Implementation of Genetic Algorithm in Predicting Diabetes

S.Sapna[1], Dr.A.Tamilarasi[2] and M.Pravin Kumar[3]

[1] Assistant Professor,
Department of Master of Computer Applications,
K.S.R College of Engineering, Tiruchengode-637215
Tamilnadu, India

[2] Prof. & Head, Department of Master of Computer Applications,
Kongu Engineering College, Perundurai,
Tamilnadu, India

[3] Assistant Professor,
Department of Electronics and Communication Engineering,
K.S.R College of Engineering, Tiruchengode-637215
Tamilnadu, India

Abstract

Data Mining aims at discovering knowledge out of data and presenting it in a form that is easily compressible to humans. Data Mining represents a process developed to examine large amounts of data routinely collected. The term also refers to a collection of tools used to perform the process. One of the useful applications in the field of medicine is the incurable chronic disease diabetes. Data Mining algorithm is used for testing the accuracy in predicting diabetic status. Fuzzy Systems are been used for solving a wide range of problems in different application domain Genetic Algorithm for designing. Fuzzy systems allows in introducing the learning and adaptation capabilities. Neural Networks are efficiently used for learning membership functions. Diabetes occurs throughout the world, but Type 2 is more common in the most developed countries. The greater increase in prevalence is however expected in Asia and Africa where most patients will likely be found by 2030.

Keywords: Data Mining, Diabetes, Fuzzy Systems, Genetic Algorithm (GA), Neural Networks.

1. Introduction

Health and Commonwealth Government have identified diabetes to be a significant and growing global public health problem with the expected incidence in Australia to increase from 4% to 10% by 20101. An estimated 40 million Indians suffer from diabetes, and the problem seems to be growing at an alarming rate. By 2020, the number is expected to double and reach epidemic proportions, even as half the numbers of diabetics in India remain undiagnosed. Diabetes has debilitating consequences on many of the body’s vital organs if remained unchecked and controlled, the biggest problem being that of eyesight. It affects eyes, kidney, heart and every single vital organ of the body. India has the dubious distinction of being the diabetic capital of the world. Home to around 33 million people with diabetes, 19% of the world’s diabetic population is from India. Nearly 12.5% of Indian’s urban populations have diabetes. The number is expected to escalate to an alarming 80 million by the year 2030. Amongst the chronic diabetic complications, diabetic foot is the most devastating result. Over 50,000 leg amputations take place every year due to diabetes in India. Diabetes patients can often experience loss of sensation in their feet. Even the smallest injury can cause infection that can be various serious. 15% of patients with diabetes will develop foot ulcers due to nerve damage and reduced blood flow. Diabetes slowly steals the person’s vision. It is the cause for common blindness and cataracts. Cardiovascular diseases are rising. Nearly 3.8 crore cases were detected in 2005 and experts believe the number will go upto 6.4 crore by 2015.

Fuzzy Systems is used for solving a wide range of problems in different application domains. The use of Genetic Algorithms for designing Fuzzy Systems allows us to introduce the learning and adaptation capabilities. The topic has attracted considerable attention in the Computation Intelligence community. The paper briefly reviews the classical models and the most recent trends for Genetic Fuzzy Systems. Accurate and reliable decision making in oncological prognosis can help in the planning of suitable surgery and therapy, and generally, improve patient management through the different stages of the disease. To indicate that the reliable prognostic marker model than the statistical and artificial neural-network-based methods.

Genetic Algorithms (GAs) are considered as a global search approach for optimization problems. Through the proper evaluation strategy, the best “chromosome” can be found from the numerous genetic combinations. Although the GA operations do provide the opportunity to find the optimum solution, they may fail in some cases, especially when the length of a chromosome is very long. In this paper, a data mining-based GA is presented to efficiently improve the Traditional GA (TGA). By analyzing support and confidence parameters, the important genes, called DNA, can be obtained. By adopting DNA extraction, it is possible that TGA will avoid stranding on a local optimum solution. Furthermore, the new GA operation, *DNA implantation*, was developed for providing potentially high quality genetic combinations to improve the performance of TGA. Experimental results in the area of digital watermarking show that our data mining based GA successfully reduces the number of evolutionary iterations needed to find a solution.

