IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 4, Ver. I (July – Aug. 2015), PP 09-16
www.iosrjournals.org

Fuzzy Inference Rule Generation Using Genetic Algorithm Variant

Shruti S. Jamsandekar[1], Ravindra R. Mudholkar [2]
[1]*(Department of Computer Studies, CSIBER(Kolhapur) ,India)*
[2]*(Department of Electronics, Shivaji University(Kolhapur), India)*

Abstract: *In essence of fuzzy inference system (FIS) for classification, Genetic algorithm (GA) which is an optimal searching technique is used for generation rules in the proposed work. This paper develops an FIS with rules generated using GA, the GA is developed with rule importance as fitness criteria. The rule importance of each rule is calculated by its rule support in each rule class. Encoding of rules is using the fuzzy membership set no for antecedent and consequent. Also the stopping criteria is a combination ofgenerations specified and Minimum rules fired count. The Proposed system using GA approach for rule generation giving consistent results with optimal rules.*

Keywords: *fuzzy rules, genetic algorithm, rule_support, rule importance, fitness function.*

I. INTRODUCTION

Fuzzy Inference have been applied to various problem, of which classification problem apart of data mining task [12] is of one which has application in fields, like medicine, education, business etc. classification process can be improved by using fuzzy sets having overlapping class definitions and if-then rules for inference that provide better reasoning in ambiguous. Fuzzy rules are usually constructed by expert based on their domain knowledge but at times this can be tedious and cumbersome.

Many researchers have worked and still trying to do automatic generation of rules using neural network, association rule mining, genetic algorithm (GA) [1][9][6][5]. The reason why GA is preferred is its global-search nature, GAs work with a population of candidate solutions (individuals).Second, in GAs a candidate solution is evaluated as a whole by the fitness function. These characteristics are in contrast with most greedy rule induction algorithms, which work with a single candidate solution at a time based on local information only. [11]In this proposed work we have intended to construct an classification system with fuzzy inference . The approach is towards automatic construction of fuzzy system. The automatic construction of fuzzy inference system(FIS) for classification is developed in 2 phases.

1.Self generated membership function for FIS
2.Generation of Fuzzy Rules.

The automatic construction of fuzzy classification system by self generated membership function already done in preceding research work by us [13].

A data driven approach for automatic generation of fuzzy rules is the focus of the experiment in this paper. Initial population of rules is built in the form of matrix and input membership functions are built with membership function parameters stored in array form. These parameter arrays are generated using clustering algorithm. Both the rules and input, output FIS variables parameters and data are passed as arguments to proposed algorithm to generate rules using genetic algorithm approach.

This paper is organized as follows: Section2 gives the state of the art for mining rules using fuzzy genetic approach. Section 3 is devoted to Rule_Gen, the algorithm we propose for generating fuzzy rules using genetic algorithm. Experimental tests on real-life datasets are described in Section4. We conclude in Section5with a summary of our contribution.

II. RELATED WORK

The authors in [5] proposed a algorithm for association rule mining named Quant-Miner using genetic algorithm, the algorithm dynamically chooses good intervals for each rule with categorical attributes. The approach used by them is evolutionary optimizing support and confidence with fitness function based on gain measure and the algorithm looking for positive gain. Soumadip Ghosh et.al have proposed in [6] an genetic algorithm approach for finding frequent item sets which caters to positive and negative association mining.

In[7] authors have proposed an algorithm to measure the quality of categorical data in association rule mining, they have converted the categorical data to asymmetric binary form by introducing new items as per the distinct attribute pair, apriori association algorithm is applied to find the frequent itemset with minimum support. Quality of data in the frequent itemset obtained is measured in terms of confidence factor, completeness, interestingness, comprehensibility, the rules with optimum values for all measures is searched using genetic algorithm. The author in [8] tried to explore the different methods of using genetic algorithm with K- nearest neighbor algorithm to improve the classification accuracy and minimize the training error. Soraj et.al in [10] proposed a genetic-fuzzy approach for the discovery of fuzzy decision rules from datasets containing categorical as well as continuous attributes. They have fuzzified the continuous attributes using the triangular fuzzy number and also designed a matching process while computing the fitness of an individual in Genetic algorithm population.

III. METHODOLOGY

The proposed algorithm for generation of Fuzzy rules comprises of four procedures .

1. The genetic algorithm for rule generation: Genetic _ Rule_Gen
2. The fitness calculation function comprising of two procedures: support_Rule_class and rule_ratio_importance
3. Evaluation of FIS with rule population generated:GARule

The algorithm for generation of fuzzy rules by using genetic algorithm is as follows.

