0 \\ t_{21} & t_{22} & t_{23} & 0 \\ t_{31} & t_{32} & t_{33} & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \end{aligned} \tag{13}$$

Furthermore, the four-dimensional rotation matrix can be obtained as a product of the above four matrices $T^{(4)} = T_{x_1}^{(4)} \times T_{x_2}^{(4)} \times T_{x_3}^{(4)} \times T_{x_4}^{(4)}$, or for any dimension n we can write

$$T^{(n)} = \prod_{i=1}^{n} T_{x_i}^{(n)}(\theta_i^{(n)}) \tag{14}$$

The rotation matrix T is a special orthogonal matrix that possess some specific properties and the most important are the following:
1)$TT^T = I$, where I is the identity matrix;
2) $DetT = 1$;
3) the sum of squares of the elements in any row or columns equals 1;
4) the dot product of any pair of rows or any pair of columns is 0.

It should be realized that as the dimensionality of the space increases the rotation matrices become very complicated to comprehend, although relatively easy to compute. In this case only a linear transformation of the co-ordinates might be used, and though not so flexible, will do the job. Alternatively, a set of N-dimensional spheres can be used instead, if a high accuracy is not required. These simplifications also considerably reduce the length of the chromosome string and thus make the calculations much faster.

b) calculate the position of the point as

$$pos = \sum_{i=1}^{n} \frac{\bar{x}_i^2}{a_i^2} \tag{15}$$

If $pos \leq 1$, then the point is within the ellipsoid.

IV. Classification problem

Let us consider the N-dimensional space described above one more time. Our goal is to find explicit rules that separate different sets of data in the space. In the simplest case we might want to separate two different sets of data. Let $\mathfrak{R}_1$ and $\mathfrak{R}_2$ represent regions that contain the points of the first and second group, respectively, then $\Re^n = \mathfrak{R}_1 \bigcup \mathfrak{R}_2$. In the general case the space $\Re^n$ can be divided into three non-overlapping subspaces $\Re^1$, $\Re^2$ and $\Re^3$ ($\Re^n = \bigcup_{i=1}^{3} \Re^i$) where $\Re^1 \subseteq \mathfrak{R}_1$, $\Re^2 \subseteq \mathfrak{R}_2$ and $\Re^3 \in (\mathfrak{R}_1 \bigcap \mathfrak{R}_2)$. In the case when $\Re^1 \neq 0$ then, in principle, only one rule is sufficient to cover all feature vectors from $\Re^1$. However, the geometry of the distribution of the feature vectors constrains the efficient analytical formulation of such a rule. This geometry and convenience of available analytic descriptions of the region $\Re^1$ will determine the number of rules necessary to cover it. An analogous situation is when $\Re^2 \neq 0$. The problem we face is how to obtain as few as possible explicit (and hopefully simple) rules using known or training sets of feature points. Employing these rules we could determine the membership of any new feature point given. It is obvious that the main difficulty here is the subspace $\Re^3$ where the "overlapping" of class domains occurs.

The implementation of the genetic algorithm is slightly different to the clustering problem. The specific feature of it is the simultaneous use of two fitness values, real and integer. The former one is used only for the selection procedure and the latter one for the rest of the calculations. We are not required anymore to evaluate the cluster density. For every ellipsoid-rule we define the normalized fitness as follows:

$$f_n = \frac{P_{\mathfrak{R}_1} - P_{\mathfrak{R}_2}}{P_{\mathfrak{R}_1} + P_{\mathfrak{R}_2}} \qquad (16)$$

where $P_{\mathfrak{R}_1}$ is the number of points from set $\mathfrak{R}_1$ contained within the ellipsoid, while $P_{\mathfrak{R}_2}$ is the number of points from set $\mathfrak{R}_2$ in the ellipsoid. In the case of more than two different sets in the space, $P_{\mathfrak{R}_2}$ will represent the number of all unwanted points in the ellipsoid. Nevertheless, we are interested only in those individuals that do not have any unwanted points enclosed in the ellipsoid, and for this purpose we use an integer value of fitness. This fitness is equal to the number of points from the desired set in the ellipsoid providing that there are no points from any other set inside the ellipsoid. This fitness is used and optimized throughout the algorithm except in the selection procedure. The above approach makes the selection process and the overall performance of the algorithm more effective.

V. Numerical Results

The proposed technique was used for discovery of specific types of rules related to detection and extraction of explicit potentially biologically active DNA motifs from nucleotide databases. The detailed description and discussion of both the problem and obtained results can be found in [18]. The characteristic of these rules is that they represent a relation of the strength of signals of two motifs and their mutual distance. In the particular case the rules govern the relationship of the *TATA*–box motifs in eukaryots, the signal that relates to the $[-40, +11]$ region relative to the transcription start site *TSS* of eukaryotic promoters, and the distance of the *TATA* motifs and *TSS*.

It is necessary to stress that recognition of the eukaryotic promoters and thus *TSS* is an extremely difficult problem which is not yet fully solved, partly because of incomplete biological knowledge as well as highly intricate structure of available databases [7], [17].