Real-life data mining applications are interesting because they often present a different set of problems for data miners. One such real-life application that we have done is on the diabetic patients databases. In this paper, knowledge discovery on this diabetic patient database is discussed. A semi-automatic means for cleaning the diabetic patient database, and present a step-by-step approach to help the health doctors explore their data and to understand the discovered rules better. Generally in Asia about 47 percent of the population is diabetic. This disease has many side effects such as higher risk of eye disease, higher risk of kidney failure, and other complications. However, early detection of the disease and proper care management can make a difference. To combat this disease a regular screening program for the diabetic patients. Patient information, clinical symptoms, eye-disease diagnosis and treatments are captured into a database. This leads naturally to the application of knowledge discovery and data mining techniques to discover interesting patterns that exist in the data. The objective is to find rules that can be used by the medical doctors to improve their daily tasks, that is, to understand more about the diabetic disease.

2. Fuzzy Systems

Application of fuzzy sets theory was recognized in the field of medicine, the uncertainty found in the process of diagnosis of disease that has most frequently been the focus of applications of fuzzy set theory. The desire to better understand and teach this difficult and important process of medical diagnosis has prompted attempts to model it with the use of fuzzy sets, These models vary in the degree to which they attempt to' deal with different complicating aspects of medical diagnosis such as the relative importance of symptoms, the varied symptom patterns of different disease stages, relations between diseases themselves, and the stages of hypothesis formation, preliminary diagnosis, and final diagnosis within the diagnostic process itself. These models also form the basis for computerized medical expert systems, which are usually designed to aid the physician in the diagnosis of some specified category of diseases.

3. Genetic Algorithms

Genetic algorithm (GA) refers to a model introduced and investigated by John Holland in 1975 for adaptation processes of nature. Generally stated, a GA is any population based model that uses selection and recombination operators to generate new sample points in a search space. GA computationally utilizes a natural evolutionary process similar to the process first described by Charles Darwin in his "The Origin of Species", to solve a given problem. GA is a global search procedure that searches from one population of points to another. GA is a probabilistic search procedure, which is being frequently applied to difficult optimization and learning problems. There are two versions of the GA, namely the natural GA and the computational GA.

Genetic algorithms were inspired by the processes observed in natural evolution. They attempt to mimic these processes and utilize them for solving a wide range of optimization problems. In general, genetic algorithms perform directed random searches through a given set of alternatives with respect to the given criteria of goodness. These criteria are required to be expressed in terms of an objective function, which is usually referred to as a fitness function.

Genetic algorithms require that the set of alternatives to be searched through be finite. If we want to apply them to an optimization problem where this requirement is not satisfied, the set involved and select an appropriate finite subset. It is further required that the alternatives be coded in strings of some specific finite length which consist of symbols from some finite alphabet. These strings are called chromosomes, the symbols that form them are called genes, and their set is called a gene pool. Genetic algorithms search for the best alternative in the sense of a given fitness function through chromosomes evolution. Basic steps in genetic algorithms are shown in figure 1.