Repeatedly for n generations following steps are carried out

1. Built the FIS with rule population and evaluate the FIS rules against the output class to return rule firing strength for given data.
2. Calculate and return the support for each rule and the classification class.
3. Calculate and return the importance of each rule which is the fitness value of each rule.
4. Select two parent chromosomes p1 and p2 from the rule population where p1 is parent chromosome with best fitness value and p2 is any parent chromosome from rule population, such that p1 ≠ p2. Usually according to GA approach 2 best fitness parent chromosomes are chosen but in this experiment we have chosen one best fitness value parent chromosome and other parent chromosome with any fitness value as this lead to more variations in children chromosome than in usual approach.
5. Select two rule chromosomes with least fitness value.
6. Crossover points are generated using 2 points crossover method.
7. 2 Offspring chromosomes are generated at the crossover points of parent chromosomes.
8. Each of the offspring generated is checked if present in current population either partially or fully.
9. If offspring chromosomes are not fully or partially matched with current population rule chromosome then replace the offsprings with selected least fitness value chromosomes to built new rule population.
10. If both offsprings are present fully in current population do not replace the offsprings with the least value chromosomes, simply go through next generation.
11. If any one or both of the offspring chromosome/s is in partial form matching with any of the rule chromosome/s in current population, then evaluate the fitness function of the offspring chromosome and compare it with matched rule chromosome/s.
12. If offspring chromosome is having better fitness value than matched rule chromosome then replace the offspring chromosome/s with respective matched rule chromosome to built new rule population.
13. Pass the new rule population to the next generation

    There has been many research on termination criteria of genetic algorithm, with simplest being the iteration based on specific number of times or time based where after specific time with ever is best fitness value obtain is taken as final [2][3][9], also bhandari et.al [4] proposed variance as stopping criteria for genetic algorithm where they have proposed that if ai be the best fitness function value obtained at the end of ith iteration of an GA. Then, a1 ≤ a2 ≤ a3 ……..≤ an ……≤ F1, as F1 is the global optimal value of the fitness function if

    $$a\overline{n} = \frac{1}{n}\sum_{i=1}^{n} ai \qquad (1)$$

    be the average of the ai's and

    $$a^{-}2n = 1/n \sum_{i=1}^{n} a2i \qquad (2)$$

    be the average of the a2i's up to the nth iteration, then variance of the best fitness values obtained up to the nth iteration, defined by bn, is $bn = \frac{1}{n}\sum_{i=1}^{n}(ai - a\overline{n})\,2$

    $$=1/n\sum_{i=1}^{n}(a2i - a\overline{2n}) \qquad (3)$$

$= \bar{a}2n - \bar{a}n2$

bn can be used as a stopping criterion for a GA. GA is stopped or terminated after N iterations when bN<ϵ , where ϵ(> 0) is a user defined small quantity.

3.1 Genetic _ Rule_Gen Procedure

Step 1: Input the initial rule population as matrix P[] n x m (where n is the no of rules and m is the no of rule parameter in form [a1,a2…an, o, w, c] a is input variables of a rule with i input variables; i =1…n,o is the output variable, w is the rule weight and c is connection and/or.) , membership function generated for fuzzy input variables as matrix input(i)[]1xmf (where i is input no. and mf=1,2,… j) and input data

Step 2: declare the two parents chromosome structures p1,p2 and two offspring chromosome structures off1,off2.

Step 3: initial the generation count to 1

Step 4: Repeat step 4 to step 10 until generation_count<100 and stop_flag ==false

Step 5: calculate the fitness of each rule from rule population

Step 5.1 : call GaRuleprocedure to automatically generate the FIS and evaluate the FIS rules against the output class to return rule firing strength for given data

Step 5.2 : call support_Rule_class procedure to calculate and return the support for each rule and for each classification class

Step 5.3: call rule_ratio_importance procedure to calculate and return the importance of each rule which is the fitness value of each rule.

Step 5.4: count the rules with importance as zero store in count variable.(indicating rules not fired)

Step 5.5 : if count <min_count specified then set stop_flag = true else stop_flag=false

Step 6: Select two parent chromosomes from the population where p1 [aij1,aij2…aijn,oi,wi,ci] is parent chromosome with best fitness value and p2 is any other parent chromosome from rule population P[] , p1 ≠ p2 .

