SOFT COMPUTING TECHNIQUES: THEORY AND APPLICATION FOR PATTERN CLASSIFICATION

SIVAKUMAR GANESAN

(B.Eng.)

A THESIS SUBMITTED
FOR THE DEGREE OF MASTER OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE
2004

Acknowledgements

It is a great pleasure to express my heartfelt gratitude and appreciation for my advisor and friend, Professor Prahlad Vadakkepat, for his advice, support and encouragement. His continuous guidance helped me to focus on my research and overcome various stumbling blocks which came in my path. I am indebted to him for his patience and his precious time in amending the drafts of this thesis. I thank Dr. Prahlad for everything I learned from him.

I also would like to thank Dr. Abdullah Al Mamun for helping me with the draft of the proposal on Genetic Algorithm Based Heuristic Fuzzy Pattern Classifier.

I thank my wife, parents and friends for their love and unfailing support and for bearing with the rigors of my research work, without which I cannot have furthered my studies.

I am also grateful to all the individuals in the Computer Center as well as the

Department of Electrical and Computer Engineering, National University of Singapore, which provides various hardware, software and research facilities to conduct my research work.

Finally, I wish to acknowledge the National University of Singapore (NUS) for providing me this wonderful opportunity in pursuing research.

List of publications

- Sivakumar Ganesan, Prahlad Vadakkepat, Abdullah Al Mamun and Cai Ji, "Handling Of Continuous Attributes And Application Of Statistical Methods In Rough Classifiers", *Proceedings of the 1st International Conference on Fuzzy Systems and Knowledge Discovery (FSKD02)*, Singapore, Nov 2002, Vol. 2, pp. 559-563.

- Sivakumar Ganesan and Prahlad Vadakkepat, "Boosting based Fuzzy-Rough Set Pattern Classifier". Submitted to *IEEE Transactions on Fuzzy Systems* and awaiting reply.

- Sivakumar Ganesan, Prahlad Vadakkepat and Abdullah Al Mamun, "Genetic Algorithm Based Heuristic Fuzzy Pattern Classifier". Submitted to *International Journal on Artificial Intelligence Tools* and awaiting reply.

Contents

Summary

Soft computing is an emerging approach to computing which parallels the remarkable ability of the human mind to reason and learn in an environment of uncertainty and imprecision. (Lotfi A. Zadeh, 1992 [1])

There are a variety of real world problems such as Pattern Recognition, Image processing, Voice recognition, Data mining etc. for which, normal computing techniques are either inadequate or very tedious to apply. Soft computing techniques have been developed to fill this gap and have gained increasing popularity in the recent years. Leading examples of popular soft computing techniques are fuzzy systems, rough sets, neural networks, genetic algorithms, simulated annealing etc. In addition to solving such real world problems, soft computing techniques are also gaining acceptance in areas such as Control Systems, IP routing systems etc. where the regular computing techniques were considered to be the de-facto standard.

In this thesis, basic ideas regarding Fuzzy systems, Rough Sets and Genetic algorithms are introduced, followed by the research work available in the literature. Then, the author's contribution on Pattern Classification based upon Rough set techniques, fuzzy systems and genetic algorithms are described. First, a classifier based on a combination of rough sets and NNR technique is proposed, which performs better than NNR technique alone. This is followed by a fuzzy classifier based on Pittsburgh approach genetic algorithm, in combination with the grade of certainty (CF). The classifier performs better than a Pittsburgh approach fuzzy classifier without Grade of certainty. It is also compared with a Michigan approach fuzzy classifier which neither tunes the membership functions nor minimizes the number of fuzzy rules. Finally, a new classifier based on fuzzy lower and upper approximations and a boosting technique is proposed, which makes use of the Plausibility factor (PF). Further, the Plausibility factor (PF) is compared with the grade of certainty to prove its efficacy. The performance of the classifiers are illustrated on well known test problems. In the final part, possible future directions for further research is discussed.

List of Tables

List of Figures

Chapter 1

Introduction

Soft computing, an innovative approach to construct computationally intelligent systems, has recently gained popularity and wide spread use. It is being realized that complex real world problems require intelligent systems that combine knowledge, techniques and methodologies from various sources. These intelligent systems are supposed to possess human-like expertise within a specific domain and the ability to adapt and learn in changing environments. To achieve this complex goal, a single computing paradigm or solution is not sufficient.

Soft computing is a wide ranging term encompassing such varied techniques as fuzzy systems, rough sets, neural networks, genetic algorithms, simulated annealing, DNA computing, Quantum computing, Membrane computing etc. While some of these techniques are still in the nascent stage, the rest of them have found wide spread use in the area of Pattern recognition, Classification, Image processing, Voice recognition, Data mining etc. Each of these methodologies have their own strength. The seamless integration of these methodologies to create intelligent systems, forms the core of soft computing. This thesis concentrates on the area of rough set techniques, fuzzy systems and genetic algorithms and their application to pattern recognition and classification.

The problem of finding models that describe or classify measurement data is encountered in many situations. This task falls into the extensive category of empirical modelling, which, broadly speaking, can be said to be the science of constructing models that describe or classify measurements. Such models may take on a wide variety of forms according to the model construction scheme used. The choice of modelling scheme bears both to the nature of the pattern recognition problem, as well as to the purpose of the modelling task. The purpose of developing such models may be twofold: In some instances, the goal may be to gain insight into the problem at hand by analyzing the constructed model, i.e., the structure of the model is itself of interest. In other applications, the transparency and explainability features of the model is of secondary importance, and the main objective is to construct a classifier of some sort that classifies arbitrary objects well. A knowledge-based approach for constructing empirical models based on labelled data, is to inductively infer a set of general rules that produce the desired class. This is an instance of an activity that is collectively referred to as knowledge discovery in databases (KDD), or, sometimes, data mining. The term machine learning is often used as a common label for many of the techniques involved in learning from examples. Soft computing techniques have been extensively studied for constructing such empirical models for pattern recognition and classification problems.

The aim of this work is to analyze and develop advanced and reliable techniques for pattern classification, based on techniques such as rough sets, fuzzy sets and evolutionary computation.

In Chapter 2, a general overview of rough sets, fuzzy inference systems and genetic algorithms are provided, along with a survey of the application of these techniques for pattern recognition and classification in literature. In Chapter 3, a rough set classifier and its integration with statistical methods is discussed. A genetic algorithm based heuristic fuzzy classifier is discussed in Chapter 4. Chapter 5 analyzes a new classification technique based on the integration of rough sets and fuzzy sets, which is combined with a boosting enhanced genetic algorithm. Finally, Chapter 6 provides possible directions for future research.

Chapter 2

General Overview of Rough sets, Fuzzy systems and Genetic Algorithms

This chapter provides a brief introduction to the three techniques: Rough sets, Fuzzy systems and Genetic Algorithms. The strength of each of these techniques is highlighted and their potential area of application is discussed.

2.1 Rough Set

The notion of rough set was proposed by Pawlak [3, 4] in early '80s. Rough set theory was developed for classificatory analysis of data tables to discover relationships in data. Though it appears similar to statistics in that it deals with data relationships, it is entirely different: instead of employing probability to express vagueness of data, it uses set theory for this purpose.

The idea of rough set consists of the approximation of a set by a pair of sets, called the lower and the upper approximation of this set [3]. The rough set concept overlaps with many other mathematical ideas developed to deal with imprecision and vagueness, in particular with Fuzzy Set theory [5] and theory of evidence [6].

Interesting comparison of rough sets and fuzzy sets has been published by Dubois and Prade [7], [8]. The authors propose that fuzzy and rough sets aim at different purposes in modelling uncertainty, namely vagueness for fuzzy sets and coarseness for rough sets, in order to get a more accurate account of imperfect information. Further, lower and upper approximations of a fuzzy set are defined when the universe of discourse is coarsened by means of an equivalence relation. The equivalence relation is turned into a fuzzy similarity relation for a more expressive modelling of coarseness and its properties are studied.

Discussion of the relationship between rough sets and the evidence theory can be found in Grzymala-Busse [9] and Skowron [10], which are however, outside the scope of this thesis.

2.1.1 Knowledge Representation System

In Rough set theory, knowledge is considered as the ability to classify objects and indiscernibility relationship forms the basis of this classification. In this context, data is represented as a Knowledge Representation System S which is a single two-dimensional table. Formally,

$$S = (U, A) \tag{2.1}$$

where U is a non-empty finite set of objects or Universe of Discourse and A is a non-empty finite set of attributes.

Each attribute $a \in A$ can be viewed as a function that maps elements of U into a set V_a. The set V_a is called the value set of attribute a.

$$a : U \rightarrow V_a \tag{2.2}$$

Discernibility Matrix:

Skowron [11] represents knowledge in the form of a discernibility matrix which enables easy computation of core, reducts and other rough set concepts. A discernibility matrix of a Knowledge Representation System $S = (U, A)$ with $U = x_1, x_2, ..., x_n$ denoted as $M(S)$, is a nxn matrix defined as:

$$(c_{ij}) = a \in A : a(x_i) \neq a(x_j) \text{ for } i, j = 1, 2, ..., n. \tag{2.3}$$

Thus c_{ij} is the set of all attributes which discern objects x_i and x_j

In rough set terminology, a reduct of knowledge is defined as its essential part, which suffices to define all basic concepts occurring in the considered knowledge, whereas the core can be interpreted as the set of the most characteristic part of the knowledge, which cannot be eliminated when reducing the knowledge. In other words, core is the most important part of a reduct and is included in every reduct.

Thus, the core can now be defined as the set of all single element entries of the discernibility matrix,

$$CORE(A) = a \in A : c_{ij} = (a) \text{ for some } i, j. \tag{2.4}$$

It can be easily seen that $B \subseteq A$ is the reduct of A, if B is minimal subset of A such that,

$$B \cap c \neq \emptyset \text{ for any nonempty entry } c\ (c \neq \emptyset) \text{ in } M(S). \tag{2.5}$$

Conceptually, this means that a reduct is the minimal subset of attributes that discerns all objects discernible by the whole set of attributes.

With every subset of attributes $B \subseteq A$, rough set theory associates a binary relation $IND(B)$ called an *indiscernibility relation* and defined as:

$$IND(B) = (x, y) \in U^2 : \text{ for every } a \in B, a(x) = a(y) \tag{2.6}$$

In other words, $IND(B)$ is an equivalence relation and

$$IND(B) = \bigcap_{a \in B} IND(a) \tag{2.7}$$

A discernibility matrix M_A defines a binary relation $R_A \subseteq U^2$. The relation R_A is called an indiscernibility relation with respect to A, and expresses which pairs of objects cannot be discerned between, with respect to A. In other words, $IND(B)$ is an equivalence relation and

$$xR_Ay \Leftrightarrow M_A(x, y) = \emptyset \tag{2.8}$$

The indiscernibility set of an object $x \in U$ is denoted $R_A(x)$ and consists of those objects that stand in relation to object x by R_A.

$$R_A(x) = y \in U|xR_Ay \tag{2.9}$$

If R_A is an equivalence relation, then the indiscernibility sets are called equivalence classes. Equivalence relations induce a partition of the universe, meaning that all equivalence classes are disjoint and their union equals the full universe U. Vice versa, a partition also induces an equivalence relation.

Lower and Upper Approximations:

The basic idea behind rough sets is to construct approximations of sets using the binary relation R_A. The indiscernibility sets $R_A(x)$ form basic building blocks from which subsets $X \subseteq U$ can be assembled. If X cannot be defined in a crisp manner using attributes A, then X is considered as a Rough Set and is defined through *lower and upper approximations* $\underline{A}X$ and $\overline{A}X$, defined below:

$$\begin{aligned} \underline{A}X &= x \in U|R_A(x) \subseteq X & (2.10) \\ \overline{A}X &= x \in U|R_A(x) \cap X \neq \emptyset & (2.11) \end{aligned}$$

The lower approximation consists of those objects that certainly belong to X whereas the upper approximation consists of the objects that possibly belong to X. In fact, the upper approximation includes the lower approximation. The boundary region is defined as the difference between the upper and the lower approximation, and consists of the objects that cannot be decisively assigned as being either a member or non-member of X. The outside region is defined as the complement of the upper approximation, and consists of the objects that are definite non-members. A rough set is any subset $X \subseteq U$ defined through its lower and upper approximations.

Rough Membership Functions:

The rough membership function [12] is a function $\mu_X^2 : U \rightarrow [0, 1]$ that, when applied to object x, quantifies the degree of relative overlap between the set X and the indiscernibility set to which x belongs. The rough membership function can be interpreted as a frequency-based estimate of $\Pr(x \in X|x, A)$, the conditional probability that object x belongs to set X, given knowledge of the information signature of x with respect to attributes A.

$$\mu_X^2(x) = \frac{| R_A(x) \cap X |}{| R_A(x) |} \tag{2.12}$$

2.1.2 Decision Systems

It often happens that each entry or object in an information system has some kind of label associated with it. For instance, in a medical database, each object may represent a patient that may have a known disease status or treatment outcome. It is typically desirable to incorporate this label into the rough set analysis. An important subclass of information systems are therefore decision systems, also called decision tables.

A decision system is any information system S of the form below, where $d \notin A$ is a distinguished attribute called the *decision attribute.* The elements of A are called *condition attributes.*

$$S = (U, A \cup d) \tag{2.13}$$

Since a decision system is a special kind of Knowledge Representation System, the mathematical machinery developed in Section 2.1.1 is applicable also to decision systems. The decision systems can be utilized to arrive at reducts, which can serve as rules, in a classifier.

2.1.3 Review of rough set techniques in the literature

Rough set theory is suitable for problems that can be formulated as classification tasks, and has gained significant scientific interest as a framework for data mining and KDD [13, 14]. One of the main processes involved in solving classification tasks is the induction of decision rules.

Problems of inducing decision rules have been extensively investigated in many fields, particularly in the machine learning domain [15, 16]. Rough set theory (RST) can also be applied to different stages of rule development and data processing. However, one aspect that distinguishes RST from typical machine learning systems is that, the RST does not correct or aggregate the inconsistency in the input data [17]. The lower and upper approximation are applied to describe the inconsistency and consequently, certain and approximate rules are induced.

Procedures of derivation of decision rules from decision tables were presented by Grzymala-Busse [18], Skowron [19], Slowinski and Stefanowski [20], Stefanowski and Vanderpooten [21] and Ziarko [22]. More advanced rule induction methods have been studied in Bazan (1998) for comparing the dynamic and non-dynamic methods of induction rules from decision tables. Grzymala-Busse and Zou [23] and Stefanowski [24] carried out work in this area with the focus on the induction rules from inconsistent decision tables. Lin [25], Lin and Yao [26] studied the rule induction from very large databases combined with database technologies.

Learning from Examples using Rough Set (LERS) is a rule induction system developed by Grzymala-Busse [18, 27]. There are two different approaches for rule induction in this system, which are computing sufficient rule set using machine learning approach and computing all rules by a knowledge acquisition approach. In both approaches, the user has an additional choice between local and global algorithms. Among these options, Learning from Examples Module, Version 2 (LEM2), which is a local algorithm using the machine learning approach, is most widely used in practice. Stefanowski [28] proposed a modified LEM2 algorithm to handle directly, continuous attributes and discretize them inside the learning algorithm while creating elementary conditions. This algorithm extracts better sets of decision rules than LEM2.

Bazan et al. [29] engaged in market data analysis with the aid of the Rough Set Expert System (RSES) (Bazan and Szczuka [30]). The rule induction system of this software is based on Boolean Reasoning and dynamic reducts [31]. Besides all the basic operations of the RST, this software also provides the discretization algorithms and template generation algorithms.

The Variable Precision Rough Set Model (VPRSM) was proposed by Ziarko [22] as a derivative of the basic RST. This model broadened the deterministic data dependencies, which is the foundation of basic RST, to non-deterministic relationships. Ziarko et al. [22, 32] proposed a VPRSM model which has been developed into a commercial software- DataLogic - was applied to extract trading rules.

Rough classifier, developed by Lenarcik and Piasta [33] and Rough Data Models introduced by Kowalcrzyk [34] are two approaches that avoid the use of reducts.

Both approaches are focussed on finding a relatively simple partition of the attribute space and then drawing some conclusions from the structure of this partition. Two systems that are representative of these approaches are ProbRough (Lenarcik and Piasta, [35]) and TRANCE (Kowalcrzyk, [34]).

ROSETTA system [13] is a comprehensive set of software components for discernibility-based data analysis. It is a generic rough set based system capable of handling various rough set algorithms along a user friendly GUI. The source code for this system is available in the public domain.

Exhaustive lists of known rough set based software systems are compiled by Polkowski and Skowron [36] and by Komorowski et al. [37].

2.2 Fuzzy Inference Systems

Since their introduction in the late sixties, fuzzy sets have been adopted to map real numbers to symbolic labels. Elements of a universe of discourse belong to a fuzzy set for a certain event, according to the so called membership function that defines it. The relationship between the fuzzy sets for a given universe of discourse, is provided by a fuzzy if-then rule. These if-then rule statements are used to formulate the conditional statements that comprise fuzzy logic. A single fuzzy if-then rule assumes the form

if x is A then y is B

where A and B are linguistic values defined by fuzzy sets on the ranges (universes of discourse) X and Y, respectively. The if-part of the rule x is A is called the antecedent or premise, while the then-part of the rule y is B is called the consequent or conclusion. An example of such a rule might be

If pressure is high, then volume is small

Note that high is represented as a number between 0 and 1, and so the antecedent is an interpretation that returns a single number between 0 and 1. On the other hand, small is represented as a fuzzy set, and so the consequent is an assignment that assigns the entire fuzzy set B to the output variable y.

In general, the input to an if-then rule is the current value for the input variable (in this case, pressure) and the output is an entire fuzzy set (in this case, small). This set is usually *defuzzified*, assigning one value to the output. Interpreting an if-then rule involves distinct parts: first evaluating the antecedent (which involves *fuzzifying* the input and applying any necessary *fuzzy operators*) and second applying that result to the consequent (known as *implication*). This entire process of formulating the mapping from a given input to an output using fuzzy logic is termed as *Fuzzy Inference*.