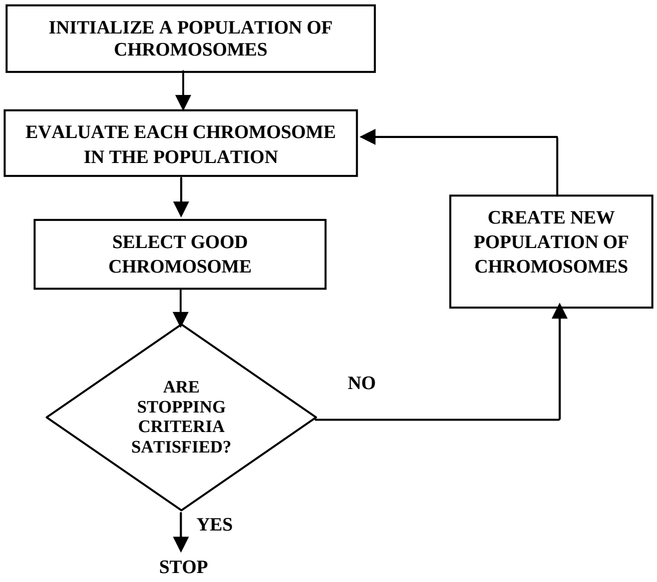
INITIALIZE A POPULATION OF CHROMOSOMES
EVALUATE EACH CHROMOSOME IN THE POPULATION
SELECT GOOD CHROMOSOME
CREATE NEW POPULATION OF CHROMOSOMES
ARE STOPPING CRITERIA SATISFIED?
NO
YES
STOP

Fig 1. Flowchart of Genetic Algorithms

Genetic Algorithms search for the best alternative (in the sense of a given fitness function) through chromosomes' evolution. Basic steps in genetic algorithms figure. First, an initial population of chromosomes is randomly selected. Then each of the chromosomes in the population is evaluated in terms of its fitness (expressed by the fitness function). Next, a new population of chromosomes is selected from the given population by giving a greater change to select chromosomes with the high fitness. This is called *natural selection*.

The new population may contain duplicates. If given stopping criteria (e.g., no change in the old and new population, specified computing time, etc.,) are not met, some specific, genetic – like operations are performed on chromosomes of the new population. These operations produce new chromosomes, called offspring's. The same steps of this process, evaluation and natural selection, are then applied to chromosomes of the resulting population. The whole process is repeated until given stopping criteria are met. The solution is expressed by the best chromosome in the final population.

There are many variations on these basic ideas of genetic algorithms. To describe a particular type a genetic algorithm is greater detail, let G denote the gene pool, and let n denote the length of strings of genes that form chromosome. That is, chromosomes are n - tuples in G^n. The size of the population of chromosomes is usually kept constant during the execution of genetic algorithm. That is, when new members are added to the population, the corresponding the umber of old members are excluded. Let m denote this constant population size. Since each population may contain duplicates of chromosomes, we express populations by m-tuples whose elements are n-tuples from the set G^n. Finally, let f denote the fitness function employed in the algorithm.

3.1 Genetic algorithm which is iterative, consists of the following six steps

1. Select an initial population, $P^{(k)}$, of a given size m, where k=1. This selection is made randomly from the set G^n. The choice of value m is important. If it is too large, the algorithm does not differ much from an exhaustive search; it is too small, the algorithm may not reach the optimal solution.
2. Evaluate each chromosome in population $P^{(k)}$ in terms if its fitness. This is done by determining for each chromosome x in the population the value of the fitness function, f(x).
3. Generate a new population $P_n^{(k)}$, from the given population $P^{(k)}$ by some procedure of natural selection. We describe only one possible procedure of natural selection, which is referred to as deterministic sampling. According to this procedure, we calculate the value e(x) = mg(x) for each x in $P^{(k)}$, where g(x) is a relative fitness defined by the formula.

$$g(k) = \frac{f(x)}{\sum_{n \in p^{(k)}} f(x)}$$

Then the number of copies of each chromosome x in $P^{(k)}$, that is chosen for $P_n^{(k)}$, is given by the integer part of e(x). If the total number of chromosomes chosen in this way is smaller than m (the usual case), then we select the remaining chromosomes for $P_n^{(k)}$ by the fractional parts of e(x), from the highest values down. In general, the purpose of this procedure is to eliminate chromosomes with low fitness and duplicate those with high fitness.