Step 7: Compute the crossover points crp1,crp2 as follows

crpi = floor(1 + (rand(1) * n)) i=1,2 (4)

Step 8: Generate offsprings at the crossover points crp1,crp2

off1[X1,X2,….Xn,w,c]

off2[Y1,Y2,….Yn,w,c]

Step 9: Select the least fitness value chromosomes from the P[]

d1= rule chromosome no with min rule importance

d2= rule chromosome no with second last min rule importance

Step 10: check if offspring off1 and off2 chromosome pattern generated are fully or partially in P[] .

Step 10.1: if (∃ (off1 ^off2) ∈P) then set flag_both =true

Step 10.2: if (∃ T[x1,x2…xn - 1]∈ P==off1[x1,x2…..xn-1]) then call GaRule, support_Rule_class, rule_ratio_importance procedure for off1

Step10.2.1:ifrule_ratio_importance(off1)>rule_ratio_importance(T) then d1 = T;

Step10.3:if(∃ T[x1,x2…xn-1]∈P==off2[x1,x2…..xn-1]) then callGaRule, support_Rule_class ,rule_ratio_importance procedure for off2

Step10.3.1: if rule_ ratio_ importance(off2)>rule_ratio_importance(T) then d2 = T;

Step 11: if(flag_both == true) thenno replacement of off1 and off2 with d1 and d2 in P.else

P[ ] = P[ ] – d1 and P[ ] = P[ ] – d2

P[ ] = P[ ] ∪off1∪off2

Step 12: return P;// new rule population.

3.2 GARuleProcedure

The procedure builds up a FIS automatically based on the input data by constructing the input and output membership function and also the rule generated. Returns FIS evaluation parameters (output, IRR,ORR,ARR)

Following are the steps of the GARule procedure.

Step1: Input the membership function array created using clustering technique for input and output variables, input data arrays and rule population matrix.

Step 2: calculate the input membership function degree of input membership functions using evalmf matlab function and store it in eval_input matrix.

Step 3: create an FISclustclass using newfis Matlab function.

Step 3.1: add input/s and output variables to the FIS using addvarmatlab function.

Step 3.2: add input/s and output membership functions to the FIS using addmf matlab function with input membership array data generated.

Step **3**.3: add rule population to the FIS using addrule matlab function

Step 4 : evaluate the FIS created with the input data using evalfis matlab function which returns.

output: the output matrix of size M-by-L, where M represents the number of input values specified previously, and L is the number of output variables for the FIS.[14].

IRR: the result of evaluating the input values through the membership functions. This matrix is of the sizenumRules-by-N, where numRules is the number of rules, and N is the number of input variables.[14

ORR: the result of evaluating the output values through the membership functions. This matrix is of the sizenumPts-by-numRules*L, where numRules is the number of rules, and L is the number of outputs. The firstnumRules columns of this matrix correspond to the first output, the next numRules columns of this matrix correspond to the second output, and so forth.[14]

ARR: the numPts-by-L matrix of the aggregate values sampled at numPts along the output range for each output. [14]

Step 5: End

3.3 Support_Rule_Class Procedure

Function support_Rule_class(OR, POP, n)

OR : array consisting result of evaluating the output values through the membership functions

POP : rule population

n : number of rules

```
Step 1: declare and initialize array for support_rule(Number of data points supporting individual rule),
    rule_class(class support for each rule),total_support_rule(total inference values by support_rule), V( is a
    temporary vector to store all OR values), support_class(Number of data points supporting rule class).
Step 2 : loop through all rows s of OR matrix
        Step 2.1: if OR[s,t] >0 then V[t] = OR[s,t]; Else V(t)=-1 ∀ i ∈{ 1,2......n}
        Step 2.2: Find the rule no I with maximum OR value in vector V
        Step 2.3: support_rule(I)= support_rule(I)+1;

        Step 2.4: total_support_rule(I)= total_support_rule(I) + V(I);
        Step 2.5: end loop
Step 3: calculate the support_class for each rule by using formula
    support_rule[i] >0→support_class[POP[i,3] ]
     = support_class[POP[i,3] ] + support_rule[i]  ∀i∈{ 1,2......n}          (5)
```

Step 4: calculate the rule_class for each rule by using formula

rule_class(i)= support_class(POP(i,3)) ∀i∈{ 1,2……n} (6)

Step 5 : end

3.4 Rule_Ratio_ImportanceProcedure

Function rule_ratio_importance(SR,RC,TSR,n)