The total number of feature vectors used is 618 from set $\mathfrak{R}_1$ and 1758 from set $\mathfrak{R}_2$ and were taken from [6]. However, only 75% randomly chosen feature vectors from each set were used for the rule extraction (training sets), while the rest were used for the verification of the rules (test sets). Using 10 obtained rules with the largest coverage one can achieve about 63.3% correct recognition and 0.9% false recognition on the training set. In order to prevent the proliferation of rules that cover only single examples, for which it is reasonable to expect that they will not generalize properly, the total number of extracted rules was limited to 75. Applying all 75 rules to the training sets we obtained 89.56% correct recognition and 0.61% false recognition. Then these rules were applied to to the test sets where 77.21% correct and 1.12% false recognition was achieved. When all 75 rules were applied to to the original full sets $\mathfrak{R}_1$ and $\mathfrak{R}_2$, we obtained 86.4% correct recognition and 0.74% false recognition.

It is difficult to compare these results with those obtained by other researchers mainly because different researchers used different training and test sets, different criteria to discern when the reported prediction by the program indicates a correct promoter and generally, a different methodology for assessing prediction accuracy. Nevertheless, in order to give a rough idea about the accuracy of the proposed rule-extracted system in relation to the reported results of some other promoter prediction programs, we compared our results with a great number of those available in the literature, related to the same *TATA*–box motif. Despite considerable effort to find an effective computational tool most of available results are lack accuracy. The highest accuracy found for a similar problem is 77% in the training set was reported in [3]. To appreciate a complexity of this problem as well as find the discussion on various techniques available and their performance see, for example, [4], [19].

VI. Conclusion

Automatic data clustering is an effective tool to manage large and complex databases with many variables and intricate internal structure. While other techniques are often necessary for a full analysis it is usually efficient to start by clustering the data. Often there is a need to classify the data as well and this can be done independently or in combination with the clustering. The method proposed here can be useful in extracting explicit rules for both data clustering and classification in various large complex databases. As with any method, its efficiency depends on the database considered. The proposed approach is expected to be very efficient since it involves a specific geometry of optimization set up for the specific density of clusters and maximum possible volumes of the ellipsoids. This clustering approach has been used successfully by others [14] but the significance of this paper is the extension of this approach to higher dimensions with an easily implemented mathematical formalism.

For classification the aim is to minimize the number of rules that allow maximal separation of different classes of feature vectors (points). Each rule is represented by a specific N-dimensional ellipsoid in the feature (data) space.

The advantage of the method proposed here is its ability to work in multidimensional data spaces as well as to separate different data classes, even if the distribution of the data is highly complex.

REFERENCES

[1] S.S.R. Abidi, K.M. Hoe and A. Goh, “Analyzing Data Clusters: A Rough Set Approach to Extract Cluster-defining Symbolic Rules.” Lecture Notes in Computer Science 2189: Advances in Intelligent Data Analysis. Fourth International Conference (IDA-01), Cascais, Portugal, 2001.
[2] P. Adriaans and D. Zantinge, *Data Mining*, Addison-Wesley, England, 1997.
[3] S. Audic and J.M. Claverie, “Detection of eukaryotic promoters using Markov transition matrices”, *Computer Chemistry*, vol.21, no.4, pp. 223-227, 1997.
[4] V.B. Bajic, Sin Lam Tan, Yutaka Suzuki and Sumio Sugano, “Promoter prediction analysis on the whole human genome”, *Nature Biotechnology*, vol.22, pp. 1467-1473, 2004.
[5] M.J.A. Berry and G. Linoff, *Data Mining Techniques. For Marketing, Sales and Customer Support*, John Wiley & Sons, Inc., 1997.
[6] P. Bucher, “Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences ”, Journal of Molecular Biology, vol.212, pp. 563-578, 1990.
[7] J.W. Fickett and A.G. Hatzigeorgiou, “Eukaryotic promoter recognition”, Genom Research, vol.7, no.9, pp. 861-878, 1997.
[8] D.B. Fogel, *Evolutionary Computation* (Second edition), IEEE Press, New York, 2000.
[9] F.R. Gantmacher, *The Theory of Matrices*, Chelsea Publishing Company, N.Y., 1959.
[10] D.E. Goldberg, *Genetic Algorithms in Search, Optimisation, and Machine Learning*, Addison-Wesley, Reading, MA, 1989.
[11] J.A. Hartigan, *Clustering Algorithms*, John Wiley & Sons, 1975.
[12] J.H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, MI, 1976.
[13] E.R. Hruschka and N.F. Ebecken, “A Clustering Genetic Algorithm for Extracting Rules from Supervised Neural Network Models in Data Mining Tasks”, Int. Journal of Computers, Systems and Signals, vol.1, no.1, pp. 17-29, 2000.
[14] A.K. Jain, M.N. Murty and P.J. Flynn, “Data Clustering: A Review”, ACM Computing Surveys, vol.31, no.3, pp. 264-323, 1999.
[15] N. Kasabov, *Evolving Neural Networks*, MIT Press, 1996.
[16] S.Y. Kung, *Digital Neural Networks*, PTR Prentice Hall, Engelwood Cliffs, NJ, 1993.
[17] A.G. Pedersen, P. Baldi, Y. Chauvin and S. Brunak, “The biology of eukaryotic promoter prediction – a review”, Computers and Chemistry, vol.23, pp. 191-207, 1999.
[18] P.Y. Tabakov and V.B. Bajić, “Genetic Algorithms and Extraction of Rules for Detection of Short DNA Motifs”, Int. Journal of Computers, Systems and Signals, vol. 1, no. 1, pp. 106-117, 2000.
[19] Xiaowo Wang, Zhenyu Xuan, Xiaoyue Zhao, Yanda Li and Michael Q. Zhang, “ High-resolution human core-promoter prediction with CoreBoost_HM”, *Genome Research*, vol.19, pp. 266-275, 2009.