4. If stopping criteria are not met, go to step 5, otherwise stop.
5. Produce a population of new chromosomes $P^{(k+1)}$, by operating on chromosomes in population $P_n^{(k)}$. Operations that are involved in this step attempt to mimic genetic operations observed in biological systems. They include some or all the following four operations:

A. Simple Crossover

Given two chromosomes

$$x = (x_1, x_2, ..., x_n),\ y = (y_1, y_2, ..., y_n)$$

and an integer $i \in N_{n-1}$, which is called a crossover position, the operation of simple crossover applied to x and y replaces these chromosomes with their offspring's,

$$x' = (x_1, ..., x_i, y_{i+1}, ..., y_n),$$
$$y' = (y_1, ..., y_i, x_{i+1}, ..., x_n)$$

chromosomes x and y, to which this operation is applied, are called *mates*.

B. Double Crossover

Given the same chromosomes mates x,y as in the simple crossover and two crossover positions $i, j \in N_{n-1}(i < j)$, the operation of double crossover applied to x and y replaces these chromosomes with their offspring's,

$$x' = (x_1, ..., x_i, y_{i+1}, ..., y_{j,}\ x_{j+1}, ..., x_n),$$
$$y' = (y_1, ..., y_i, x_{i+1}, ..., x_{j,}\ y_{j+1}, ..., y_n)$$

C. Mutation

Given a chromosome $x = (x_1, x_2, ..., x_n)$ and an integer $i \in N_n$, which is called a mutation position, the operation of mutation replaces x with

$$x' = (x_1, ..., x_{i-1}, z,\ x_{i+1}, ...,\ x_n)$$

where z is a randomly chosen gene from the gene pool G.

D. Inversion

Given a chromosome $x = (x_1, x_2, ..., x_n)$ and two integers $i, j \in N_{n-1}(i < j)$, which are called inversion positions, the operation of inversion replaces x with

$$x' = (x_1, ..., x_i, x_j, x_{j-1}, x_{i+1}, x_{j+1}, ...,\ x_n)$$

E. Replace population $P_n^{(k)}$ with $P^{(k+1)}$ produced in Step 4, increase k by one, and go to Step 2.

3.2 Sample Problem

(a) k = 1 : Step 2 and 3

Chromosome in $P^{(1)}$	Integers	Fitness	g(x)	4g(x)	Number of Selected Copies
00010	2	3.75	0.068	0.272	0
01001	9	12.94	0.292	1.168	1
10011	19	15.44	0.350	1.400	2
11000	24	12.00	0.291	1.164	1

(b) k = 1 : Step 5

Chromosome in Pn(^{1})	Mate (randomly Selected)	Crossover Site (randomly Selected)	Resulting Chromosomes in P(^{2})
01001	10011	3	01011
10011	01001	3	10001
10011	11000	1	11000
11000	10011	1	10011

Similarly for the values of k 2,3, values are set to calculate the fitness value, mate and the crossover site.

(c) k = 2 : Step 2 and 3

Chromosome in $P^{(2)}$	Integers	Fitness	g(x)	4g(x)	Number of Selected Copies
01011	11	14.44	0.250	0.100	0
10001	17	15.94	0.276	1.104	2
11000	24	12.00	0.207	0.828	1
10011	19	15.44	0.267	1.068	1

(d) k = 2 : Step 5

Chromosome in Pn(2)	**Mate (randomly Selected)**	**Crossover Site (randomly Selected)**	**Resulting Chromosomes in P(3)**
10001	3	2	10000
10001	4	3	10011
11000	1	2	11001
10011	2	3	10001

(e) k = 3: Step 2 and 3

Chromosome in $P^{(3)}$	Integers	Fitness	g(x)	4g(x)	Number of Selected Copies
10000	16	16.00	0.274	1.096	1
10011	19	15.44	0.265	1.060	1
11001	25	10.94	0.188	0.752	1
10001	17	15.94	0.273	1.092	1

A crossover operation is employed in virtually all types of genetic algorithms, but the operations of mutation and inversion are sometimes omitted. Their role is to produce new chromosomes not on the basis of the fitness function, but for the purpose of avoiding a local minimum. This role is similar to the role of a disturbance employed in neural networks. If these operations are employed they are usually chosen with small probabilities. The mates in the crossover operations and the crossover positions in the algorithm are selected randomly. When the algorithm terminates, the chromosome in $P^{(k)}$ with the highest fitness represents the solution.