SR: rule support

RC: rule class

TSR: total support for the rule in each class

n: number of rules

Step1: declare ratio and importance array for n rules

Step2: calculate ratio and importance

RC[i]==0 -> ratio[i]= SR[i] ie(if rule class is not defined assign rule support as the ratio for the rule) else

ratio[i] = SR[i]/RC[i] (7)

(ratio will give the number of data points supporting rule with the rule class) and

importance [i] = ratio[i]*TSR[i] (8)

(if ratio is bigger more the importance of the rule ∀ i ∈{ 1,2……n})

Step 3: return ratio and importance

Step 4: end

IV. Experimental Results

The experiment is carried out using 51 student data set of MSc. Course of Shivaji university on fuzzy tool box of MATLAB Rb2009 and genetic algorithm is implemented using MATLAB programming .The GA Population comprises of the fuzzy rule list with 25 candidate rules coded in the form of if-then rules in table 1:

Rule no.	**Input 1**	**Input 2**	**Input 3**	**Input 4**	**Class Output**	**Rule Weight**	**Implication (And/Or)**
1	1	1	1	1	1	1	1
2	1	2	2	1	2	1	1
3	1	3	2	3	3	1	1
4	1	4	3	3	3	1	1
5	1	5	3	5	4	1	1
6	2	1	1	2	1	1	1
7	2	2	2	2	2	1	1
8	2	3	2	2	2	1	1
9	2	4	3	3	3	1	1
10	2	5	4	3	4	1	1
11	3	1	2	3	2	1	1
12	3	2	3	3	3	1	1
13	3	3	3	3	3	1	1
14	3	4	4	4	4	1	1
15	3	4	4	2	3	1	1
16	4	1	2	4	3	1	1
17	4	2	2	3	2	1	1
18	4	3	3	4	2	1	1
19	4	4	4	4	4	1	1
20	4	5	5	5	5	1	1
21	5	1	2	3	3	1	1
22	5	2	2	3	4	1	1
23	5	3	3	3	4	1	1
24	5	4	4	4	5	1	1
25	5	5	5	5	5	1	1

Table 1: Initial Rule List

Parent Chromosome 1:

3	3	3	3	3	1	1

Parent chromosome 2:

4	5	5	5	5	1	1

Cross over points: two point crossover operation at 2 – 4

Offspring chromosome 1 :

4	3	3	3	5	1	1

Offspring chromosome 2:

3	5	5	5	3	1	1

The rules generated using GA is in Table 2 where each rule antecedent, consequent and rule importance is shown.

The proposed system was tested for many epochs and was giving the almost same optimal rules but terminated at after different no of generation. 11 epoch result are shown in Table 3

V. Conclusion

The Proposed GA algorithm with rule importance as fitness criteria and with termination criteria as combination of specified generations and rules firing count is stable and has shown consistent results.It is been observed that out the 25 rules from initial population only 16 rules have been fired with good rule importance in classifying the items. GA terminated at different generations for each epoch but the rules identified with good rule importance ie with higher firing strength are almost same for each epoch. Also fuzzy inference of the 51 data sets using the rules generated a consistent result for each epoch generated rules. The results are tabulated in Table 4. The Figure 1 shows the epoch wise results for 6 data values.

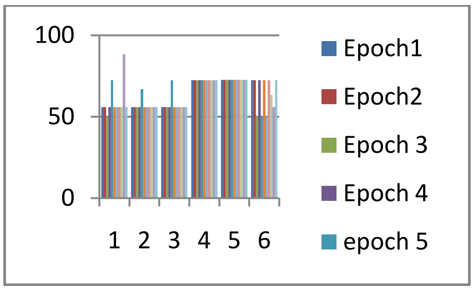

Figure 1: Epoch Wise Results Of Data

Rule No.	Input 1	Input 2	Input 3	Input 4	Class Output	Rule Weight	Implication (And/Or)	Rule Importance
1	1	1	1	1	1	1	1	0.059
2	1	2	2	1	2	1	1	0.077
3	1	3	2	3	3	1	1	0.009
4	1	4	3	3	3	1	1	0.011
5	1	5	3	5	4	1	1	0.022
6	2	1	1	2	1	1	1	0.073
7	2	2	2	2	2	1	1	0.213
8	2	3	2	2	2	1	1	0.358
9	2	4	3	3	3	1	1	2.570
10	2	5	4	3	4	1	1	1.184
11	3	1	2	3	2	1	1	0
12	3	2	3	3	3	1	1	0
13	3	3	3	3	3	1	1	0