Fuzzy inference systems have been successfully applied in fields such as automatic control, data classification, decision analysis, expert systems, and computer vision. Because of its multidisciplinary nature, fuzzy inference systems are associated with a number of names, such as fuzzy-rule-based systems, fuzzy expert systems, fuzzy modeling, fuzzy associative memory, fuzzy logic controllers, and simply fuzzy systems.

In the case of Pattern classification problems, this fuzzy inference model is slightly modified. For pattern classification, the output is a class instead of a fuzzy set. It is of the form,

If near-vision is good and far-vision is poor, patient is myopic.

In this case, the *defuzzification* process is ignored and the output class is decided based on the *fuzzified* inputs and the *fuzzy operators* in the fuzzy inference engine. Figure 2.1 shows the structure of a fuzzy rule based system, used in pattern classification. The Knowledge base of the fuzzy rule based system comprises of three components: definition of the scaling factors, definition of, the membership functions of the fuzzy sets utilized in the rule base and a rule base constituted by a collection of fuzzy rules.

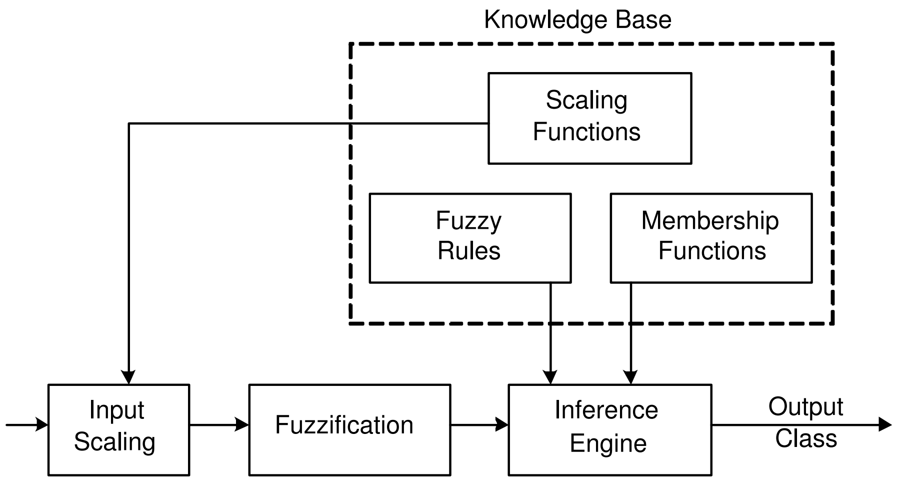


Figure 2.1: Fuzzy Rule Based System for Pattern Classification

2.2.1 Review of fuzzy techniques in the literature

Some of the approaches in fuzzy inferencing and rule generation are outlined here. In the fuzzy classification rule described by Ishibuchi et al. [38], the partitioning is uniform, i.e., the regions continue to be split until a sufficiently high certainty of the rule, generated by each region, is achieved. Ishibuchi et al. extended this work later [39] by using an idea of sequential partitioning of the feature space into fuzzy subspaces until a predetermined stopping criterion is satisfied and studied its application for solving various pattern classification problems.

Wang and Mendel [40] developed a slightly different method for creating a fuzzy rule base, made up of a combination of rules generated from numerical examples and linguistic rules supplied by human experts. The input and output domain spaces are divided into a number of linguistic subspaces. Human intervention is sought to assign degrees to the rules and conflicts are resolved by selecting those rules yielding the maximum of a computed measure corresponding to each linguistic subspace.

Rovatti and Guerrieri [41] attempted to identify the correct rule structure of a fuzzy system when the target input-output behavior is sampled at random points. The assumption that a rule can either be included or excluded from the rule set is relaxed, and degrees of membership are exploited to achieve good approximation results. Defuzzification methodologies are then used to extract well-behaving crisp rule sets. Symbolic minimization is carried out to obtain a compact structure that captures the high-level characteristics of the target behavior. For other details, one may refer to standard literature [42, 43, 44].

While the fuzzy inference system can classify inputs based on existing rule base with easily comprehensible rules, it doesn't have the ability to learn rules from samples. The rules have to be either provided by an expert, or it has be to generated by a separate learning system. Considerable research has been conducted to incorporate learning capabilities to fuzzy systems. Two of the most successful approaches have been, the hybridization attempts made in the framework of soft computing, where different techniques such as neural and evolutionary, provide fuzzy systems with learning capabilities. Neuro-fuzzy systems are one of the most visible directions of that effort [45, 46, 47]. A different approach to hybridization lead to genetic fuzzy systems [48, 49, 50, 51, 52].

2.3 Genetic Algorithms

Genetic Algorithms (GA) are general purpose stochastic, global search algorithms that use principles inspired by natural genetics to evolve solutions for problems. They have been proven to provide robust search capabilities in complex spaces [53, 54]. GAs have had a great measure of success in search and optimization problems. The main reason for their success is their ability to adapt information accumulated about an initially unknown search space, in order to bias subsequent searches into useful subspaces. GAs differ substantially from more traditional search and optimization methods. The five most significant differences are:

1. GAs search a population of points in parallel, not a single point.
2. GAs do not require derivative information or other auxiliary knowledge; only the objective function and corresponding fitness levels influence the directions of search.
3. GAs use probabilistic transition rules, not deterministic ones.
4. GAs work on an encoding of the parameter set rather than the parameter set itself (except in where real-valued individuals are used).
5. GAs provides a number of potential solutions to a given problem and the choice of final solution is left to the user. This makes GA a very powerful tool, for problems such as scheduling, to identify alternative solutions simultaneously.

A GA begins with a randomly generated population of chromosomes and advances towards better chromosomes by applying genetic operators. The population evolves into a better one, thereby narrowing the search space, in a form of natural selection. There are many possible variations to the basic GA, which is outlined as follows:

1. Create an initial population;
2. Evaluate fitness of each chromosome in population;
3. Based on fitness, select chromosomes for reproduction;
4. Apply the genetic operations, *crossover* and *mutation* on selected chromosomes, to form new chromosomes;
5. Replace a part of the current population with newly generated chromosomes;
6. Terminate GA if stopping condition is satisfied, else return to step 2.

Though GAs are search algorithms and not learning systems, they have been used to augment fuzzy systems with learning capabilities. Three of the major approaches which have been successfully applied to fuzzy systems are, the Michigan approach, the Pittsburgh approach and the Iterative Rule Learning approach.

2.3.1 Michigan Approach

In this approach, each chromosome is an individual rule and the entire population represents the rule base. Figure 2.2 illustrates the organization of a fuzzy system based on Michigan approach. This approach is very efficient and is not as computationally intensive as the Pittsburgh approach. However, it does not directly optimize the fuzzy rule base. Even if the selected fuzzy rules are good, this approach may not provide a good fuzzy classification system. Hence, to ensure the evolution of a cohesive and accurate fuzzy classifier, a Credit Assignment System for the individual fuzzy rules is needed. This approach is applied in [55, 56, 57, 58] to generate fuzzy if-then rules, mainly in the domain of control and function approximation problems. Ishibuchi *et al.* [38, 59, 60, 61] proposed Michigan based methods to design fuzzy systems in the domain of pattern classification.

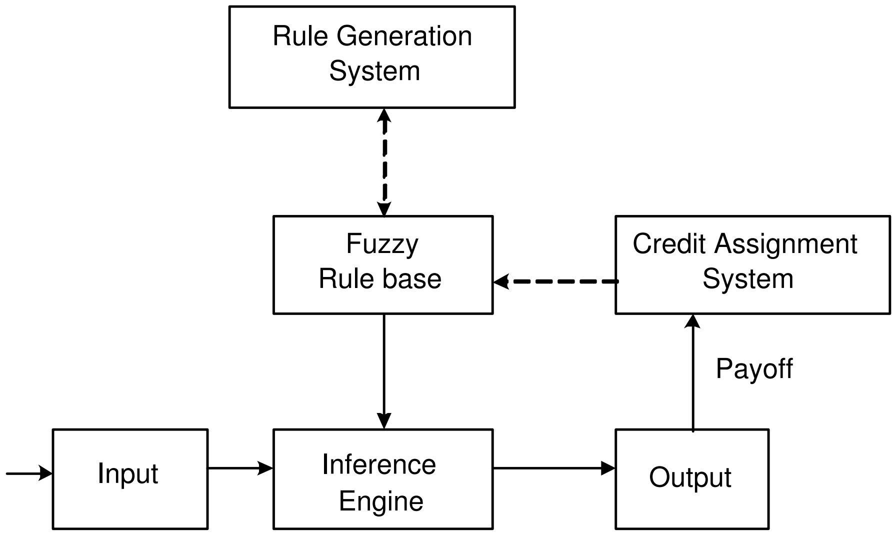


Figure 2.2: Genetic Fuzzy System based on Michigan Approach

2.3.2 Pittsburgh Approach

In this approach, each chromosome encodes a whole fuzzy rule base, including the rules, definitions of the membership functions and optionally, any other parameter of the fuzzy system. This approach is computationally intensive, time consuming and is not as good as Michigan approach in selecting individual fuzzy rules. However, it evolves the entire fuzzy rule base, which results is very good fuzzy classifiers. Figure 2.3 provides the organization of a fuzzy system based on Pittsburgh approach. This approach has been applied by Hwang *et al.* [62], Thrift *et al.* [63], Karr *et al.* [64], Hamaifar *et al.* [65] and Takagi *el al.* [66], Shi *et al.* [2].

2.3.3 Iterative Rule Learning Approach

An interesting discussion about the problems generated by the use of these approaches can be seen in [67]. In the recent literature, a new learning model based on GAs called Iterative Rule Learning approach is gaining popularity. In this approach, the GA provides a partial solution to the problem of learning and attempts to reduce the search space for the possible solutions. This model has been used in papers such as [68, 69, 70, 49, 71].

In order to obtain a set of rules which will be a true solution to the problem, the GA is placed in an iterative scheme to the following [67]:

1. Use a GA to obtain a rule for the system
2. Incorporate the rule into the final set of rules
3. Penalize this rule by reducing its weight in the decision system
4. If the set of obtained rules is sufficient to represent the examples in the training set, then it is returned as the solution set.

This approach obtains a rule in each step and the iteration process obtains the complete set of rules.

2.3.4 Review of genetic fuzzy techniques in the literature

Much research has been conducted for integrating fuzzy systems with genetic algorithms. A fuzzy model, containing a large number of IF-THEN rules, is liable to encounter the risk of overfitting and, hence, poor generalization. The strong searching capacity of GAs has been utilized in fuzzy-genetic hybridization to circumvent this problem by [72] a) determining membership functions with a fixed number of fuzzy rules [64, 73]; b) finding fuzzy rules with known membership functions [63]; and c) finding both membership functions and fuzzy rules simultaneously [72, 65, 59].

Ishibuchi et al. [59] selected a small number of significant fuzzy IF-THEN rules to construct a compact and efficient fuzzy classification system. GAs are used to solve this combinatorial optimization problem, with an objective function for simultaneously maximizing the number of correctly classified patterns and minimizing the number of fuzzy rules.

Wang and Yen [72] have designed a hybrid algorithm that uses GAs for extracting important fuzzy rules from a given rulebase to construct a parsimonious fuzzy model with a high generalization ability. The parameters of the model are estimated using the Kalman filter.

Bonissone et al. [74] applied evolutionary techniques to tune a fuzzy decision system. The fuzzy system automatically classifies the risk of an insurance application, which in turn determines the premium to be paid by the applicant. The evolutionary algorithm tunes decision thresholds and internal parameters of the fuzzy decision system in order to optimizes the coverage and relative and absolute accuracy of the decision process. Maintenance of automated decision systems is critical, as the decision guidelines as well as the distribution of applicants and their profiles changes over time.

In [75], Damousis et al. presented a fuzzy expert system that forecasts the wind speed for power generation in wind farms. The TSK fuzzy model is optimized by a GA that adapts the input fuzzy membership functions and the gain factors in the rule conclusion. The training procedure minimizes the error between forecast and actual wind speeds in the training set. The accuracy of the model was evaluated with real wind data obtained from groups of wind stations located in two different regions.

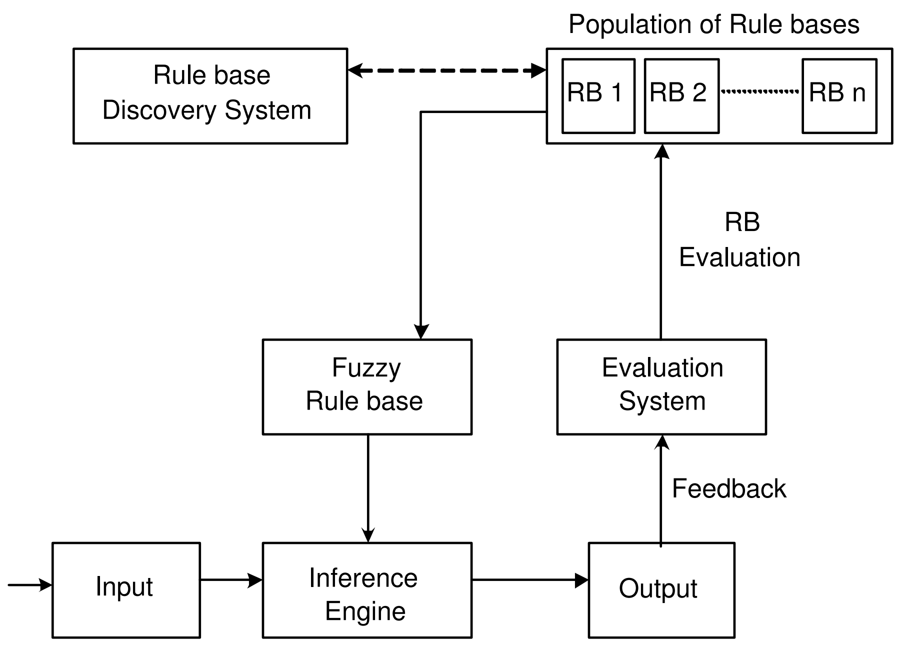


Figure 2.3: Genetic Fuzzy System based on Pittsburgh Approach

Chapter 3

Handling of continuous attributes and application of statistical methods in Rough classifiers

3.1 Overview

This section provides a procedure for handling continuous attributes in rough classifiers. This approach makes use of the Rough Set Knowledge Representation System, namely decision tables. While this section concentrates on continuous attributes alone, the approach can apply for a mixture of discrete and continuous attributes as well. It further shows how various statistical techniques are applied with rough set theory, to handle imprecision and vagueness in data. While probabilistic rough classifier techniques can be used where probability values are available, this section further outlines methods for handling imprecision where probabilities are unknown.

3.2 Introduction

Decision Table as defined by Z.Pawlak [4] is a way for representing knowledge as conditional attributes and decisions corresponding to them. A formal definition for decision tables is provided by Z.Pawlak [4]. Let K = (U,A) be a knowledge representation system and let C, D $\subset$ A where C is the conditional attribute subset and D is the decision attribute subset. The 4-tuple denoted by T=(U,A,C,D) is called the decision table. Such decision table works well for representing discrete attributes, but not suitable for continuous-valued attributes.

While many researchers [76, 77, 78] have developed Rough Classifiers based on Z.Pawlak's Rough Set theory [3, 4] the problem of classification exceeds the basic concepts of rough set theory. Statistical techniques will further enhance classification based on rough set theory. The decision tables help in identification of imprecision or vagueness in the data. After identifying the imprecision, application of statistical techniques will help in further classifying the imprecise data.

The procedure for representation of continuous data in decision tables is demonstrated using Iris flower data. The iris flower data (SAS Institute, 1988) were originally published by Fisher (1936) for examples in discriminant analysis and cluster analysis. Four parameters, including *sepal length, sepal width, petal length* and *petal width*, are measured in meters on fifty iris specimens from each of three species, *Iris setosa, Iris versicolor*, and *Iris virginica.* The dataset is given in full, in Kendall and Stuart (1983) [79].

3.3 Discretization Of Continuous Data

Each of the four attributes for the three flowers fall in a range, shown in Table 3.1.

To represent this continuous data in a decision table for forming decision rules, it has to be discretized.

Table 3.1: Attributes of Iris Flower

Flower Type	**Sepal Length**	**Petal Length**	**Sepal Width**	**Petal Width**
Setosa	0-0.416	0.125-1	0-0.153	0-0.208
Versicolor	0.165-0.749	0-0.584	0.337-0.694	0.376-0.667
Virginica	0.165-1	0.082-0.749	0.51-1	0.541-1

One way to discretize it, is to split the continuous data into ranges and assign integers to the ranges. Granularity of the splitting can play an important role in generating the classification rules.

Each attribute is a scale on which the individual flowers have ranges. Some of these ranges are unique to each flower, while the other are shared. For example, the sepal length range from 0 - 0.165 is unique to Setosa alone, while the range 0.166 - 0.416 is shared by all the flowers. Similarly, the range 0.417 - 0.749 is shared by Versicolor and Virginica whereas, the range 0.750 - 1 is unique to Virginica. Based on this technique of identifying the unique and shared ranges of attributes, table 3.2 shows a possible way to split the continuous data into discrete values.

The above discretization, considers the entire region of an attribute and splits it into smaller regions based on the limits of the individual flower species. This technique has the advantage of clearly de-marking the regions having imprecise data (i.e. data which cannot be clearly classified into one of the three species) and regions having precise data. The entire dataset has been taken into consideration for arriving at these ranges.

Table 3.2: Discretization of Attributes

Sepal Length	**Petal Length**	**Sepal Width**	**Petal Width**
0 - 0.165 : **1**	0 - 0.082 : **1**	0 - 0.153 : **1**	0 - 0.208 : **1**
0.166 - 0.416 : **2**	0.083 - 0.125 : **2**	0.154 - 0.337 : **2**	0.209 - 0.376 : **2**
0.417 - 0.749 : **3**	0.126 - 0.584 : **3**	0.338 - 0.51 : **3**	0.377 - 0.541 : **3**
0.750 - 1 : **4**	0.585 - 0.749 : **4**	0.52 - 0.694 : **4**	0.542 - 0.677 : **4**
	0.750 - 1 : **5**	0.695 - 1 : **5**	0.668 - 1 : **5**

3.4 Decision Table Formation

Once the ranges have been identified and the range numbers are assigned to them, the decision tables can be built. For each record in the data set, replace individual attribute values with the range numbers, of the range in which the values are occurring. Assign a serial number for each record in the data set, so as to revert back to the original values of the attributes at a later stage.