3.3 Natural Genetic Algorithm

The natural genetic algorithm is as follows :

- randomly generate an initial population M(O)
- loop

a. Compute and save the fitness u(m) for each individual m in current population M(t).
b. Define the selection probabilities p(m) for each individual m in M(t) (so that p(m) is proportional to u(m).
c. Generate M(k+1) by probabilistically selecting individuals from M(t) to produce a new population via genetic operators.

Fuzzy Genetic Algorithm can be implemented to check the patients affected by diabetes based upon the fitness value and the accuracy chromosome value.

3.4 Diabetes

Most of the food we eat is converted to glucose, or sugar which is used for energy. The pancreas secretes insulin which carries glucose into the cells of our bodies, which in turn produces energy for the perfect functioning of the body. When you have diabetes, your body either doesn't make enough insulin or can not use its own insulin as well as it should. This causes sugar to build up in your blood leading to complications like heart disease, stroke, and neuropathy, poor circulation leading to loss of limbs, blindness, kidney failure, nerve damage, and death.

3.4.1 General Symptoms of Diabetes

- Increased thirst
- Increased urination - Weight loss
- Increased appetite - Fatigue
- Nausea and/or vomiting - Blurred vision
- Slow-healing infections - Impotence in men

3.4.2 Types of Diabetes

Type 1 - Diabetes also called as *Insulin Dependent Diabetes Mellitus* **(IDDM)**, or *Juvenile Onset Diabetes Mellitus* is commonly seen in children and young adults however, older patients do present with this form of diabetes on occasion. In type 1 diabetes, the pancreas undergoes an autoimmune attack by the body itself therefore; pancreas does not produce the hormone insulin. The body does not properly metabolize food resulting in high blood sugar (glucose) and the patient must rely on insulin shots. Type I disorder appears in people younger than 35, usually from the ages 10 to 16.

Type II - Diabetes is also called as *Non-Insulin Dependent Diabetes Mellitus* (NIDDM), or *Adult Onset Diabetes Mellitus*. Patients produce adequate insulin but the body cannot make use of it as there is a lack of sensitivity to insulin by the cells of the body. Type II disorder occurs mostly after the 40.

Gestational Diabetes- Diabetes can occur temporarily during *Pregnancy* called as *Gestational Diabetes* which is due to the hormonal changes and usually begins in the fifth or sixth month of pregnancy (between the 24th and 28th weeks). Gestational diabetes usually resolves once the baby is born. However, 25-50% of women with gestational diabetes will eventually develop diabetes later in life, especially in those who require insulin during pregnancy and those who are overweight after their delivery.

3.4.3 Diagnostic Tests

- Urine Test
- Fasting Blood Glucose Level
- Post Prandial Blood Sugar
- Random Blood Glucose Level
- Oral Glucose Tolerance Test
- Glycosylated Hemoglobin (HbAlc)

4. Computation Genetic Algorithm

The major difference between the natural and computational GA is that at some point in the loop, termination conditions are checked and the process is terminated if a termination condition is reached. Genetic Algorithm (simplified): Step of the algorithm
a) Initialize the population
b) Calculate the population's fitness
c) While the number of generation is not the maximum number of generations do:
    a) Select all the solutions whose genetic material will propagate to the next generation
    b) Perform the crossover operation
    c) Perform the mutation operation
    d) Calculate the new population's fitness
    e) Get population statistics

A genetic algorithm begins with a collection of solutions to the problem being solved, called a population. Along with each individual in the population is an associated fitness value, or measure of solution quality. The larger the fitness value, the better the solution. With the initial population generated and the fitness calculated, the algorithm iterates through a series of five operations which progressively improves the quality of the solutions in the population. These steps are:

1. Selection

The algorithm spins a "roulette wheel" to randomly select two individuals from the population whose genetic material will propagate to the next generation. This roulette wheel is not a fair wheel - solutions with better fitness's are more likely to be chosen.