14	3	4	4	4	4	1	1	0.018
15	3	4	4	2	3	1	1	0.009
16	4	1	2	4	3	1	1	0
17	4	2	2	3	2	1	1	0
18	4	3	3	4	2	1	1	0.008
19	4	4	4	4	4	1	1	0.340
20	4	5	5	5	5	1	1	0
21	5	1	2	3	3	1	1	0
22	5	2	2	3	4	1	1	0
23	5	3	3	3	4	1	1	0
24	5	4	4	4	5	1	1	0.226
25	5	5	5	5	5	1	1	0

Table 2: Rules generated using GA with its rule importance

Rules	**Epochs**										
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Rule 1	1.09	0.02	3.04	1.18	0.27	1.09	0.27	0.02	0.01	0.22	0.06
Rule 2	0.07	0.05	0.19	0.23	0.00	0.07	0.00	0.01	0.01	0.00	0.08
Rule 3	0.06	0.07	0.06	0.02	0.04	0.03	0.01	0.01	0.01	0.01	0.01
Rule 4	0.01	0.01	0.00	0.07	0.01	0.07	0.00	0.01	0.03	0.51	0.01
Rule 5	0.01	0.01	0.00	0.05	0.01	0.01	0.00	0.06	0.09	0.13	0.02
Rule 6	0.01	1.18	0.00	0.01	0.00	0.01	0.00	2.57	1.09	1.94	0.07
Rule 7	0.21	0.21	0.18	0.21	0.04	0.21	0.21	0.21	0.21	0.16	0.21
Rule 8	0.36	0.36	0.31	0.36	0.48	0.56	0.56	0.36	0.36	0.27	0.36
Rule 9	0.02	0.23	0.00	0.07	0.00	2.91	0.00	0.08	2.91	0.19	2.57
Rule 10	2.57	0.07	0.00	0.01	0.00	0.00	0.00	1.09	0.07	0.00	1.18
Rule 11	0.08	2.04	0.00	2.04	0.00	0.00	0.00	0.07	0.00	0.02	0.00
Rule 12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rule 13	0.00	0.00	0.00	0.00	2.56	0.00	2.72	0.00	0.00	0.00	0.00
Rule 14	0.02	0.02	0.12	0.02	0.00	0.02	0.12	0.02	0.02	0.12	0.02
Rule 15	0.01	0.01	0.01	0.01	0.15	0.01	0.01	0.01	0.01	0.01	0.01
Rule 16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rule 17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rule 18	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Rule 19	0.31	0.34	0.62	0.34	0.20	0.31	0.62	0.31	0.31	0.62	0.34
Rule 20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rule 21	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rule 22	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rule 23	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rule 24	0.23	0.23	0.45	0.23	0.45	0.23	0.45	0.23	0.23	0.23	0.23
Rule 25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Generations for termination	34	64	100	46	100	100	100	72	58	65	94

Table 3: Generations wise rules generated with its rule importance

Sr. No	**Epochs**										
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
2	72.54	72.54	50.00	72.54	50.00	72.54	50.00	72.54	63.26	55.80	72.54
3	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
4	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
5	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
6	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00

7	55.93	55.93	55.93	55.93	66.67	55.93	55.93	55.93	55.93	55.93	55.93
8	72.48	72.48	50.00	72.48	50.00	72.48	50.00	72.48	63.42	55.93	72.48
9	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
10	55.89	55.89	50.00	55.89	72.54	55.89	55.85	55.89	55.89	88.48	55.89
11	55.93	55.93	55.93	55.93	66.91	55.93	55.93	55.93	55.93	55.93	55.93
12	55.94	55.94	55.94	55.94	72.47	55.94	55.94	55.94	55.94	55.94	55.94
13	72.47	72.47	72.47	72.47	72.47	72.47	72.47	72.47	72.47	72.47	72.47
14	72.59	72.59	72.59	72.59	72.59	72.59	72.59	72.59	72.59	72.59	72.59
15	72.48	72.48	50.00	72.48	50.00	72.48	50.00	72.48	63.42	55.93	72.48
16	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
17	50.00	50.00	50.00	50.00	50.00	50.00	55.83	50.00	50.00	50.00	50.00
18	72.50	72.50	55.90	72.50	50.00	72.50	50.00	72.50	63.38	55.89	72.50
19	36.90	36.90	36.90	36.90	47.07	36.90	36.90	36.90	36.90	36.90	36.90
20	36.50	36.50	36.50	36.50	36.50	36.50	25.78	36.50	36.50	36.50	36.50
21	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
22	72.54	55.87	50.00	55.87	36.71	72.54	55.87	72.54	72.54	55.87	50.00
23	65.96	65.96	65.96	65.96	65.96	65.96	65.96	65.96	65.96	65.96	65.96
24	55.90	55.90	55.90	55.90	58.81	55.90	55.90	55.90	55.90	55.90	55.90
25	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
26	37.82	37.82	37.82	37.82	37.82	37.82	37.82	37.82	37.82	37.82	37.82
27	72.59	72.59	72.59	72.59	72.59	72.59	72.59	72.59	72.59	72.59	72.59
28	50.00	55.95	50.00	55.95	37.93	50.00	55.95	50.00	50.00	55.95	50.00
29	50.00	50.00	37.80	55.95	50.00	50.00	50.00	50.00	50.00	50.00	50.00
30	55.93	55.93	55.93	55.93	72.48	55.93	55.93	55.93	55.93	55.93	55.93