Now that a Decision Table as defined by Z.Pawlak[4] is available, basic rough set concepts are applied to it. The species Setosa has unique attribute values (and hence unique range numbers) for Sepal Width and Petal Width, which clearly classifies it from the other species without any overlap.

Hence, only the other two species, Versicolor and Virginica are considered. The duplicate records are removed so that, only the unique rules are available. It is observed that, out of 100 records (for Versicolor and Virginica alone) available in the original table, a set of 10 unique rules for Virginica and 16 unique rules for Versicolor are formed. There is an overlap of 2 rules occurring in both Versicolor and Virginica. These overlapping rules define the region where data is imprecise. All other regions have clearly defined rules and hence clearly defined classifiability.

The imprecise region is considered separately. The granularity of the range can be made smaller to have finer classifiability. Each of the range numbers of the four attributes in this region can be further subdivided into smaller ranges and the whole procedure can be repeated in this region.

A few iterations like this give a tolerable error range boundary. More data at this stage, will give a better idea about this technique. Statistical techniques are applied to further classify the imprecise data at this stage.

3.5 Statistical Techniques

The formation of the decision table has facilitated the identification of the imprecise region in the data. Statistical techniques provide one way to handle this imprecision.

The basic statistical technique used is the Bayes decision theory. A classifier is implemented which has a set of discriminant functions $g_i(\mathbf{x}), i = 1, 2.., c$ where c corresponds to the total number of classes. For the case of Iris flower c is equal to 3. The classifier assigns feature vector $\mathbf{x}$ to class ω_i if $g_i(\mathbf{x}) > g_j(\mathbf{x})$ for all j $\neq$ i.

The function is set as $g_i(\mathbf{x}) = P(\omega_i|\mathbf{x})$. The a-posteriori probability $P(\omega_i|\mathbf{x})$ is calculated as,

$$P(\omega_i|\mathbf{x}) = \frac{P(\mathbf{x}|\omega_i) * P(\omega_i)}{P(\mathbf{x})} \tag{3.1}$$

where $P(\mathbf{x}|\omega_i)$ is the class conditional probability and $P(\omega_i)$ is the a-priori probability. $P(\mathbf{x})$, calculated as $\sum_{i=1}^{c} P(x|\omega_i) * P(\omega_i)$, is the probability of the feature vector and basically acts as the weight for making the probabilities add upto 1. Readers may refer to [80] for a detailed analysis of the Bayes decision theory.

In the case of decision tables, the feature vector $\mathbf{x}$ corresponds to each row in the decision table. The classifier is used to classify the imprecise data based on the a-posteriori probabilities. However, we need to calculate the a-posteriori probabilities first in order to use this classifier. We need the values of class conditional probability as well as the a-priori probability to calculate the a-posteriori probabilities. Since the a-priori probability is simply the probability of observing a sample from one particular class out of all possible classes, the a-priori probabilities can be usually calculated without many problems. But the class conditional probabilities which are the probabilities of observing a sample from one particular class, having a particular feature, out of all possible classes, are not so easy to estimate, mainly due to lack of samples to cater to all possible cases and also due to problems caused by dimensionality of the feature vector which may be very large. One way to handle this is to estimate the parameters of the underlying distribution. For example, if $P(\mathbf{x}|\omega_i)$ is a multivariate Normal Density with mean μ_i and covariance Σ_i, then the problem is to estimate μ_i, Σ_i. The statistical techniques used to estimate these parameters are called Parametric Techniques. If the underlying density functions are unknown, we can use Statistical Nonparametric methods for estimation of the probability values.

3.6 Parametric Techniques

We can use various techniques like Maximum Likelihood, Bayesian Parameter Estimation (Maximum a-posteriori estimator) etc. to calculate the parameters of the underlying probability density functions.

Maximum Likelihood views the parameters as quantities whose values are fixed but unknown. The best estimate of their value is the one, which maximizes the probability of obtaining the samples actually observed. A detailed analysis of the Maximum Likelihood method and Bayesian Parametric Estimation methods are given in [80]. If θ is the parameter vector (which consists of parameters μ_j, Σ_j for a normal density case), then the p - component vector θ can be estimated from the set of p equations given by

$$\nabla_\theta l = \sum_{k=1}^{n} \nabla_\theta \ln p(x_k|\theta) = 0 \tag{3.2}$$

However, care must be taken with regard to the θ calculated from the above equations. It can represent a global maximum, a local maximum or even a minimum. If all solutions are found, we can be sure that the value of θ obtained is the global optimum.

While Maximum Likelihood considers the parameter vector θ to be fixed, Bayesian Parametric estimation method treats θ as a random variable. The training data is used to convert a distribution on this variable to a-posterior probability density.

Given set of samples D, feature vector $\mathbf{x}$ and class ω_i where $i = 1, 2, .., c$, we obtain the desired class conditional probability $p(\mathbf{x}|D)$ by integrating the joint density function $p(\mathbf{x,}\ \theta|D)$ over θ like this :

$$p(x|D) = \int p\left(x, \theta|D\right) d\theta \tag{3.3}$$

By Bayesian rule, the equation can be rewritten as

$$p(x|D) = \int p\left(x|\theta\right) p\left(\theta|D\right) d\theta \tag{3.4}$$

The Bayesian technique is a Maximum a-posteriori estimator (MAP). MAP estimators have a drawback. If we use an arbitrary nonlinear transformation of the parameter space, say for dimensionality reduction, the density will change and the MAP solution will no longer be appropriate.

These techniques perform well if we know the underlying probability distribution. However, there are many cases where the underlying distribution is unknown. We need to use nonparametric statistical techniques to handle such cases.

3.7 Nonparametric Techniques

Many cases exist where the underlying density function of the dataset is unknown. All of the classical parametric densities are unimodal while many practical problems involve multimodal densities. Nonparametric techniques can be used in such cases. The nonparametric techniques fall under three streams:

- Nonparametric Estimation of density function $p(\mathbf{x}|\omega_i)$: Generalization of multi dimensional histogram approach comes under this category.
- Direct non-parametric estimation of the a-posteriori probability $P(\omega_i|\mathbf{x})$: Related to Nearest Neighbor rule which bypass probability estimation and directly get into decision functions.
- Transforming the feature space: Usually estimation in transformed space is easier to handle. These are the discriminant analysis techniques.

In particular we will look at Nearest Neighbor rule, which can be used for direct estimation of a-posteriori probabilities. A detailed treatment of all these techniques can be found in [80].

The Nearest Neighbor technique is used to estimate a-posteriori probabilities $P(\omega_i|\mathbf{x})$ from a set of n labeled samples as follows. If a cell of volume V is placed around $\mathbf{x}$ which captures k samples, k_i of which are labeled ω_i, then the joint probability $p(\mathbf{x}, \omega_i)$ is given by,

$$p_n(x, \omega_i) = \frac{(k_i/n)}{V} \tag{3.5}$$

Thus a reasonable estimate of $P(\omega_i|\mathbf{x})$ from Bayes rule is given by,

$$P_n(\omega_i|x) = \frac{p_n(x, \omega_i)}{\sum_{i=1}^{c} p_n(x, \omega_i)} \tag{3.6}$$

So,

$$P_n(\omega_i|\mathbf{x}) = k_i/k \tag{3.7}$$

The a-posteriori probability is merely the fraction of samples within the cells labeled as ω_i. If the cell volume can become arbitrarily small and yet contain an arbitrarily large number of samples, then the probabilities can be calculated with virtual certainty and obtain optimum performance. This is the theoretical basis behind Nearest Neighbor technique.

In practice, the Nearest Neighbor rule is applied as follows. Let $D^n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ denote a set of n labeled samples and let $\mathbf{x}' \in D^n$ be the sample nearest to x. Then, according to Nearest Neighbor rule, $\mathbf{x}$ is assigned the same class as $\mathbf{x}'$. It has been proven that the error rate of Nearest Neighbor Rule is lower bounded by the Bayes error rate, which is the optimal error rate. On the upper side it is bounded by twice the Bayes error rate. However, it can also be shown that the convergence can be arbitrarily slow and the error rate need not even decrease monotonically.

An extension of the Nearest Neighbor Rule is the k-Nearest Neighbor rule. This rule classifies $\mathbf{x}$ by assigning it the same class as that one which is most frequently represented among the k nearest samples. As k increases, the upper bounds of error get progressively closer to the lower bound, which is the Bayes rate. As k goes to infinity, the two bounds meet and the k-Nearest Neighbor rule becomes optimal.

A large value of k provides a reliable estimate of the probability. On the other hand, the k nearest neighbors should be as close to $\mathbf{x}$ as possible to provide the closest approximation to $P(\omega_i|\mathbf{x}\ )$. Hence we need to choose a compromise value for k. In this case, the k-Nearest Neighbor Rule becomes optimal when n approaches infinity.

The Nearest Neighbor classifier relies on the distance function to identify the nearest neighbors. While Euclidean distance can be used as the distance measure, it does have some disadvantages. Scaling the coordinates of the attribute space can change the distance relationships computed by Euclidean metric. Other transformations such as overall rotation would also not be well accommodated by Euclidean distance. An example of this is given in [80] for handwriting classification. The same reference proposes the construction of a tangent vector for each transformation as a solution.

Other Nonparametric classifiers such as Probabilistic Neural Networks and Fuzzy Classifiers are available as well. However they cannot be strictly classified as statistical techniques. Interested readers may refer to [80] for more details on these classifiers. Statistical techniques like Fisher's Linear Discriminant functions and Minimum Squared error methods can be used to reduce dimensionality of the attribute space. However, rough set theory itself provides techniques for dimensionality reduction by formation of reducts and hence these statistical techniques have not been explored in this chapter. Since rough sets basically deal with labeled samples, unsupervised statistical classification techniques like clustering are also not explored in this paper.

So far, we have been concentrating on techniques for estimating the class conditional and a-posteriori probabilities. However, in many practical situations we need to consider the cost of decision making in addition to the probabilities. It may be more costly to make one kind of classification error than the other. So we need to build a classifier based on the minimization of the misclassification cost in addition to the a-posteriori probabilities.

In our classifier, instead of having the discriminant function as $g_i(\mathbf{x}) = P(\omega_i|\mathbf{x})$, we modify it to include a cost component also. Let λ_{ij} be the loss incurred in classifying the sample as ω_i when the true state of nature is ω_j. Then, the conditional risk for making classification as ω_i (denoted as α_i) for the sample with feature vector $\mathbf{x}$ is given by the equation,

$$R\left(\alpha_i|x\right) = \sum_{j=1}^{c} \lambda_{ij} P\left(\omega_i|x\right) \tag{3.8}$$

Then $g_i(\mathbf{x})$ is given as,

$$g_i(x) = -R(\alpha_i|x) \tag{3.9}$$

This representation of $g_i(\mathbf{x})$ helps us in selecting the action α_i for which the overall risk is the minimum. The resulting minimum overall risk is called the Bayes Risk (R*).

To minimize the probability of error, we can introduce the following costs (called symmetric loss function) in the above equation,

$$\begin{aligned} \lambda_{ij} &= 0 \; for \; i = j \\ \lambda_{ij} &= 1 \; for \; i \neq j \end{aligned} \tag{3.10}$$

This assigns no loss for a correct decision and unit loss for all errors, thereby making all errors equally costly. Then the risk is nothing but the average probability of error because the above equations lead to

$$R(\alpha_i|x) = 1 - P(\omega_i|x) \tag{3.11}$$

A procedure for construction of a rule inductive classifier is given in Lin *et al.* [81]. It makes use of the above mentioned loss functions for building a classifier. Lin *et al.* [81] also makes use of kernel based and frequency based estimators for approximating the probabilistic structure of the data.

3.8 Illustrative Example and discussion

As an example we have combined the Nearest Neighbor Technique (NNR) with the presented Rough Set discretization technique and applied it on a thyroid gland classification dataset. The thyroid dataset provided by Garavan Institute, Sydney, Australia was obtained from the UCI Repository of Machine Learning Databases and Domain Theories. The dataset consists of 3 classes, 215 instances, 5 attributes without any missing values.

The 3 classes were split equally into training and test sets. The discretization technique specified in Section 3.3 is applied on the training set and a decision table is formed with the unique rules derived from the training set.

The test set is also discretized. The discretized test records are classified using the decision table. The test set is found to contain 19 records for normal (class1), 8 records for hyper thyroid (class2), 6 records for hypo thyroid (class 3), which are unique and available only in the test set.

Since these unique records are nothing but new classification rules which did not occur in the training set, the decision table cannot be used for classifying them. So, the Nearest Neighbor (NNR) method is used for classifying these records. The results which give the number of records classified, misclassified, unclassified are as follows:

Table 3.3: Normal(Class1), 75 Training set,75 Test set

	Rough set	**NNR**	**Rough set + NNR**
Classified	49	73	74
Misclassified	0	2	1
Unclassified	26	0	0
Average Time in millisec	160	1200	520

As can be seen from the results, there is a slight improvement for class1, when this method is employed. However, the biggest benefit is the speed of classification. Using rough sets, the classification is just a table lookup procedure. And NNR is used only for unclassified cases. Using NNR alone is much slower as the distance has to be calculated for each test record and the nearest neighbor has to be identified.

Table 3.4: Hyper (Class2), 17 Training set, 18 Test set

	Rough set	**NNR**	**Rough set + NNR**
Classified	7	16	16
Misclassified	1	2	2
Unclassified	10	0	0
Average Time in millisec	40	340	250

Table 3.5: Hypo (Class3), 15 Training set, 15 Test set

	Rough set	**NNR**	**Rough set + NNR**
Classified	4	11	11
Misclassified	3	4	4
Unclassified	8	0	0
Average Time in millisec	40	330	250

The rough + NNR classifier can be made into a learning classifier which will further speed up the classification as well as the accuracy. In the learning classifier, any new rule encountered will be added to the decision table. This can classify similar future records and NNR need not be employed for these similar records.

The proposed classifier performs well with a good time response, on datasets which have a good representation of samples. However, if the dataset consists of a large number of unknown samples whose attribute ranges were not represented in the dataset, then the classifier is forced to rely on the NNR technique rather than the rough set technique for classification, resulting in degradation of time performance.

3.9 Summary

The presented classifier combines features of Knowledge Representation System based on rough set theory with statistical approximation of dataset's probabilistic structure. While the procedure has been proposed for handling continuous attributes, it can work equally well with a mixture of continuous and discrete attributes. Various statistical techniques are outlined for the estimation of the probabilistic structure when the underlying probability density functions are known as well as when they are unknown. A statistical classifier using discriminant functions is proposed, which further builds on the outcome of the rough classifier.

A statistical technique for incorporating costs in decision making is also outlined. A particular form of the classifier uses symmetric loss functions for minimizing the error probability.

Finally an illustrative example is shown, which combines the rough classifier with the Nearest Neighbor Method to speed up the classification process.

Chapter 4

Genetic Algorithm Based Heuristic Fuzzy Pattern Classifier

4.1 Overview

A fuzzy classification technique for multidimensional pattern classification problems with continuous and discrete attributes is discussed in this section. The proposed method combines the best characteristics of the Pittsburgh based technique by Shi *et al.* [2] and the Michigan based technique by Ishibuchi *et al.*[60]. The proposed approach has three main objectives: to minimize the number of misclassifications, to minimize the number of fuzzy if-then rules required for classification and to remove unnecessary attributes from the fuzzy if-then rules. The objectives are accomplished by evolving the membership function shapes of the antecedent attributes and the fuzzy rule set including the number of rules involved, using a genetic algorithm(GA). The performance of this algorithm is illustrated on three well known real world test problems. The results are compared with previously documented results. Directions for further improvement of this approach are discussed.

4.2 Introduction

4.2.1 Fuzzy Pattern Classification: Background

Lately, a large number of commercial and industrial fuzzy systems have been successfully developed. Fuzzy systems are being employed in wide ranging areas from consumer electronic products, automotive and subway systems to cement kiln control and chemical injection control in water purification plants. The most successful domain is the area of fuzzy control systems. In most existing applications, domain experts generate the fuzzy rules.

With increase in the number of input variables, it becomes difficult for a domain expert to define a complete set of fuzzy rules for optimum performance. This has greater significance in the area of fuzzy pattern classification, where high dimensionality of classification problems lead to an explosion of rules. While there are other techniques such as Neural networks, statistical methods etc., for pattern classification, the use of linguistic variables and the ease in interpretation of fuzzy if-then rules make fuzzy classification systems an attractive alternative.

Hence, it is important to have automated techniques, for the generation of fuzzy rules from numerical data, to construct Fuzzy Systems. There are many existing techniques such as c-means clustering, fuzzy c-means clustering [40, 42, 82] etc. These techniques however, result in rules which are independent of the membership functions, due to which the resultant fuzzy system may not be an optimal one, especially for complex systems with large number of input variables. One way to improve the performance of fuzzy systems is to select suitable membership functions and to fine tune them further.

4.2.2 Fuzzy Pattern Classification: Existing Approaches

In the design of fuzzy pattern classifier, the number of classified patterns and the number of fuzzy rules are two important performance indicators. Choice of fuzzy partitions, membership functions, fuzzification and defuzzification methods etc., are all important parameters in deciding the performance of a fuzzy system. The automatic generation of a fuzzy system from numerical data can be considered as an optimization or search process.

Genetic algorithms (GA) are one of the best known global search techniques, for exploring the operating space using available performance measures. GAs are capable of finding near optimal solutions in complex search spaces. Given some performance criteria, the solutions for a fuzzy pattern classification problem form a hyper-surface in space. Due to infinitely large, non-differentiable, complex, noisy and multimodal nature of this hyper-surface, GAs are especially well suited to search this surface when compared with other conventional methods [2].

Recently many alternative approaches [83, 84, 85, 40] based on Genetic Algorithms have been proposed for evolving fuzzy systems. Carse *el al.* [86] provides a survey of genetic algorithm based techniques for generating fuzzy if-then rules and tuning the membership functions.