2. Crossover (recombination operators)

The pairs of solutions chosen through selection, certain randomly chosen pairs undergo crossover, i.e., have their genetic material combined to create a new pair of solutions. Each of the newly created solutions inherits characteristics from both parents and is placed in the new population. Those solutions not chosen to be combined through crossover are simply copied into the new population. As its name implies, a crossover operator forms new chromosomes by combining (generally) two chromosomes with a (usually) predetermined crossover probability, p_c. p_c depends on the problem and other parameters, but it is often taken about 70-80%. Crossover is the main search operator of GAs.

3. Mutation

After a new generation has been created through copy and crossover, a certain number of the new solutions are randomly chosen to experience mutation. This operation perturbs the gene pool by introducing new solutions not directly related to existing solutions through crossover.

4. Fitness

Next, all solutions in the new population have their fitness's calculated.

5. Population Statistics

The new population with its fitness values is evaluated to determine and record the best solution found to date during the execution of the genetic algorithm. One complete pass through all five of these steps is referred to as a generation and results in the creation of a new population of solutions, equal in size to the starting population. Once this is complete the new population is used as the starting point for the next iteration and the process is repeated. The computation concludes when either a certain number of generations has been completed or when a given solution quality has been reached. The new generated population will be equal in size to the starting population. The new population is used as the starting point for the next iteration and the entire process is repeated until some determined termination conditions are reached. These termination conditions are when a certain number of generations is generated or when a given solution quality is reached.

4.2 Generic Genetic Algorithm

```
Procedure GA_IPD_Run
Initialize_Population ($P_{old}$)
 // fills the chromosome of population $P_{old}$ with 0's and 1's randomly. while termination criteria not satisfied do for each chromosome $c_i$ in $P_{old}$ do
Evaluate ($c_i$, $P_{old}$)  // runs chromosome $c_i$ against every member of $P_{old}$ includes itself to compute fitness end
Generate_New_Population ($P_{new}$, $P_{old}$)
 // generate new population using
$P_{old}$ $P_{old}$ $\rightarrow$ $P_{new}$   end , end
```

4.3 Algorithm for Generating New Population

```
Procedure Generate _ New _ Population ($P_{old}$, $P_{New}$)
    $P_{New} \rightarrow 0$
while Size ($P_{New}$) < Size ($P_{old}$) do
// Selection
    $c_1 \leftarrow$ Select ($P_{old}$) $c_2 \leftarrow$ Select ($P_{old}$)
// Crossover
  if $P_c < r(.)$ then  // return random nos. in the interval (0,1)
   // $P_c$ : Crossover Probability  Crossover ($c_1$, $c_2$) // implements uniform crossover end  // Mutation
for i = 1 to chromosome_length do
  if $r(.) < P_m$ then // $P_m$ Mutation Probability
    // Chromosome swapping each bit at the
    corresponding position with fixed probability usually
    0.5 percent
        $c_{1i} \leftarrow c_{1i}$  // ith bit of the Ist chromosome
    end
```

if $r(.) < P_m$ then $c_{2i} \leftarrow c_{2i}$ end, end

$P_{New} \rightarrow P_{New} \lhd c_1 \lhd c_2$ // $\lhd$: Inserts the chromosome on the right hand side to the population to the left hand side.

5. Conclusion

Inorder to obtain the accuracy of chromosome and to evaluate the diabetes in diabetic patient GA is implemented. The connection between fuzzy systems and genetic algorithms is bidirectional. In one direction, genetic algorithms are utilized to deal with various optimization problems involving fuzzy systems. One important problem for which genetic algorithms have proven very useful is the problem of optimizing fuzzy inference rules in fuzzy controllers. In the other direction classical genetic algorithms can be fuzzified. The resulting fuzzy genetic algorithms tend to be more efficient and more suitable for some applications. Research on complex diseases only seems to be approaching the final goal, the prevention and cure of the diseases, very slowly. Diabetes is a disease in which the body does not produce or properly use insulin. Insulin is a hormone that is needed to convert sugar, starches and other food into energy needed for daily life. The cause of diabetes continuous to be ambiguous although both genetics and environmental factors such as obesity and lack of exercise. Symptoms of low blood sugar, side effects, science of complication are to be noted else it leads to severe problems. Using GA optimization of chromosome is obtained and based on the rate of old population diabetes can be restricted in new population to get chromosomal accuracy.