Table 4: Results Of FIS System Of 11 Epochs for The Dataset

References

[1]. Johannes A. Roubos a, MagneSetnes b, Janos Abonyi c, “Learning fuzzy classification rules from labeled data”, Information Sciences 150 , Elsevier Publication (2003) pg:77–93

[2]. HisaoIshibuchi, Ken Nozaki, Naohisa Yamamoto, Hideo Tanaka, “ Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms”, Fuzzy Sets and Systems 65 Elsevier Publication (1994) pg: 237-253

[3]. UjjwalMaulik, SanghamitraBandyopadhyay,“Genetic algorithm-based clustering technique”, Pattern Recognition 33, A Journal of Pattern Recognition Society, Published by Elsevier Science (2000) 1455-1465

[4]. DinabandhuBhandari, C. A. Murthy, Sankar K. Pal, “Variance as a Stopping Criterion for Genetic Algorithms with Elitist Model”,FundamentaInformaticae 120 (2012) 145–164

[5]. AnsafSalleb-Aouissi, ChristelVrain, Cyril Nortet, “QuantMiner: A Genetic Algorithm for Mining Quantitative Association Rules”, Proceeding of International Joint Conference on Artificial Intelligence,pg 1035-1040, 2007.

[6]. SoumadipGhosh, SushantaBiswas, DebasreeSarkar, ParthaPratimSarkar, “Mining Frequent Itemsets Using Genetic Algorithm”,International Journal of Artificial Intelligence & Applications (IJAIA), Vol.1, No.4,pg. 133-144, October 2010.

[7]. J.MalarVizhiAnd Dr. T.Bhuvaneswari,“Data Quality Measurement On Categorical Data Using Genetic Algorithm”, International Journal of Data Mining &Knowledge Management Process (IJDKP) Vol.2, No.1, January 2012.

[8]. Robert Marmelstien, “Application of Genetic Algorithms toData Mining”, Eighth Midwest Artificial Intelligence And Cognitive Science Conference , Maics-97 Proceedings, AAAI (www.aaai.Org),pg 53-57,1997.

[9]. C.H. Lo , P.T. Chan, Y.K. Wong, A.B. Rad, K.L. Cheung, “Fuzzy-genetic algorithm for automatic fault detection in HVAC systems”, Applied Soft Computing 7, Elsevier Publication, (2007) 554–560

[10]. Dr. Saroj, NishantPrabhat, “A Genetic-Fuzzy Algorithm to Discover Fuzzy Classification Rules for Mixed Attributes Datasets” International Journal of Computer Applications (0975 – 8887) Volume 34– No.5, November 2011

[11]. Goldberg D.E.: “Genetic Algorithms in Search, Optimization and Machine Learning”, Addison-Wesley Publishing Company, Inc. MA, New York, 1989.

[12]. Han J. and Kamber M.: “Data Mining: Concepts and Techniques,” Morgan Kaufmann Publishers, San Francisco, 2000.

[13]. Mrs. S.S Jamsandekar and Dr. R.R. Mudholkar, “Fuzzy Classification System by self generated membership function using clustering technique”, BIJIT - BVICAM’s International Journal of Information Technology Special issue on Fuzzy logic,Vol.6 No.1 Issue 11,(2014)

[14]. Matlabevalfisfuntionavailable: http://in.mathworks.com/help/fuzzy/evalfis.html(accessed: 18 March 2015)