Most of these approaches encode rules into chromosomes and utilize GA to adjust the rules or membership functions. Hwang [62] and Thrift [63] use fixed membership functions while encoding all rules in the chromosomes. The fuzzy membership functions are tuned indirectly in [64], by adjusting the critical points that represent the membership functions. Hamaifar [65] and Takagi *el al.* [66] tune membership functions and simultaneously evolve rule sets. Triangular membership functions are used and all possible rules are encoded in a chromosome. Though membership functions and rule sets are evolved simultaneously, these methods have some drawbacks. Since all possible rules are encoded, the computational efficiency as well as interpretability of the rules associated with fuzzy logic are lost. It is difficult to apply these techniques for classification of patterns having high dimensionality. As most applications do not require all possible rules, only a portion of the rules need to be encoded and evolved.

In fact, it is highly difficult to know before-hand, as to how many rules are required to describe a fuzzy system. It is better to encode the number of rules, along with the rules and membership functions. Inoue *et al.* [87] and Shimojima *et al.* [88] encoded the membership functions and the effectiveness information of each rule. Fitness functions used in [88] encouraged minimizing the number of rules.

Shi *et al.* [2] proposed a technique to evolve conventional fuzzy systems using GA. The number of rules along with the membership functions and rule set are encoded in a chromosome. A combination of linear (*triangular, left-triangular etc.*) and nonlinear (*Gaussian, Sigmoid etc.*) functions are used as membership functions. The parameters of the GA are determined dynamically using a fuzzy system. GA is used to tune the membership functions and the rule set to minimize misclassifications. The number of rules are also evolved to minimize the number of fuzzy if-then rules.

The above mentioned methods for generating fuzzy if-then rules and tuning membership functions can be categorized as Pittsburgh approach (Fig. 2.3). In this approach, each chromosome encodes a rule base. The performance of the rule base is used as the chromosome's fitness value. Hence, the genetic search for finding rule sets with high fitness value is equivalent to search for fuzzy systems with high performance. GA optimizes the entire fuzzy system, rather than individual fuzzy if-then rules. There is an alternative approach to generate fuzzy if-then rules through GA: the Michigan approach.

In the Michigan approach (Fig. 2.2), each chromosome encodes a single rule and the rule set is represented by the entire population. The performance of each rule decides its fitness value as opposed to that of the entire rule base as in Pittsburgh approach. GA optimizes individual rules, rather than the entire fuzzy system.

The performance of a fuzzy classification system depends on the choice of fuzzy partitions. For certain applications, fine partitioning is required for some parts of the pattern space and coarse partitioning for others. To cope with this difficulty, Ishibuchi *et al.* [38] introduced the concept of distributed fuzzy if-then rules. They encoded all the fuzzy if-then rules corresponding to several different fuzzy partitions into a tri value string set {-1, 0, 1}, and applied GA to remove the unnecessary rules. As every possible rule for each subspace is coded into a chromosome, the length of the chromosome becomes very large for problems with high dimensionality.

Ishibuchi *et al.*[60] utilized *"don't care"* value for attributes, to simplify and reduce the number of fuzzy if-then rules for high dimensional pattern classification problems. Triangular membership functions are used with homogeneous fuzzy partitioning and the fuzzy if-then rules are generated heuristically. The authors proposed the use of a heuristic modifier to conventional fuzzy rules termed as Certainty Factor. The Certainty Factor facilitates the generation of classification boundaries without modifying the membership functions. The membership functions are fixed a-priori and the GA is used to tune only the rule set.

4.2.3 Proposed Approach

In this work, a GA based fuzzy classification system is discussed. In the proposed system, the membership function shapes and the fuzzy rule set, along with the number of rules are evolved using the Pittsburgh approach based Genetic Algorithm. While the proposed approach can evolve different membership function types, only the Gaussian membership functions are used. The fuzzy system adopts some characteristics of the approach mentioned in [60] and utilizes "don't care" values for attributes to remove unwanted attributes. The proposed system makes use of an encoding scheme similar to that proposed in Shi *et al.* [2] with some modifications.

In Ishibuchi's *et al.* [60] proposal, the Michigan approach based fuzzy system neither tunes the membership functions nor minimizes the number of fuzzy if-then rules. Shi's *et al.* [2] proposal based on Pittsburgh approach, makes use of conventional fuzzy if-then rules. Since conventional fuzzy rules are utilized without Grade of certainty (CF) which provides better convergence due to the modification of classification boundaries, the generation of fuzzy rules takes a longer time.

The current proposal based on Pittsburgh approach, combines the best features of these two proposals by utilizing fuzzy rules with Grade of certainty, while tuning the membership functions and minimizing the number of fuzzy if-then rules. The effectiveness of the system in convergence speed and in producing a compact, easily comprehensible rule set, is demonstrated with several well known real-world test problems.

The rest of the chapter is organized as follows : In section 4.3, the structure of the fuzzy system for pattern classification is described. In section 4.4, the fuzzy system design by genetic algorithm is outlined. In section 4.5, the performance of the proposed approach is illustrated with the test problems. In section 4.6, directions for further improvement of this approach are provided. Finally, section 4.7 concludes the Chapter.

4.3 Fuzzy System Structure

4.3.1 Fuzzy if-then rules

The proposed approach makes use of fuzzy if-then rules of the form

$$\begin{aligned} &\text{Rule } R_j: \text{ If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn} \\ &\quad \text{then Class } C_j \text{ with } CF = CF_j, \\ &for\ j = 1, 2, \ldots, c, \end{aligned} \tag{4.1}$$

where,

R_j	Label of the jth fuzzy if-then rule
A_{j1},...,A_{jn}	Antecedent fuzzy sets
C_j	Consequent class
CF_j	Grade of certainty of rule R_j.

Gaussian membership functions are utilized in this work. The grade of certainty CF_j of a rule rule R_j, indicates as to how uniquely the rule classifies samples of class C_j. The heuristic method proposed by Ishibuchi et al.[38] to determine the class label C_j and the grade of certainty CF_j from given training patterns, is used in this work.

4.3.2 Fuzzy Reasoning

Consider K fuzzy if-then rules generated from the training patterns. An input vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is classified by the single winner rule $R_{j'}$ that has the maximum product of compatibility and certainty grade:

$$\mu_{j'}(\mathbf{x}) \cdot CF_{j'} = \max\{\mu_j(\mathbf{x}) \cdot CF_j \mid j = 1, 2, \ldots, K\}, \tag{4.2}$$

where, $\mu_j(\mathbf{x})$ is the compatibility of the input vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ with the fuzzy if-then rule R_j, which is defined by the product operator as follows:

$$\mu_j(\mathbf{x}) = \mu_{j1}(x_1) \cdot \mu_{j2}(x_2) \cdots \mu_{jn}(x_n), \tag{4.3}$$

where, $\mu_{ji}(x_i)$ is the membership function of the antecedent fuzzy set A_{ji} defined for the i-th attribute a_i. The rule $R_{j'}$ is specified by (4.2) as the single winner rule for the outlined fuzzy reasoning procedure. The input vector $\mathbf{x}$ is classified as the class label $C_{j'}$ of the single winner rule $R_{j'}$ specified by (4.2).

4.4 Fuzzy System Design By Genetic Algorithm

The basic idea of a GA is to maintain a population of chromosomes, which represents candidate solutions to the problem. The population evolves over time through a process of competition and controlled variation. Each chromosome in the population has an associated fitness. Based on the fitness values, new chromosomes are generated using genetic operators such as *crossover* and *mutation*.

4.4.1 Coding of Fuzzy If-Then Rule

In the proposed approach, only the antecedent fuzzy sets are encoded in chromosomes. The number of fuzzy sets or linguistic variables for each attribute axis is pre-specified. Each linguistic variable is associated with a Gaussian membership function. Definition of the Gaussian membership function utilized in this chapter is as follows.

$$f_{Gaussian}(x) = e^{-0.5y^2} \quad \text{where } y = \frac{8(x - x_1)}{x_2 - x_1} - 4 \tag{4.4}$$

The membership function is completely determined by two values: the start_point x_1 and the end_point x_2. In order to have homogeneous chromosomes, integers are chosen to represent x_1 and x_2. Let $[r_1, r_2]$ be the dynamic range of variable x and let x consist of n fuzzy sets. If the fuzzy memberships functions are uniformly distributed over the range with half-way overlap as shown in Fig. 4.1, then the center point $c_i (i = 1, \cdots, n)$ of the ith membership function is located at

$$c_i = r_1 + i * \text{step} \quad i = 1, \cdots, n, \quad \text{where step } = \frac{r_2 - r_1}{n+1} \tag{4.5}$$

The start_point x_1^i of the ith membership function is constrained to vary only between c_{i-1} and c_i. The end_point x_2^i is similarly constrained to vary only between c_i and c_{i+1}. Let an integer s ($s = 0, \cdots, 10$, *indicating actual divisions on attribute axis*) be used to represent x_1^i and x_2^i. Then x_1^i and x_2^i can be calculated as:

$$x_1^i = i * \text{step} - \frac{\text{step} * (10 + s)}{2 * 10} + r_1 \tag{4.6}$$

$$x_2^i = i * \text{step} + \frac{\text{step} * (10 + s)}{2 * 10} + r_1, \quad i = 1, \cdots, n. \tag{4.7}$$

The number of fuzzy if-then rules needed to design a high performance fuzzy system, cannot be known a-priori. Hence, the number of fuzzy rules is also encoded as a constraint in the chromosome. Consider an example fuzzy system consisting of four attributes. Each attribute is split into three fuzzy sets. The maximum number of rules be set to twenty. A sample chromosome is then encoded as,

$$b_1 b_2 b_3 \cdots b_{104} b_{105} \tag{4.8}$$

The detailed encoding schema is outlined in Table 4.1.

4.4.2 Initial Population and Fitness Function

The population strength is decided in order to provide a sufficient genetic pool base. Initial population is randomly generated, while ensuring that individual bits

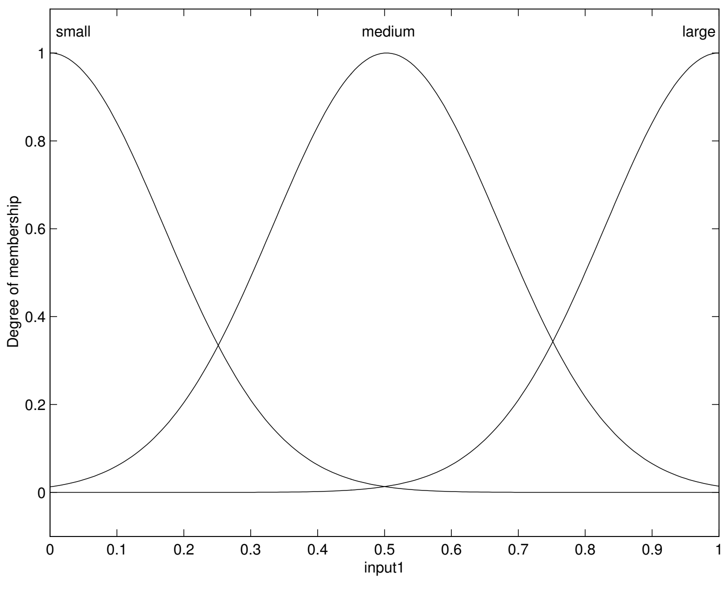


Figure 4.1: Uniformly distributed membership functions

Table 4.1: Encoding of chromosomes

b_1	Number of fuzzy rules ranging from 1 to 20
b_2, b_3	Start and End points of the first fuzzy set, of the first attribute. Values can range between 0 and 10
$\vdots$	$\vdots$
b_{24}, b_{25}	Start and End points of the third fuzzy set, of the fourth attribute. Values can range between 0 and 10
b_{26}	Linguistic variable for the first attribute of the first fuzzy rule. Values can range from 0 to 3. The absence of this attribute or in other words, the "*don't care*" value of this attribute, is represented by 0. The values 1 to 3, represent the linguistic variables of the corresponding fuzzy sets, of this attribute
b_{27}, b_{28}, b_{29}	Remaining three attributes of the first fuzzy rule
$\vdots$	$\vdots$
$b_{102}, b_{103}, b_{104}, b_{105}$	The linguistic variables for the four attributes of the twentieth rule

in the chromosomes are within the respective ranges (Table 4.1).

The cost of misclassification is assumed uniform across the classes. The fitness function is determined by the performance of the fuzzy system. The performance of the classification system is dependant on two important factors: the number of misclassifications and the number of fuzzy if-then rules. Hence, the fitness function is encoded as the sum of the number of misclassifications and the number of fuzzy rules. The lower the fitness value, the better is the performance of the fuzzy classification system.

4.4.3 Selection for reproduction

One of the commonly used techniques is the "*Roulette Wheel*" selection method. In this technique, each chromosome is allocated a sector on a "*Roulette Wheel*", proportionate to its fitness value compared to other chromosomes. A chromosome is selected by choosing a random number (considered as the pointer of "*Roulette Wheel*") across all sectors of the "*Roulette Wheel*". The higher the fitness of a chromosome, the bigger is the sector allocated to it and better is the probability of the chromosome being selected.

The Stochastic Universal Sampling (SUS), which is a derivative of the roulette wheel method, is employed for selection in this work. Instead of the single pointer used in the roulette wheel method, SUS uses N equally spaced pointers, where N is the number of selections required. Since multiple selections are completed in a single attempt, SUS is a simpler algorithm with a lesser time complexity.

4.4.4 Genetic Operators

Crossover and Mutation are two important operators in GA which play a critical role in the exploration of search space and the convergence of GA.

Crossover Operator The crossover operator facilitates the exchange of "*genetic*" material to form "*child*" chromosomes from selected "*parent*" chromosomes. Uniform Crossover method is employed in this work. A crossover mask with the same length as of the chromosomes, is created at random. The mask bits are binary valued. The mask bits indicate as to which parent contributes to the offspring and with which bits. Table 4.2 illustrates Uniform Crossover. The first child C_1 is produced by taking a bit from P_1 if the corresponding mask bit is 1, or from P_2 if the corresponding mask bit is 0. Child C_2 is formed in a similar fashion, but by using the inverse of the mask.

Table 4.2: Uniform Crossover

P1	=	0	2	4	6	8
P2	=	1	3	5	7	9
Mask	=	0	0	1	1	0
C1	=	1	3	4	6	9
C2	=	0	2	5	7	8

Mutation Operator As illustrated in Table 4.1, the chromosomes use integer encoding. Each bit in the chromosome has an integer range according to the parameter it represents in the fuzzy system. For example, s_1 which represents the number of rules considered in the fuzzy system, has an integer range from 1 to 20. If a bit is chosen for mutation as per the mutation rate which is fixed at 0.01, it is incremented or decremented by 1 within its range. If the selected bit has the lower limit of its range as the value, it is incremented by 1. However, if it has the upper limit of its range as the value, it is decremented by 1. For all others values, the bit is mutated with an equal 50% chance to get incremented or decremented.

4.4.5 Re-insertion of chromosomes

Once a new population is generated, the fitness of the new population is evaluated. In the commonly used *elitist strategy*, a percentage of the population comprising of the fittest individuals, is allowed to propagate through the subsequent generations. It is however noticed that some of the child chromosomes, despite having a lesser fitness compared to the parent chromosomes, still replace fitter parents in the *elitist strategy*. Hence a modification is proposed in the current approach where the subsequent generations are formed from fittest individuals, selected from a combined parent and child pool.

4.4.6 Termination of GA

The GA is terminated once the specified fitness value for the chromosomes is reached, or the specified number of generations is completed.

4.5 Performance Evaluation And Discussion

Various data sets such as Iris data set, Wine data set etc. are commonly used for performance evaluation in the literature. The performance of the proposed fuzzy classification system is illustrated on the Iris, Wine and Glass data sets.

4.5.1 Performance Evaluation: Iris Data Set

The iris flower data set (SAS Institute, 1988) originally published by Fisher (1936) for examples in discriminant analysis and cluster analysis, is available from the

University of California, Irvine database by anonymous ftp [89]. The four attributes associated with Iris flower: sepal length, sepal width, petal length and petal width, are measured on Fifty iris specimens from each of three species, Iris setosa, Iris versicolor, and Iris virginica. The attribute values are normalized in the range [0,1]. The problem is to classify the specimens into three classes based on the four attributes.

For the purpose of illustration, a crossover rate of 0.9 and a mutation rate of 0.1 are considered. The population size is fixed to 40 chromosomes. The GA is allowed to run for a maximum of 50 generations. The number of newly created chromosomes in each generation is limited to 90% of the population. In each generation, a new population is formed based on fitness value from a combined parent and child pool. Fitness is considered as the sum of the number of misclassifications and the number of fuzzy rules. The smaller the fitness value, the better is the performance of the fuzzy classifier system.

To check the convergence speed rather than the generalization capability, all 150 patterns are used as training patterns. The experiment is conducted for 10 runs. The number of generations taken to reach the specified number of misclassifications within 50 generations is determined. The experiment is performed thrice, once each with 3 fuzzy sets, 4 fuzzy sets and 5 fuzzy sets per attribute. The results are provided in Tables 4.3 - 4.5. The right-most column of each table provides the average number of generations and the average number of rules required, to reach the number of misclassifications listed in the left-most column.