Reference

[1] Diabetes and you your guide to living well with diabetes, Novo Nordisk, LEAD GROUP.

[2] How to cut out all Diabetic Problems by 50% - The Alphabet way, Dr.Vinod Patel, Department of Diabetes and Endocrinology George Eliot Hospital, UK.

[3] Genetic mapping of complex traits: the case of Type 1 diabetes, Päivi Onkamo, Diabetes and Genetic Epidemiology Unit, Department of Epidemiology and Health Promotion, National Public Health Institute and Division of Biometry, Rolf evanlinna Institute and Finnish Genome Center Faculty of Science University of Helsinki Academic, 2002.

[4] Genetic Fuzzy Systems: Status, Critical Considerations and Future Directions, Francisco Herrera, International Journal of Computational Intelligence Research. ISSN 0973-1873 Vol.1, No.1 (2005), pp.59-67.

[5] A Fuzzy Logic Based-Method for Prognostic Decision Making in Breast and Prostate Cancers, Huseyin Seker, *Student Member, IEEE*, Michael O. Odetayo, Dobrila Petrovic, and Raouf N. G. Naguib, *Senior Member, IEEE*

[6] TRANSACTIONS on Information Technology in Biomedicine, Vol. 7, No. 2, June 2003.

[7] A Data Mining Based Genetic Algorithm, YI-TA WU1, YOO JUNG AN2, JAMES GELLER2 AND YIH-TYNG WU3, 2006 IEEE.

[8] Data Mining Diabetic Databases: Are Rough Sets a Useful Addition?, Joseph L. Breault, MD, MPH, MS, Department of Health Systems Management, Tulane University, Department of Family Practice, Alton Ochsner Medical Foundation, joebreault@tulanealumni.net.

[9] Parallel Medical Image Analysis for Diabetic Diagnosis, Yueh-Min Huang, E-mail: huang@mail.ncku.edu.tw, Shu-Chen Cheng, ROC, E-mail: kittyc@mail.stut.edu.tw, *Int. J. Computer Applications in Technology, Vol. 22, No. 1, 2005.*

[10] Feature Subset Selection Using a Genetic Algorithm, Jihoon Yang and Vasant Honavar, lowa state University, IEEE Intelligent Systems
Exploration Mining in Diabetic Patients Databases : Findings and Conclusions, Wynne Hsu Mong Li Lee Bing Liu Tok Wang Ling, School of Computing, National University of Singapore, Lower Kent Ridge Road, Singapore 119260, {whsu, leeml, liub, lingtw} @comp.nus.edu.sg.

Biography

S.Sapna received her B.Sc Degree, M.C.A and M.Phil Degree, Bharathiar University. She is currently working as Assistant Professor in the Department of MCA, K.S.R. College of Engineering. She has presented more than 20 papers on various topics including national, international conference and journals. She is a research scholar of Mother Teresa Women's University, Kodaikanal. Her research interest includes Soft computing, Data Mining, Mathematical Computations and Networks. She is a member of life member of ISTE and CSI.

Dr.A.Tamilarasi, currently serving as Prof. & Head, Department of M.C.A, Kongu Engineering College. She has published various papers in the filed of fuzzy logics. She has published various books in mathematical field. She is guiding several research scholars in her area of interest like Data Mining, Soft Computing and Networks.

M.Pravin Kumar received his B.E & M.E Degree, from Anna University. He is currently working as Assistant Professor in the Department of ECE, K.S.R. College of Engineering. He has presented more than 5 papers on various topics including national, international conference and journals. He is a research scholar of Anna University, Coimbatore. His research interest includes Soft Computing and Networks. He is a member of life member of ISTE.