Table 4.3: Three fuzzy sets per attribute for Iris Data Set

Misclassified	Generations to reach specified misclassifications / Number of fuzzy rules needed										Average
	Run1	**Run2**	**Run3**	**Run4**	**Run5**	**Run6**	**Run7**	**Run8**	**Run9**	**Run10**	
9	/	/	/	/	/	/	3 / 10	/	/	/	3 / 10
8	/	1 / 9	/	/	/	/	/	/	/	/	1 / 9
7	1 / 10	2 / 8	1 / 8	/	4 / 7	1 / 6	4 / 6	4 / 7	1 / 4	/	2.25 / 7
6	/	3 / 8	9 / 8	/	/	4 / 6	/	5 / 7	/	8 / 5	5.8 / 6.8
5	6 / 5	9 / 8	/	/	8 / 7	7 / 8	6 / 10	15 / 7	/	12 / 5	9 / 7.1
4	16 / 4	11 / 8	12 / 5	2 / 8	9 / 7	17 / 9	/	16 / 7	8 / 4	15 / 5	11.8 / 6.3
3	21 / 5	/	18 / 6	19 / 6	10 / 7	19 / 6	9 / 10	25 / 6	23 / 4	17 / 5	17.9 / 5.55
2	49 / 4	16 / 8	34 / 6	/	/	23 / 6	18 / 5	31 / 7	/	21 / 5	27.4 / 5.85
1	/	29 / 8	/	/	/	/	/	/	/	/	29 / 8

"/" refers to the case where the specified number of misclassifications did not occur within maximum generations

Eg: 1/10 in the third row of Run1 refers to 7 misclassifications in the first generation with ten fuzzy if-then rules

Average number of generations is calculated only for those misclassifications found within maximum generations

Table 4.4: Four fuzzy sets per attribute for Iris Data Set

Misclassified	**Generations to reach specified misclassifications / Number of fuzzy rules needed**										**Average**
	Run1	**Run2**	**Run3**	**Run4**	**Run5**	**Run6**	**Run7**	**Run8**	**Run9**	**Run10**	
9	/	3 / 9	/	/	/	/	/	/	/	/	3 / 9
8	/	4 / 9	1 / 4	/	/	2 / 10	/	/	/	/	2.33 / 7.67
7	2 / 10	5 / 8	2 / 4	/	/	4 / 7	/	4 / 7	/	1 / 8	3 / 7.33
6	/	/	/	/	1 / 3	9 / 8	2 / 5	/	1 / 8	/	3.25 / 6
5	10 / 7	7 / 9	/	2 / 7	12 / 8	/	4 / 5	/	4 / 8	8 / 10	6.7 / 7.7
4	15 / 6	/	8 / 8	9 / 9	17 / 8	10 / 7	10 / 4	6 / 10	15 / 6	10 / 10	11.1 / 7.55
3	31 / 6	18 / 9	16 / 5	39 / 5	24 / 8	33 / 6	18 / 5	24 / 5	23 / 6	19 / 7	24.5 / 6.2
2	35 / 6	25 / 8	/	/	/	/	25 / 4	27 / 5	/	/	28 / 5.75
1	37 / 6	/	/	/	/	/	/	/	/	/	37 / 6

"/" refers to the case where the specified number of misclassifications did not occur within maximum generations

Eg: 2/10 in the third row of Run1 refers to 7 misclassifications in the 2nd generation with ten fuzzy if-then rules

Average number of generations is calculated only for those misclassifications found within maximum generations

Table 4.5: Five fuzzy sets per attribute for Iris Data Set

Misclassified	**Generations to reach specified misclassifications / Number of fuzzy rules needed**										**Average**
	Run1	**Run2**	**Run3**	**Run4**	**Run5**	**Run6**	**Run7**	**Run8**	**Run9**	**Run10**	
9	/	/	/	/	/	/	/	/	/	/	/
8	1 / 6	/	/	1 / 5	1 / 7	3 / 8	1 / 7	5 / 5	/	/	2 / 6.3
7	/	2 / 9	/	/	8 / 7	/	7 / 10	/	1 / 7	/	4.5 / 8.25
6	5 / 6	/	4 / 10	2 / 5	9 / 8	/	11 / 5	6 / 5	/	5 / 7	6 / 6.6
5	10 / 7	5 / 9	15 / 4	/	11 / 7	6 / 10	14 / 5	/	7 / 9	11 / 5	9.9 / 7
4	12 / 5	7 / 9	24 / 5	5 / 9	16 / 7	11 / 10	17 / 5	/	10 / 7	20 / 5	13.56 / 6.9
3	15 / 5	25 / 5	39 / 4	13 / 5	24 / 10	13 / 10	20 / 7	9 / 6	18 / 5	/	19.56 / 6.3
2	27 / 5	30 / 5	/	39 / 5	39 / 7	41 / 7	46 / 5	46 / 6	25 / 5	29 / 5	35.8 / 5.56
1	41 / 5	/	/	/	/	/	/	/	/	/	41 / 5

"/" refers to the case where the specified number of misclassifications did not occur within maximum generations

Eg: 1/6 in the second row of Run1 refers to 8 misclassifications in the first generation with six fuzzy if-then rules

Average number of generations is calculated only for those misclassifications found within maximum generations

Shi *et al.* [2] examined the performance of their system on the iris data set. The results in Table 4.6 are reported in [2]. The entire fuzzy classification system with 3 fuzzy sets per attribute is evolved along with the membership functions. A fuzzy expert system is also used to adapt the crossover and mutation rates. The system is evolved for a maximum of 1000 generations.

Table 4.6: Result reported by Shi *et al.* [2] for Iris Data Set

Misclassifications	**Average Generations**
9	13.9
8	18.9
7	22.3
6	30.1
5	59.9
4	105.9
3	143.2

As per the results (Tables 4.3, 4.4 and 4.5), the proposed approach converges at a faster rate compared to the method in [2], even with a single membership function type (Gaussian) and without a fuzzy expert system to adapt the crossover and mutation rates. In addition, the proposed method resulted in better performance with one or two misclassifications while the best performance reported in [2] is three misclassifications.

The results indicate that as the number of fuzzy sets per attribute increases, the number of rules required is smaller. However, the number of iterations required for GA to converge increases slightly. Though the number of fuzzy sets is increased, the use of "*don't care*" value for attributes reduced the complexity of the rules by eliminating some of the attributes from the rules. The five rules for achieving a single misclassification, in the case of five fuzzy sets ({*tiny, small, medium, large,* *very large}* *and the "don't care" value*) per attribute are provided below. The attributes with "*don't care*" value have been removed from the rules:

1. If *sepal length* is tiny (Sigma = 0.030208, Mean = 0.1875), and *petal length* is large (Sigma = 0.034375, Mean = 0.67083), and *petal width* is small (Sigma = 0.038542, Mean = 0.3375), then classification is *Setosa* with CF = 1

2. If *sepal length* is medium (Sigma = 0.032292, Mean = 0.47917), and *sepal width* is medium (Sigma = 0.032292, Mean = 0.52083), and *petal width* is small (Sigma = 0.038542, Mean = 0.3375), then classification is *Versicolor* with CF = 1

3. If *sepal length* is small (Sigma = 0.033333, Mean = 0.33333), and *petal length* is tiny (Sigma = 0.030208, Mean = 0.1875), and *sepal width* is large (Sigma = 0.039583, Mean = 0.65833), and *petal width* is very large (Sigma = 0.023958, Mean = 0.84583), then classification is *Virginica* with CF = 1

4. If *sepal length* is small (Sigma=0.033333, Mean=0.33333), and *sepal width* is very large (Sigma=0.028125, Mean=0.8375), then classification is *Virginica* with CF = 0.99932

5. If *petal length* is small (Sigma=0.029167, Mean=0.33333), and *sepal width* is small (Sigma=0.034375, Mean=0.30417), and *petal width* is very large (Sigma=0.023958, Mean=0.84583), then classification is *Virginica* with CF = 1

4.5.2 Performance Evaluation: Wine Data Set

The Wine data set is obtained from the University Of California, Irvine database [89]. It consists of 178 samples with 13 continuous attributes from 3 classes. Corcoran *et al.* [90] applied the wine data set to their GA evolved Machine learning system, based on Pittsburgh approach. The test is repeated over ten trials. Each trial is performed with 60 non-fuzzy if-then rules in each chromosome, 1500 chromosomes in each population and 300 generations. All 178 samples are used as test data and the following results are reported in [90]:

- **Best classification rate**: 100% with 60 rules in 300 generations / 1500 chromosomes in each population
- **Average classification rate**: 99.5% with 60 rules in 300 generations / 1500 chromosomes in each population
- **Worst classification rate**: 98.3% with 60 rules in 300 generations / 1500 chromosomes in each population

Ishibuchi *et al.* [60] applied the wine data set to their Michigan approach based fuzzy classifier system. The data is normalized to the unit interval [0, 1]. Each fuzzy if-then rule is encoded in a single chromosome. The trial is conducted for 1000 generations, with 60 chromosomes in each population. The following results are reported in [60]:

- **Best classification rate**: 99.4% with 60 rules in 1000 generations / 60 chromosomes in each population

- **Average classification rate**: 98.5% with 60 rules in 1000 generations / 60 chromosomes in each population
- **Worst classification rate**: 97.8% with 60 rules in 1000 generations / 60 chromosomes in each population

In an extension to their base work, Ishibuchi *et al.* [60] proposed a method to learn the Grade of Certainty CF and they reported the following result:

- **Average classification rate**: 100% with 60 rules in 138 generations / 60 chromosomes in each population

The proposed fuzzy system is evaluated with the wine data set. The data is normalized to the unit interval [0, 1]. A crossover rate of 0.9 and a mutation rate of 0.1 are specified. The population size is fixed to 40 chromosomes. The GA is allowed to run for a maximum of 100 generations. However, the GA is terminated on reaching 100% classification rate. The number of newly created chromosomes in each generation is limited to 90% of the population. In each generation, a new population is formed based on fitness value from a combined parent and child pool. Fitness is considered as the sum of the number of misclassifications and the number of fuzzy rules. The smaller the fitness value, better is the performance of the fuzzy classifier system.

To compare with the previously published results, all the 178 patterns are used as training patterns. The experiment is conducted for 10 runs. The number of generations needed to reach the specified number of misclassifications within 100 generations is determined. The experiment is performed thrice, once each with 3 fuzzy sets, 4 fuzzy sets and 5 fuzzy sets per attribute. The results are provided in

Tables 4.7 - 4.9. The right-most column of each table provides the average number of generations and the average number of rules required, to reach the number of misclassifications listed in the left-most column.

The results with five fuzzy sets per attribute (Table 4.9) is considered for comparison with the published results. The results obtained are as follows:

- **Best classification rate**:
    1. 100% with 4 rules in 50 generations / 40 chromosomes in each population
    2. 100% with 3 rules in 77 generations / 40 chromosomes in each population
- **Average classification rate**: 99.7% with 6 rules in 55 generations / 40 chromosomes in each population
- **Worst classification rate**: 99.5% with 9 rules in 81 generations / 40 chromosomes in each population

While the test conditions are not exactly the same as that of [90] and [60], it is observed that the proposed system outperformed the approach in [90] and the base Michigan method in [60]. While the average classification rate of 99.7% is slightly inferior to the 100% average classification rate of the learning Grade of Certainty method proposed by Ishibuchi *et al.* [60], the number of rules required as well as the generations required are much lower than the Ishibuchi learning method.

While [90] and [60] retain the same number of fuzzy rules from the start of the GA till its end (*i.e 60*), the proposed system started with a maximum of 10 fuzzy if-then rules and reduced it to a minimum of 3 rules in the best performance case, resulting in a very compact rule base. The 100% classification rate achieved with only 3 fuzzy rules is superior to the 100% classification rate achieved with 5 rules, as reported by Ishibuchi *et al.* [91] using a multi-objective genetic algorithm. Four of the ten runs conducted resulted in 100% classification rate, which indicates a high convergence rate for the proposed system.

The use of "*don't care*" value for attributes, simplifies the rules and makes the rules more comprehensive by eliminating unnecessary attributes from the rule set. This is especially effective for data sets with large number of attributes such as the wine data set. The 3 rules obtained for best performance with the five fuzzy set case ({*tiny, small, medium, large, very large*} *and the "don't care" value*) are provided below. The attributes with the "*don't care*" value are removed from the rules

1. If attrib1 is large (Sigma=0.022917, Mean=0.66667),
   and attrib2 is medium (Sigma=0.030208, Mean=0.50417),
   and attrib3 is medium (Sigma=0.032292, Mean=0.50417),
   and attrib4 is tiny (Sigma=0.026042, Mean=0.17917),
   and attrib6 is very large (Sigma=0.032292, Mean=0.8375),
   and attrib7 is very large (Sigma=0.03125, Mean=0.875),
   and attrib9 is medium (Sigma=0.040625, Mean=0.49583),
   and attrib10 is small (Sigma=0.03125, Mean=0.31667),
   and attrib12 is large (Sigma=0.030208, Mean=0.69583),
   and attrib13 is very large (Sigma=0.032292, Mean=0.80417),
   then classification is 1 with CF = 1

2. If attrib1 is tiny (Sigma=0.03125, Mean=0.16667),
   and attrib2 is small (Sigma=0.0375, Mean=0.33333),
   and attrib3 is tiny (Sigma=0.029167, Mean=0.15833),
   and attrib5 is medium (Sigma=0.036458, Mean=0.50417),
   and attrib7 is large (Sigma=0.036458, Mean=0.6625),

and attrib10 is small (Sigma=0.03125, Mean=0.31667),
and attrib11 is very large (Sigma=0.036458, Mean=0.84583),
and attrib12 is very large (Sigma=0.033333, Mean=0.80833),
and attrib13 is tiny (Sigma=0.034375, Mean=0.17083),
then classification is 2 with CF = 1

3. If attrib1 is very large (Sigma=0.033333, Mean=0.8),
and attrib2 is tiny (Sigma=0.035417, Mean=0.14167),
and attrib4 is very large (Sigma=0.029167, Mean=0.80833),
and attrib5 is medium (Sigma=0.036458, Mean=0.50417),
and attrib6 is medium (Sigma=0.035417, Mean=0.525),
and attrib7 is tiny (Sigma=0.035417, Mean=0.15833),
and attrib8 is medium (Sigma=0.03125, Mean=0.45833),
and attrib10 is very large (Sigma=0.028125, Mean=0.8625),
and attrib12 is small (Sigma=0.025, Mean=0.35),
and attrib13 is tiny (Sigma=0.034375, Mean=0.17083),
then classification is 3 with CF = 1

It is observed from the average number of generations that as the number of fuzzy sets increases, the convergence requires lesser number of generations. However, the lesser number of fuzzy sets reduces the complexity of the fuzzy system with no appreciable increase in the number of misclassifications. Even with the three fuzzy set case, the proposed system is able to achieve 100% classification in three of the ten runs.

Table 4.7: Three fuzzy sets per attribute for Wine Data Set

Misclassified	**Generations to reach specified misclassifications / Number of fuzzy rules needed**										**Average**
	Run1	**Run2**	**Run3**	**Run4**	**Run5**	**Run6**	**Run7**	**Run8**	**Run9**	**Run10**	
9	11 / 7	/	/	/	/	/	18 / 5	27 / 8	/	20 / 5	19 / 6.25
8	14 / 9	14 / 5	/	/	16 / 6	13 / 5	/	36 / 9	20 / 5	/	18.8 / 6.5
7	22 / 7	/	/	/	/	26 / 5	22 / 5	39 / 8	/	21 / 3	26 / 5.6
6	/	/	23 / 8	/	/	/	/	/	/	/	23 / 8
5	/	21 / 6	31 / 8	16 / 7	32 / 6	28 / 5	25 / 5	47 / 8	24 / 5	/	28 / 6.25
4	25 / 7	34 / 5	35 / 7	24 / 8	40 / 6	35 / 6	32 / 5	50 / 9	/	24 / 4	33.2 / 6.3
3	31 / 7	40 / 5	/	29 / 8	45 / 5	58 / 5	/	79 / 4	25 / 4	30 / 4	42.1 / 5.25
2	37 / 4	57 / 5	53 / 7	50 / 7	46 / 6	74 / 5	36 / 5	94 / 4	43 / 3	/	54.4 / 5.1
1	63 / 4	66 / 5	93 / 7	/	66 / 5	85 / 5	/	97 / 5	48 / 3	32 / 1	68.75 / 4.4
0	90 / 4	/	97 / 7	/	/	/	41 / 5	/	/	/	76 / 5.3

"/" refers to the case where the specified number of misclassifications did not occur within maximum generations

Eg: 22/7 in the third row of Run1 refers to 7 misclassifications in the 22nd generation with seven fuzzy rules

Average number of generations is calculated only for those misclassifications found within maximum generations

Table 4.8: Four fuzzy sets per attribute for Wine Data Set

Misclassified	Generations to reach specified misclassifications / Number of fuzzy rules needed										Average
	Run1	**Run2**	**Run3**	**Run4**	**Run5**	**Run6**	**Run7**	**Run8**	**Run9**	**Run10**	
9	23 / 5	22 / 8	/	/	/	5 / 10	9 / 9	/	/	17 / 9	15.2 / 8.2
8	/	27 / 8	/	11 / 8	/	10 / 9	/	/	/	/	16 / 8.3
7	24 / 5	29 / 8	/	16 / 4	17 / 8	/	14 / 9	18 / 10	/	/	19.7 / 7.3
6	/	/	21 / 9	17 / 4	/	/	/	/	21 / 6	22 / 9	20.25 / 7
5	29 / 5	/	29 / 8	22 / 4	25 / 6	12 / 10	/	29 / 7	25 / 5	25 / 9	24.5 / 6.8
4	31 / 4	32 / 8	37 / 9	26 / 4	/	/	15 / 9	/	27 / 6	30 / 9	28.3 / 7
3	39 / 5	37 / 8	41 / 8	33 / 4	29 / 7	/	29 / 9	37 / 7	/	38 / 9	35.4 / 7.1
2	40 / 4	/	/	/	40 / 6	22 / 10	53 / 9	57 / 7	44	58 / 9	44.9 / 7.3
1	81 / 3	49 / 8	42 / 8	41 / 3	69 / 6	48 / 9	59 / 9	/	89 / 5	71 / 9	61 / 6.7
0	/	/	/	50 / 3	/	/	99 / 9	/	/	/	74.5 / 6

"/" refers to the case where the specified number of misclassifications did not occur within maximum generations

Eg: 24/5 in the third row of Run1 refers to 7 misclassifications in the 24th generation with five fuzzy rules

Average number of generations is calculated only for those misclassifications found within maximum generations

Table 4.9: Five fuzzy sets per attribute for Wine Data Set

Misclassified	Generations to reach specified misclassifications / Number of fuzzy rules needed										Average
	Run1	Run2	Run3	Run4	Run5	Run6	Run7	Run8	Run9	Run10	
9	/	6 / 10	/	19 / 8	20 / 5	13 / 9	/	19 / 9	11 / 9	/	14.7 / 8.3
8	/	18 / 6	/	21 / 8	/	15 / 9	/	26 / 7	12 / 7	19 / 9	18.5 / 7.7
7	22 / 9	19 / 6	14 / 7	22 / 8	/	/	10 / 5	/	/	30 / 9	19.5 / 7.3
6	29 / 9	22 / 6	15 / 8	24 / 8	21 / 5	/	14 / 4	/	18 / 10	/	20.4 / 7.1
5	35 / 9	23 / 5	24 / 7	/	24 / 5	17 / 9	17 / 5	28 / 7	21 / 10	34 / 8	24.8 / 7.2
4	/	25 / 6	27 / 7	25 / 8	28 / 5	/	19 / 4	32 / 7	32 / 7	/	26.9 / 6.3
3	/	27 / 5	29 / 6	/	34 / 5	20 / 9	26 / 4	34 / 8	39 / 7	39 / 9	31 / 6.6
2	44 / 9	/	30 / 5	34 / 8	/	26 / 8	34 / 4	40 / 7	53 / 6	47 / 9	38.5 / 7
1	81 / 9	29 / 5	40 / 5	42 / 8	41 / 4	34 / 9	97 / 3	52 / 7	74 / 5	62 / 9	55.2 / 6.4
0	/	50 / 4	71 / 5	56 / 8	77 / 3	/	/	/	/	/	63.5 / 5

"/" refers to the case where the specified number of misclassifications did not occur within maximum generations

Eg: 22/9 in the third row of Run1 refers to 7 misclassifications in the 22nd generation with nine fuzzy rules

Average number of generations is calculated only for those misclassifications found within maximum generations

4.5.3 Performance Evaluation: Glass Data Set

The Glass data set is available from University Of California, Irvine database. It consists of 214 samples with 9 continuous attributes from 6 classes. In [92] Holte analyzed the performance of the 1R algorithm and Quinlan's C4 algorithm [93] on sixteen data sets including the glass data set. Holte used a random sub-sampling technique to evaluate performance of the algorithms on test data. Two thirds of the given samples are used as training samples and the remaining one third are used as test data. The data is randomly split and iterated 25 times. The following results are reported in [92]

- Classification rate for test data by 1R: 53.8%
- Classification rate for test data by C4: 63.2%

Ishibuchi *et al.* [60] applied the glass data set with the same random sub-sampling technique, to the Michigan approach based fuzzy classifier system. The data is normalized to the unit interval [0, 1]. Each fuzzy if-then rule is encoded in a single chromosome. The trial is conducted for 1000 generations with 100 chromosomes in each population. Ishibuchi reported the following results:

- Classification rate for test data: 64.4% with 100 rules for 1000 generations
- Error rate for test data: 35.6% with 100 rules for 1000 generations

The proposed system is evaluated with the glass data set. The data is normalized to the unit interval [0, 1]. A crossover rate of 0.9 and a mutation rate of 0.1

are specified. The population size is fixed to 40 chromosomes. The GA is allowed to run for a maximum of 100 generations. The number of replaced chromosomes in each rule is limited to 10% of the population. Each attribute is split into 3 fuzzy sets and can acquire a *"don't care"* value during GA iterations. Fitness is evaluated in the same way as it is done for Iris and wine data sets.

To compare with the previous published results, the same random sub-sampling technique is used for splitting the samples into 2/3rd training data and 1/3rd test data. The experiment is conducted for 10 trials. In each trial, the samples are randomly split into training and test sets. The same training and test sets are applied to the system for 5 runs, with each run consisting of 100 generations. This is done to average the performance of the system for the pair of training and test sets. The results are provided in Table 4.10.

Table 4.10: 3 fuzzy sets per attribute for Glass Data Set

Trial	**Classification rate**	**Error rate**	**Number of rules**
1	64.8%	35.2%	12
2	64.8%	35.2%	14
3	64.8%	35.2%	15
4	73.2%	26.8%	15
5	73.2%	26.8%	13
6	69.0%	31.0%	12
7	64.8%	35.2%	13
8	67.6%	32.4%	12
9	64.8%	35.2%	12
10	73.2%	26.8%	13
Average	68%	34%	13

By comparing these results with those obtained for 1R, C4 algorithms and the Michigan approach, it is observed that the average performance of the proposed fuzzy system is superior to the other systems. In three of the ten trials, the system achieved a high performance of 73.2% classification rate which is better compared with most of the other classification methods.

4.6 Further Improvements

The effect of training membership function types (rather than using only Gaussian function) as well as utilizing a fuzzy system to adapt crossover and mutation rates has to be explored.

Most of the techniques including the proposed classification system, keep the number of fuzzy sets per attribute fixed throughout the GA run. The system can be further improved to evolve the number of fuzzy sets per attribute, during the GA run with an aim to obtain fewer fuzzy sets and lesser complexity of the fuzzy if-then rule set.

The proposed system can be improved further by having a larger mutation probability biased towards using *"don't care"* value for attributes to reduce the dimensionality of the system further.

Currently, the initial population of the GA system is randomly generated. The proposed system has improved performance by assigning the consequent part of the rule, based on effectiveness of the antecedent part in classification of a particular class. The performance of the system can be improved further by selecting the initial population, based on compatibility with training patterns rather than randomly generating it.

4.7 Conclusion

In this chapter, a GA based fuzzy system is proposed, in which the membership function shapes and the fuzzy rule set, along with the number of rules in the rule set are evolved using a Pittsburgh approach based Genetic Algorithm. While the system is capable of evolving different membership function types, only the Gaussian membership functions are used for simplicity. The fuzzy system makes use of "don't care" values for attributes to remove unwanted attributes from the fuzzy rule set.

By computer simulations on real world test problems, the performance of the system is examined. The simulation results demonstrate that the proposed fuzzy classifier outperformed many other classification methods. It is concluded that the proposed classification system is useful for a wide range of classification problems. Future directions for improving performance of the proposed system is also discussed.

Chapter 5

Boosting based Fuzzy-Rough Set Pattern Classifier

5.1 Overview

This chapter analyzes an algorithm that automatically evolves low dimensionality fuzzy rules and corresponding membership functions for pattern classification, based on the rough set concepts of fuzzy lower and upper approximations. The proposed method transforms each quantitative value into a fuzzy set of linguistic terms using membership functions and then calculates the fuzzy lower and upper approximations. The membership functions are derived from cluster points generated by subtractive clustering technique. A certain rule set based on the fuzzy lower approximation and a possible rule set based on fuzzy upper approximation are generated. A genetic algorithm based on iterative rule learning is employed in combination with a boosting technique, for generating the possible rule set incrementally by optimizing one fuzzy classifier rule at a time. The proposed fuzzy if-then rules include a plausibility factor. The effect of the plausibility factor on the learning of rules is studied. The validity of the proposed technique is tested on some well known data sets from the UCI repository.

5.2 Introduction

Machine learning from examples has been a prime area of research in recent years. In this approach, learning is based on similarities between positive examples representing a similar concept and dissimilarities between positive and negative examples representing differing concepts. Knowledge, in the form of rules is induced by learning from training examples. One of the main problems faced in this approach is the uncertainty in the data. There are two main reasons for uncertainty: incomplete evidence or conflicting evidence [94].

Z.Pawlak introduced rough set theory in the early eighties, as a new tool to deal with uncertainty [3, 95, 4]. One of the advantages of rough set theory is that it does not need additional knowledge about data such as prior probability in probabilistic approach. Rough set theory is especially well suited for handling inconsistencies [17]. It deals with inconsistencies by computing lower and upper approximations of the concept under consideration. Based on these approximations, certain and possible rules are induced. Many applications and extensions of the rough set theory have been proposed, such as reasoning with incomplete information [96], knowledge-base reduction [97], extensions for data mining [98], probabilistic rule discovery [99] and variable precision extension [100]. Most of these previous studies focus on the handling of discrete or binary valued data. However, real world applications often involve continuous, quantitative data which presents a challenge in designing sophisticated learning systems capable of handling different types of data.

One way to handle continuous data is by partitioning the data into crisp or discrete intervals. This process of discretization determines how coarsely the data is split into intervals. Various approaches have been proposed in the literature for discretizing data in the context of rough set theory [101, 102, 103]. The crisp discretization is achieved by generating a set of 'cuts' of attributes within the dynamic ranges of the corresponding attributes. The positions of cuts are very sensitive to the subsets of the information system, which are used to generate the cuts, as well as to the methodology adopted. The position sensitivity of cuts may adversely affect the classification accuracy. Moreover, a very large number of if-then rules have to be derived from the crisp intervals, to cover the whole attribute space and to make the classification system generic. Fuzzy discretization offers solutions for tackling difficulties associated with continuous attributes.

Fuzzy set concepts are often used to represent continuous quantitative data expressed in linguistic terms and membership functions because of its simplicity and similarity to human reasoning [104]. Fuzzy inference systems have been successfully applied in fields such as automatic control, data classification, decision analysis, expert systems, manufacturing, and computer vision [105]. Being multidisciplinary nature, fuzzy inference systems are associated with a number of names, such as fuzzy-rule-based systems, fuzzy expert systems, fuzzy modelling, fuzzy associative memory, fuzzy logic controllers, and simply fuzzy systems.

Fuzzy set theory and rough set theory are considered complementary in that they both deal with models of uncertainty: vagueness for fuzzy sets and indiscernibility for rough sets [8]. Combining the two theories provides the concepts of lower and upper approximations of fuzzy sets by similarity relations. This chapter proposes to partition data by fuzzy-discretization through fuzzy membership functions. The fuzzy lower and upper approximations are calculated for the training data set and fuzzy if-then rules are derived from these approximations. The rest of this chapter is organized as follows. In Section 5.3, the integration of rough sets and fuzzy sets is briefly reviewed. Section 5.4 highlights approaches in the existing literature which are similar to the proposed technique, for solving pattern classification problems. Section 5.5 describes the proposed technique by which fuzzy if-then rules are derived, based on the fuzzy lower and upper approximations. In Section 5.6, the plausibility factor is compared with the Certainty Grade from the literature. The proposed algorithm is tested on some well known data sets and the results are discussed in Section 5.7. Conclusions are provided in Section 5.8.

5.3 Integration of rough and fuzzy sets

5.3.1 Rough Set Theory

A rough set is an approximation of a vague concept by a pair of precise concepts, called lower and upper approximations [4]. Objects belonging to the same category characterized by the same attributes are not distinguishable and are said to be indiscernible. Let $I = (U, A)$ be an information system, where U is the universe of discourse and A is a non-empty finite set of features such that $a : U \rightarrow V_a, \forall a \in A, V_a$ being the value set of feature a. In many applications, the outcome of classification is known and represented by a special attribute set termed 'decision attribute' set. Such information systems are known as 'decision systems'. Consider a decision system, $A = C \cup D$ where C is a set of conditional attributes and D is a set of decision attributes. With any $P \subseteq A$ there associated is an equivalence relation $IND(P)$:

$$IND(P) = (x, y) \in U^2 | \forall a \in P, a(x) = a(y). \tag{5.1}$$

If $(x, y) \in IND(P)$, then x and y are indiscernible by features from P. The equivalence classes of the P-indiscernibility relation are denoted by $[x]_P$. For any $X \subseteq U$, X can be approximated using only the information contained in P by constructing the *P-lower* and *P-upper approximations* of X, denoted by $\underline{P}X$ and

$\overline{P}X$ respectively,

$$\underline{P}X = x|[x]_P \subseteq X \tag{5.2}$$

$$\overline{P}X = x|[x]_P \cap X \neq \phi \tag{5.3}$$

If P and Q are equivalence relations over U, then the positive region $POS_P(Q)$ is defined as,

$$POS_P(Q) = \bigcup_{x \in U/Q} \underline{P}X. \tag{5.4}$$

For pattern classification purposes, the positive region contains all the objects of U that can be classified into the classes of U/Q using the knowledge in attribute set P.

5.3.2 Fuzzy sets and Equivalence classes

Fuzzy set theory was first proposed by Zadeh in 1963 [5]. A Fuzzy set F is characterized by a membership function $\mu_F(x)$ which defines degrees of set membership, usually over the range $[0, 1]$. The concept of equivalence classes that form the basis for rough set theory, can be extended to fuzzy set theory to form fuzzy equivalence classes [8].

The concept of crisp equivalence classes can be extended by the inclusion of a fuzzy similarity relation S on the universe, which determines the extent by which two elements are similar in S. Using the fuzzy similarity relation, the fuzzy equivalence class $[x]_S$ for objects close to x can be defined as $\mu_{[x]_S}(y) = \mu_S(x, y)$. This definition degenerates to the normal definition of equivalence classes when S is non-fuzzy. The family of normal fuzzy sets produced by a fuzzy partitioning of the universe of discourse, can play the role of fuzzy equivalence classes [8]. The fuzzy *P-lower* and *P-upper approximations* are defined as [8]:

$$\begin{aligned}\mu_{\underline{P}X}(F_i) &= \inf_x \max\{1-\mu_{F_i}(x), \mu_X(x)\}\ \forall i, \\ \mu_{\overline{P}X}(F_i) &= \sup_x \min\{\mu_{F_i}(x), \mu_X(x)\}\ \forall i,\end{aligned} \tag{5.5}$$

where F_i denotes a fuzzy equivalence class belonging to U/P. The crisp positive region in traditional rough set theory is defined as the union of the lower approximations. By the extension principle, the membership of an object $x \in U$, belonging to the fuzzy positive region is defined by Eqn. 5.6.

$$\mu_{POS_P(Q)}(x) = \sup_{X \in U/Q} \mu_{\underline{P}X}(x). \tag{5.6}$$

Hong *et al.* [106, 107] provides another definition for the fuzzy lower and upper approximations. For an arbitrary subset X of the universe U and an arbitrary subset B of the attribute set A, the fuzzy lower approximation $B_*(X)$ and fuzzy upper approximation $B^*(X)$ for B on X are defined by Eqn. 5.7.

$$\begin{aligned}B_*(X) &= \{(B_k(x), \mu_{B_k}(x)) | x \in U, B_k(x) \subseteq X, 1 \leq k \leq |B(x)|\}, \\ B^*(X) &= \{(B_k(x), \mu_{B_k}(x)) | x \in U, B(x) \cap X \neq \emptyset, 1 \leq k \leq |B(x)|\}.\end{aligned} \tag{5.7}$$

where $B_k(x)$ is the k^{th} partition in the set of partitions U/B induced by the fuzzy equivalence relation based on the attribute subset B. $\mu_{B_k}(x)$ is the minimum of all the membership values attained by the instances in the k^{th} partition and is defined as the membership value of the k^{th} partition.

Hong *et al.* [106, 107] also define a plausibility factor for each partition $B_k(x)$, given by Eqn. 5.8.

$$p(B_k(x)) = \frac{\sum\limits_{x \in (B_k(x) \cap X_l)} \mu_{B_k}(x)}{\sum\limits_{x \in B_k(x)} \mu_{B_k}(x)} \tag{5.8}$$

where X_l refers to the set of all instances belonging to the class C_l. As seen from Eqn. 5.8, all the instances in the fuzzy lower approximation $B_*(X)$ are classified with a plausibility of 1, while the instances in the fuzzy upper approximation $B^*(X)$ are classified only with a partial certainty given by the plausibility factor.

Fuzzy equivalence classes and fuzzy lower and upper approximations have been successfully applied for classification purposes in the literature. Hong *et al.* [106, 107] successfully applied fuzzy-rough hybridization for data-mining and expert systems by generating maximally general fuzzy rules. Shen *et al.* [108] integrated a fuzzy inference system with the 'QuickReduct' algorithm, for generating fuzzy rules for pattern classification. Roy *et al.* [109] proposed a fuzzy discretization technique for improving the performance of rough set theoretic classifier.

Most of these techniques make use of pre-defined fuzzy membership functions in the classifier development system. The current work proposes an algorithm that automatically evolves low dimensionality fuzzy rules and corresponding membership functions for pattern classification, directly from the training data set based on fuzzy lower and upper approximations.

5.3.3 Fuzzy System Design By Genetic Algorithm

Genetic Algorithms (GAs) are random search algorithms inspired by natural genetics to evolve solutions to problems [53, 54]. The basic idea is to maintain a population of chromosomes, which represents candidate solutions to the problem. The population evolves over time through a process of competition and controlled variation. Each chromosome in the population has an associated fitness. Based on the fitness values, new chromosomes are generated using genetic operators such as *crossover* and *mutation*. The outline of a basic GA is as follows:

1. Create an initial population;

2. Evaluate fitness of each chromosome in population;
3. Based on fitness, select chromosomes for reproduction;
4. Apply genetic operations, *crossover* and *mutation* on selected chromosomes, to form new chromosomes;
5. Replace a part of the current population with newly generated chromosomes; and,
6. Terminate GA if stopping condition is satisfied, else return to step 2.

GAs have long been associated with fuzzy inference systems for generating fuzzy rules and training membership functions [2, 59, 61]. Three different approaches have been proposed for applying GA to learning processes, the Michigan, the Pittsburgh and the Iterative Rule Learning (IRL) approaches. In Michigan approach, the chromosomes correspond to classifier rules which are evolved as a whole, whereas in the Pittsburgh approach each chromosome encodes a complete set of classifiers. In the IRL approach, each chromosome represents only one rule, but contrary to the Michigan approach, only the best individual is considered as solution while discarding the rest of the chromosomes or rules. Since a single rule provides only a partial solution, the GA is placed in an iterative scheme for generating a set of rules. In the iterative scheme, either the selected rules are penalized or the classified samples are penalized, by way of reduced weights, to ensure that the search for new rules is focussed on the unclassified samples.

A major problem in the Michigan approach is related to resolving the conflict between the interests of individual rules and the collective classifier. This problem does not arise in the Pittsburgh approach since competition occurs between complete rule sets rather than among individual rules. However, the Pittsburgh approach is much more computationally intensive due to the maintenance and evaluation of complete rule sets. The advantage of the IRL approach over the other two approaches is that, it is computationally less intensive since it looks for only one rule in each sequence of iteration. At the same time a cohesive classifier is generated through the penalizing mechanism, thereby avoiding conflict between individual rule and the whole classifier. The genetic algorithm employed in the proposed method follows the IRL approach [67, 70].

5.4 Existing approaches

Traditionally, the rules in a fuzzy inference system are generated from expert knowledge. If no expert knowledge is available, the usual approach is to identify and train fuzzy membership functions so as to match data clusters in the training set. Ishibushi *et al.* [60, 110] proved that it is possible to build effective classifiers by making use of uniformly distributed membership functions irrespective of data clusters, by training rule-weights. Nauck *et al.* [111] showed that it is not necessary to use rule weights in fuzzy rule-based systems, as the learning of rule weights can be equivalently replaced by the modification of antecedent or consequent membership functions. Hence, a high performance fuzzy classifier can be evolved either by modifying membership functions or rule weights. However, if the data clusters are known beforehand, it is easier to design membership functions to suit them and derive rule-weights from the training data itself, rather than modifying membership functions or rule weights, to improve classifier performance. This can be achieved by a suitable clustering technique. One such technique is the subtractive clustering technique [112] which is a fast single pass algorithm for determining data clusters.

The subtractive clustering technique is attractive as it does not require a predetermination of the number of clusters. The subtractive clustering method assumes that each data point is a potential cluster center and calculates a measure of the potential for each data point based on the density of the surrounding data points. The algorithm selects the data point with the highest potential as the first cluster center and then reduces the potential of data points near the first cluster center. The algorithm then selects the data point with the highest remaining potential. This process of acquiring a new cluster center and reducing the potential of surrounding data points is repeated until the potentials of remaining data points fall below a threshold. The subtractive clustering method is an extension of Yager's mountain clustering method [113].

Generally, it is much easier to generate a lot of inaccurate or 'weak' rules that apply only to certain instances in a training data set, than generating a few rules which are accurate or 'strong' rules, which classify almost all instances in the data set. Boosting [114, 115] is a technique which aggregates multiple 'weak' rules or hypotheses invoked over different distributions of the training data into a single composite classifier [116]. Schapire [117] came up with the first provable polynomial-time boosting algorithm in 1989. The AdaBoost algorithm which solved many of the practical difficulties of the earlier boosting algorithms, was introduced in 1995 by Freund and Schapire [118].

The AdaBoost algorithm has undergone intense theoretical study and empirical testing and various modifications has been proposed [119, 71]. It has been proven to be successful in building good classifier systems. Hoffman [71] proposed a modification to the AdaBoost algorithm which employs a GA iteratively, for generating individual fuzzy rules. This boosting scheme reduces the weights of training samples which are correctly classified by the newly generated rule. Hoffmann [71] and Gonzalez *et al.* [120] proposed a number of fitness criteria such as covering degree, frequency and penalization of negative samples.

In order to implement a GA based on IRL, three criteria are essential [67]. A criteria for selecting the best rule in each iteration, a criteria for penalizing classified samples and a termination criteria. Fitness functions are usually defined in the GA to satisfy the first criteria. The classified samples can be removed as in SLAVE [70] or the training samples can be assigned with reduced weights [71] to satisfy the second criteria. Finally, the iteration can be terminated after all training instances are classified.

The main idea behind boosting is to repeatedly apply a weak learning algorithm on various distributions of training data and finally aggregate the individual classifiers into a single classifier [71]. After each iteration, the distribution of training instances are changed by updating their weights based on the current classifier error.

5.4.1 Grade of Certainty

Ishibuchi *et al.* [60, 110] employed a heuristic rule weight termed as Certainty Grade, CF and defined fuzzy rules of the type given in Eqn. 5.9.

$$\begin{aligned} &\text{Rule } R_j\text{: If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn} \\ &\quad \text{then Class } C_j \text{ with } CF = CF_j, \\ &\quad for\ j = 1, 2, \ldots, N, \end{aligned} \tag{5.9}$$

where,

R_j	Label of the jth fuzzy if-then rule
A_{j1},...,A_{jn}	Antecedent fuzzy sets
C_j	Consequent class
CF_j	Certainty grade of rule R_j
N	Number of rules.

The heuristic certainty grade CF_j is computed as in Eqn. 5.10.

$$CF_j = \frac{\beta_{Class\ C_j}(R_j) - \overline{\beta}}{\sum\limits_{k=1}^{c} \beta_{Class\ k}(R_j)}, \tag{5.10}$$

where,

$$\overline{\beta} = \frac{\sum\limits_{k \neq C_j} \beta_{Class\ k}(R_j)}{(c-1)}, \tag{5.11}$$

and,

$$\beta_{Class\ k}(R_j) = \sum_{x_p \in Class\ k} \mu_j(x_p). \tag{5.12}$$

The effect of certainty grade on the fuzzy rule as a replacement for training of membership functions has been studied extensively in [110].

5.5 Proposed approach

In this section, a new algorithm based on fuzzy lower and upper approximations is proposed. The aim is to come up with a technique which is intuitive and computationally less intensive while automatically evolving a classifier from the training set, based on a combination of fuzzy and rough set concepts, without compromising on the classification ability.

Subtractive clustering is performed as a pre-processing step for obtaining the initial membership functions. This provides a good starting point for the membership functions since the subtractive clustering technique provides the data cluster points where the data is most concentrated.

Fuzzy and rough sets aim at different purposes in modelling uncertainty, namely vagueness for fuzzy sets and coarseness for rough sets. Fuzzy lower and upper approximations of a fuzzy set are defined, when the universe of discourse is coarsened by means of an equivalence relation. Usage of the fuzzy lower and upper approximations provides a more accurate account of imperfect information.

In the proposed classifier, the fuzzy lower and upper approximations are obtained from the initial membership functions generated from subtractive clustering. A set of certain rules is derived from the fuzzy lower approximation. Further, a boosting technique is proposed to obtain a set of possible rules from the fuzzy upper approximation, for classifying the misclassified and unclassified samples. A new fitness measure is proposed to evaluate the possible fuzzy rules, based on four objectives. The fuzzy possible rule set utilizes a modifier known as Plausibility factor (PF). The effect of this modifier in fuzzy classification is compared to another similar modifier, the Grade of certainty (CF).

The technique proposed by Hong *et al.* [106, 107] utilizes the fuzzy upper and lower approximations for generating maximally general fuzzy rules. However, the fuzzy membership functions are not derived from the training samples and are pre-defined, which necessitates some external means for generating the membership functions.

The classifier proposed by Hoffman [71] obtains fuzzy rules from a boosting technique based on IRL approach. However, this classifier does not rely on fuzzy lower and upper approximations and modifiers such as plausibility factor. Hence, the classifier does not differentiate between certain rules and possible rules.

The proposed technique overcomes these limitations by generating certain and possible fuzzy rules directly from the training set. Also, the plausibility factor helps in tuning the fuzzy rules without modification of membership functions. This provides learning capabilities for the system, which can rely on membership functions generated from data clusters, rather than identify and tune membership functions from scratch.

5.5.1 Fuzzy if-then rules for classification

The proposed classifier utilizes fuzzy rules of the type given in Eqn. 5.13,

$$\begin{aligned}\text{Rule } R_j:\ & \text{If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn} \\ & \text{then } Y = c_j \text{ with plausibility } PF_j, \qquad j = 1, 2, \ldots, m \end{aligned} \tag{5.13}$$

where,

R_j	Label of the jth fuzzy if-then rule
A_{j1},...,A_{jn}	Antecedent fuzzy sets
c_j	Consequent class
PF_j	Plausibility Factor

The plausibility factor PF_j is defined as the ratio of membership attained by the instances belonging to consequent class c_j, with the membership attained by all instances in the fuzzy upper approximation of the partition formed by rule R_j, as in Eqn. 5.14. This definition of plausibility factor is similar to Eqn. 5.8. It is observed that the plausibility factor attains the maximum value of 1 for the fuzzy lower approximation.

$$PF_j = \frac{\sum\limits_{k|c^k=c_j} \mu_{R_j}(x_k)}{\sum\limits_{k} \mu_{R_j}(x_k)}, \text{ where } k \text{ is the cardinality of training data} \tag{5.14}$$

In a fuzzy inference system, if a rule is activated, i.e if any of the attributes attain a membership value greater than zero in the rule, the degree of support for the rule is obtained by applying a fuzzy operator such as *min* or *prod* to the fuzzy membership functions from the antecedent attributes as in Eqn. 5.15.

$$\mu_{R_j}(x^k) = \mu_{R_j}(\{x_1^k, \ldots, x_n^k\}) = \min_{n=1}^{N} \mu_{A_{jn}}(x_n^k) \tag{5.15}$$

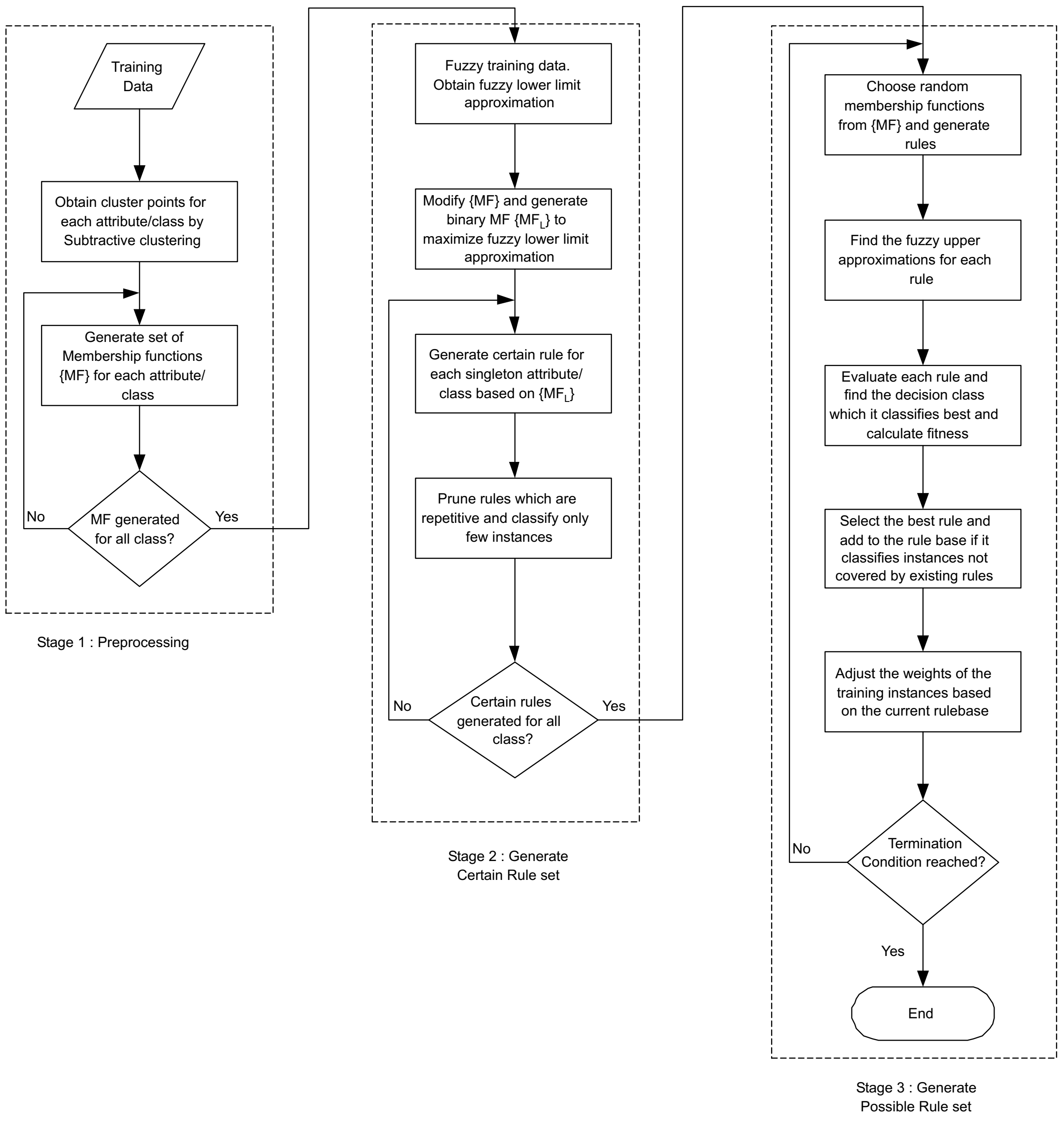
Each possible classification c_j accumulates the degree of activation of fuzzy rules R_j weighted by the plausibility factor PF_j, with a matching consequent $c_j = C_M$. The instance x^k is classified according to the class label $C_{max}(x^k)$ by Eqn. 5.16

$$C_{max}(x^k) = \text{argmax}_{C_m} \sum_{R_j/c_j=C_m} PF_j \, \mu_{R_j}(x^k) \tag{5.16}$$

Generally, fuzzy rules are classified into two categories: approximate rules and descriptive rules. In the case of approximate rules, each rule has its own definition of fuzzy linguistic terms, whereas in descriptive rules, the fuzzy linguistic terms refer to a commonly defined set of membership functions [49]. The two types of fuzzy rules involve a trade-off between accuracy and comprehensibility. Usually, the approximate rules tend to be more accurate while being less comprehensible as compared to descriptive rules due to the difficulty in linguistic interpretation. This proposed approach employs the more comprehensible descriptive rules, for classification. Experimental studies indicate that the classifier is able to achieve classification accuracy, comparable to well known classifier techniques in the literature.

5.5.2 Learning fuzzy if-then rules for classification

In the proposed method, subtractive clustering is performed as a preprocessing step to obtain cluster points. A set of membership functions are generated from these cluster points. The training data are fuzzified with the generated membership functions and the fuzzy lower and upper approximations are obtained. A set of certain rules are generated from the fuzzy lower approximations. Another set of possible rules are generated from the fuzzy upper approximations by a boosting enhanced genetic algorithm, for those training instances which are not classified by the certain rule set. Each of these steps is considered as a stage as indicated in Fig. 5.1. These steps are further explained in the following sections.



Training Data
Obtain cluster points for each attribute/class by Subtractive clustering
Generate set of Membership functions {MF} for each attribute/ class
MF generated for all class?
No
Yes
Stage 1 : Preprocessing
Fuzzy training data. Obtain fuzzy lower limit approximation
Modify {MF} and generate binary MF {MF$_L$} to maximize fuzzy lower limit approximation
Generate certain rule for each singleton attribute/ class based on {MF$_L$}
Prune rules which are repetitive and classify only few instances
Certain rules generated for all class?
No
Yes
Stage 2 : Generate Certain Rule set
Choose random membership functions from {MF} and generate rules
Find the fuzzy upper approximations for each rule
Evaluate each rule and find the decision class which it classifies best and calculate fitness
Select the best rule and add to the rule base if it classifies instances not covered by existing rules
Adjust the weights of the training instances based on the current rulebase
Termination Condition reached?
No
Yes
End
Stage 3 : Generate Possible Rule set

Figure 5.1: Sequence of steps involved in generating certain and possible rule sets

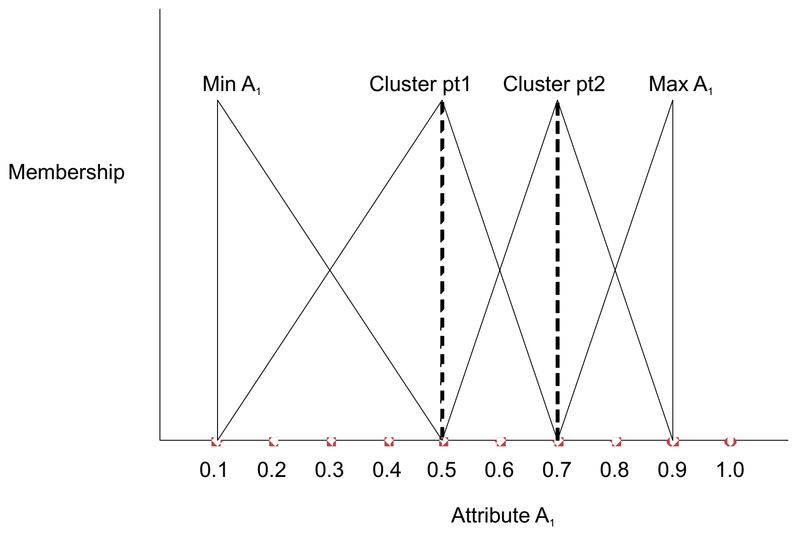


Figure 5.2: Obtaining membership functions from cluster points

5.5.3 Stage1: Preprocessing step - Membership functions from cluster points

In the proposed technique, the data clusters are determined from the training data set by the subtractive clustering method for each consequent class and each attribute, as a pre-processing step. Once the data clusters are identified for each consequent class and attribute, membership functions are formed with the data clusters as center points. For example, if a set of cluster points $cp_1 = \{0.5, 0.7\}$ are obtained for attribute A_1 ranging $[0.1, 0.9]$ for class c_1, then the triangular membership functions for A_1, for class c_1 are obtained as in Fig. 5.2.

A set of membership functions $\{MF_{A_i}\}$ is obtained for each attribute per class from the cluster points. This set of membership functions acts as the base for the crisp sets generated in Stage2 and defines all the descriptive fuzzy if-then rules generated in Stage3.

5.5.4 Stage2: - Generation of certain rule set from membership functions

The set of membership functions $\{MF_{A_i}\}$ obtained in Stage1, serves as the starting point for defining a crisp set of *certain* regions $\{MF_L\}$. These crisp sets are formed at data clusters so as to maximize the fuzzy lower limit approximations as shown in Fig. 5.3.

Fig. 5.3(b) plots a sample distribution of data for a two attribute classification case. Fig. 5.3(a) maps the data distribution on attribute A_1 to one of the triangular membership functions $\{MF_{A_1}\}$, obtained in Stage1 from the cluster points. To obtain the fuzzy lower approximation, the triangular membership function is considered as a singleton rule. The consequent class c_{lj} for this rule is identified by determining the class C_{max} that dominates among the training instances for the attribute A_1 by Eqn. 5.17.

$$C_{max}(x^k_{A_i}) = \text{argmax}_{C_m} \sum_{R_{lj}/c_{lj}=C_m} \mu_{R_{lj}}(x^k_{A_i}) \tag{5.17}$$

The maximum membership μ_u attained by instances belonging to other classes is as per Eqn. 5.18.

$$\mu_u = \{\max_{k|c^k \neq c_{lj}} (\mu_{A_i}(x^k_{A_i}))\} \tag{5.18}$$

All the instances which have memberships greater than μ_u can be conclusively identified as belonging to class c_{lj}. Attribute A_1 values for all the instances belonging to class c_{lj}, which have a membership larger than μ_u, are obtained. The attribute values obtained are sorted and the smallest A_1 indicated as x_{L1} and the largest A_1 indicated as x_{L2} are identified. These values serve as the limits of a crisp set as indicated in Fig. 5.3. Only instances belonging to class c_{lj} attain a membership of 1, while instances from other classes attain a membership of 0.

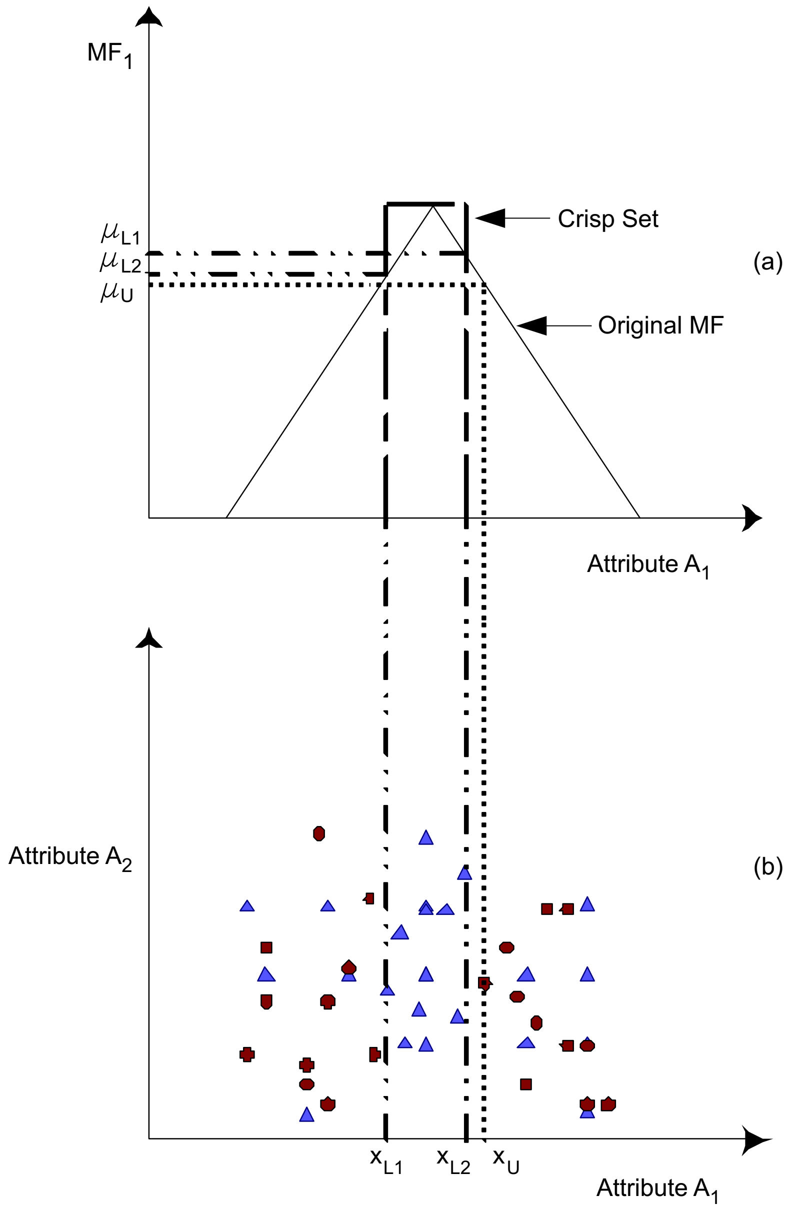


Figure 5.3: Obtaining crisp sets from cluster points

Each crisp set serves as a singleton attribute rule with $PF = 1$. The set of all such *certain* regions serves as the certain rule base and all instances which attain a membership of 1 for the certain rule base, form a fuzzy lower approximation for the partition induced by class c_{lj}. Some of the crisp sets classify only a small number of samples, if the samples from various classes are well mixed. Such rules are eliminated from the certain rule base and the few instances classified by them are handled in Stage3.

This method of obtaining singleton attribute rules does not identify all core knowledge as in a rough set reduct which is computationally NP-hard [121, 122]. However, it serves to generate concise rules for data pockets identified through the singleton attributes, while being computationally very simple.

5.5.5 Stage3: - Generation of possible rule set from membership functions

The training instances which are not classified by the certain rule set in Stage 2, serve as the training data set for Stage 3. In this stage, possible rules are generated based on fuzzy upper approximations derived from $\{MF_{A_i}\}$ through a boosting enhanced GA.

Boosting enhanced genetic algorithm for learning if-then rules

The overall architecture of the proposed approach is depicted in Fig. 5.4. The fuzzy rule chromosome population is created by the rule generator, from the set of membership functions $\{MF_{A_i}\}$ obtained in Stage1. The genetic rule learner selects the best rule from the population based on fitness, after employing the *crossover* and *mutation* genetic operators. The generated rule is added to the rule base and the weights of the training samples are adjusted by the boosting algorithm for the current set of rules. These steps are iterated till the error rate converges, or the

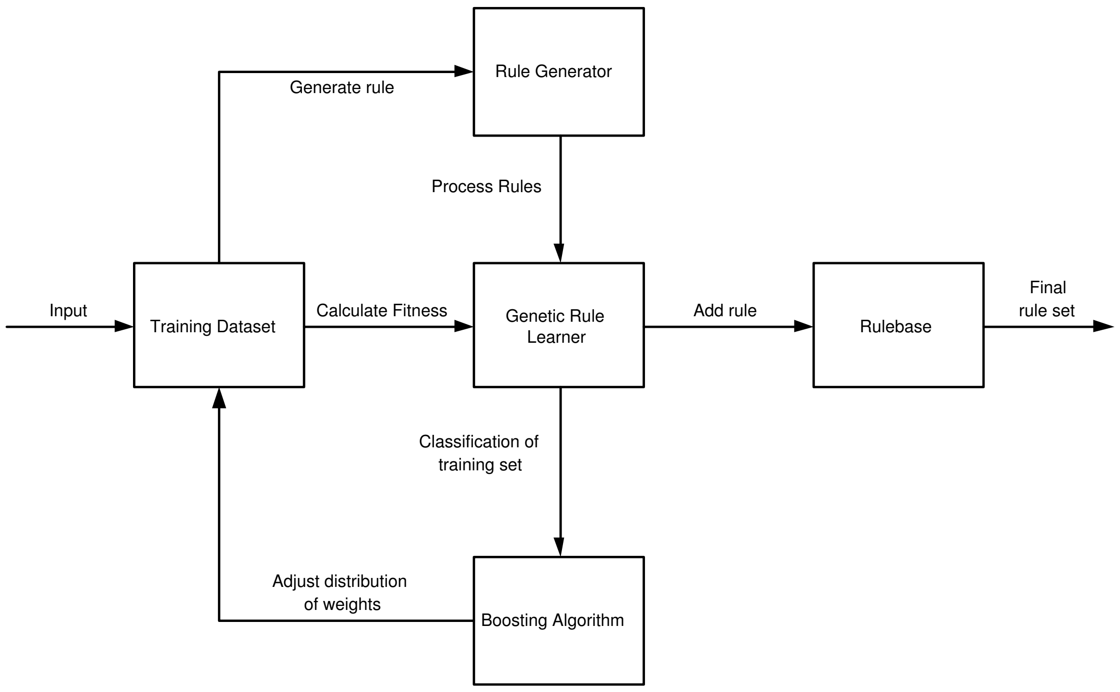


Figure 5.4: Boosting enhanced genetic fuzzy-rough classifier

maximum iteration count is reached. Each of these steps are explained in more detail, in the following sections.

Rule generation and encoding of fuzzy if-then rules

Every chromosome in the population encodes a single fuzzy rule, with each of the attributes $[A_i, i = 1, \ldots, t]^T$ being represented by a membership function. The membership function for each of the attributes A_i, is chosen at random from the set $\{MF_{A_i}\}$ obtained in Stage1. Each of the triangular membership functions in the set $\{MF_{A_i}\}$ is assigned an integer label. The triangular function for each attribute is encoded in the chromosomes by its integer label. The rule is represented by an integer valued vector $a_1, a_2, \ldots, a_t$ in the chromosome.

A specific fuzzy rule may involve most of the attributes. However, general fuzzy rules which are shorter, usually involve only a few of the attributes. To allow for the generation of general rules in addition to specific rules, the chromosome also encodes an additional bit string $S = \{s_1, \ldots, s_t\}$. Each of the bits $(s_1, \ldots, s_t)$ indicate the presence or absence of the respective attribute $(A_1, \ldots, A_t)$ in the fuzzy rule antecedent. When most of the bits in $(s_1, \ldots, s_t)$ become zero indicating the absence of respective attributes in the rule, it leads to the generation of a general rule. In contrast, when most of the bits in $(s_1, \ldots, s_t)$ become one indicating the presence of respective attributes in the rule, it leads to the generation of a specific rule. The consequent c_j is obtained by determining the class C_{max} that dominates among the training instances covered by the rule antecedent as in Eqn. 5.19.

$$C_{max}(x^k) = \text{argmax}_{C_m} \sum_{R_j/c_j=C_m} \mu_{R_j}(x^k) \tag{5.19}$$

The GA iterations are fast due to two reasons. The first reason is, the training set has become comparatively smaller after Stage2. The second reason is, the chromosome encodes only the integer labels of the membership functions, rather than real valued attributes of the triangular membership functions for each attribute. This reduces the length of the chromosome to one-third of that of a real valued chromosome. For data sets with large dimensionality, this leads to considerable increase in the speed of the iterations.

Fitness Criteria

The genetic rule learner evaluates each rule based on a fitness criteria. The proposed fitness criteria is designed for optimizing four objectives based on the fuzzy rough set perspective. Each objective is evaluated separately and finally aggregated into a single scalar fitness value in the range $[0, 1]$. The weights w_k assigned to individual samples by the boosting algorithm, based on the relative difficulty in classifying the sample, also forms a part in the fitness criteria.

The first objective is to ensure that the rule covers positive samples with large weights as compared to the negative samples. This is obtained by the weighted plausibility factor f_{j_1} as,

$$f_{j_1} = \frac{\sum\limits_{k|c^k=c_j} w_k \mu_{R_j}(x_k)}{\sum\limits_{k} w_k \mu_{R_j}(x_k)} \tag{5.20}$$

At the later stages of iteration, the weights of the unclassified samples tend to be high. To ensure that the rule is generic rather than covering only large weighted samples in the later stages, the plausibility factor defined in Eqn. 5.14 is utilized as another objective f_{j_2}.

$$f_{j_2} = \frac{\sum\limits_{k|c^k=c_j} \mu_{R_j}(x_k)}{\sum\limits_{k} \mu_{R_j}(x_k)} \tag{5.21}$$

The first two objectives indicate the relative aggregate membership attained by the samples belonging to class c_j. If there are a large number of samples from class c_j as compared to other classes in the training set, the objectives f_{j_1} and f_{j_2} do not ensure that the individual membership attained by samples belonging to c_j are large. The third measure f_{j_3} maximizes the individual membership attained by samples from class c_j.

$$f_{j_3} = \frac{\sum\limits_{k|c^k=c_j} \mu_{R_j}(x_k)}{l = \left|\mu_{R_j}(x_k) > 0 \forall k|c^k = c_j\right|} \tag{5.22}$$

The final objective f_{j_4} is to maximize the number of positive samples l_1 covered as compared to the negative ones l_2. This fitness value is normalized by the number of samples l_3 in the training set belonging to class c_j.

$$\begin{aligned} l_1 &= \left|\mu_{R_j}(x_k) > 0 \; \forall \; k|c^k = c_j\right| \\ l_2 &= \left|\mu_{R_j}(x_k) > 0 \; \forall \; k|c^k \neq c_j\right| \end{aligned}$$

$$
\begin{aligned}
l_3 &= \left|k|c^k = c_j\right| \\
f_{j_4} &= \begin{cases} 0 : \, l1 < l2 \\ \dfrac{l1 - l2}{l3} \end{cases}
\end{aligned}
\tag{5.23}
$$

The aggregate fitness f_j of the rule is computed as the product of individual measures as defined by Eqn. 5.24. The aggregate fitness is calculated as a product rather than a sum because, the aggregate membership should become zero, when one or more of the individual objective measures become zero.

$$
f_j = \prod_{1 \leq i \leq 4} f_{j_i} \tag{5.24}
$$

Iterative Boosting and assignment of weights

In the current proposal based on IRL approach, the first criteria is satisfied by the fitness defined in Eqn. 5.24. The training instances are assigned with weights, which are reduced upon classification, to satisfy the second criteria. The iterations are terminated when all the training instances are represented by at least one rule, or when the maximum number of iterations are reached, whichever occurs first.

In the current approach, all the training instances are assigned uniform weights $w^k = 1$ at the beginning of the iterations. After each iteration, the weights of all the instances successfully classified by the selected rule are reduced by a fixed quantity $d = 0.2$. The weights of all the misclassified instances are increased by the same fixed quantity $d = 0.2$ as in Eqn. 5.25. The weights of the unclassified instances remain unchanged. This ensures that the classified instances carry a lesser weight than the misclassified and unclassified instances. The focus of the next rule generation is more on the misclassified and unclassified instances.

$$w^k(t+1) = \begin{cases} 0 \ \text{ if classified and } w^k(t) - d < 0, \\ w^k(t) - d \ \text{ if classified by selected rule,} \\ w^k(t) + d \ \text{ if misclassified by selected rule} \end{cases} \tag{5.25}$$

Though the algorithm is designed to focus on the misclassified instances, some of the best rules generated tend to classify instances already covered by the existing rules. The new rule generated is not added to the rule base, if it does not cover at least a few instances which are previously misclassified or unclassified and the existing instance weights are not disturbed.

Once the final rule base is obtained, the classification of new instances is done based on the single winner rule, which is activated based on the fuzzy lower approximation. If more than one rule is activated, the winner is determined by the rule having the largest aggregate membership as in Eqn. 5.15. This allows for the intuitive interpretation of the fuzzy rule base.

5.6 Effect of Plausibility factor on rule learning and its comparison to Grade of certainty

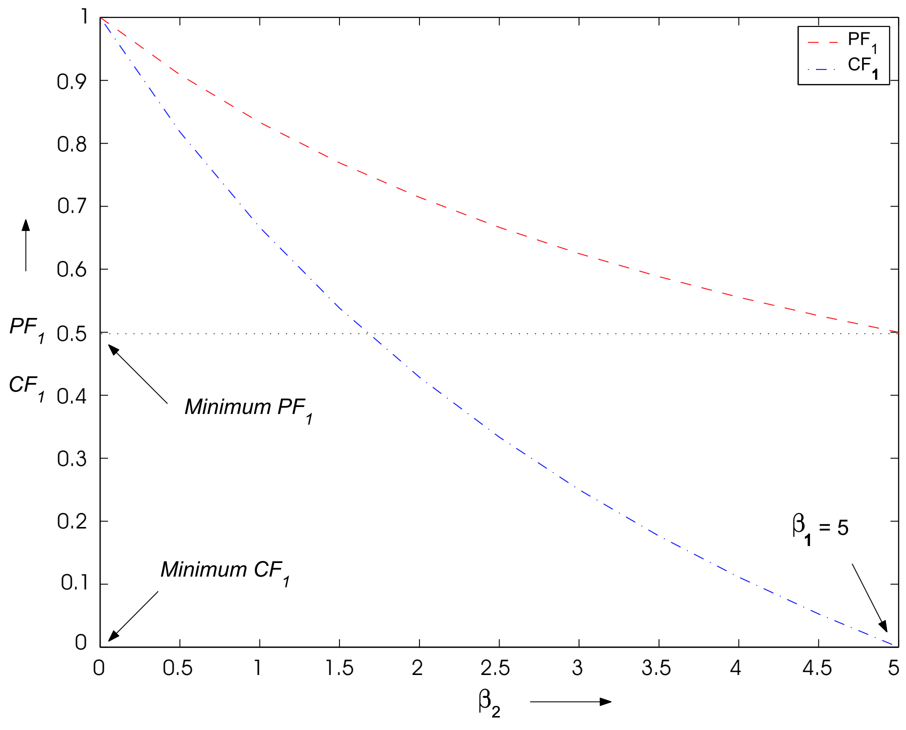
While PF is a normalized measure of the membership of rule class, CF is a normalized measure of the quantity by which the membership of rule class is larger than the membership of other classes. It has been observed that the plausibility factor PF_j has a learning effect similar to the certainty grade CF_j, though on a restricted scale. This restriction is due to the fact that, while the value of CF_j can range from $[0, 1]$, the value of PF_j ranges only from $[1/c, 1]$, where c is the number of consequent classes. This is because, if PF_j becomes less than $1/c$ for class c_j, it means that the rule classifies instances of some other class c'_j with a better aggregate membership. The rule generator would have assigned c'_j as the consequent class for rule R_j, with the effect that PF_j becomes greater than $1/c$.

Consider a sample single dimensional, two class classification problem for which CF and PF has been calculated as in Table. 5.1. For illustrative purposes, the aggregate membership of Class 1, β_1, has been maintained constant, while the aggregate membership of Class 2, β_2, has been varied over the range $[0, 5]$.

Table 5.1: Sample Two class membership, PF and CF

$\sum_{x_p \in Class\ 1} \mu_1(x_p)$	$\sum_{x_p \in Class\ 2} \mu_2(x_p)$	PF	CF
5	0.0	1	1
5	0.5	0.91	0.82
5	1.0	0.83	0.67
5	1.5	0.77	0.54
5	2.0	0.71	0.43
5	2.5	0.67	0.33
5	3.0	0.63	0.25
5	3.5	0.59	0.18
5	4.0	0.56	0.11
5	4.5	0.53	0.05
5	5.0	0.50	0.00

Fig. 5.5(a) illustrates the variation of the PF_1 and CF_1 over the range of β_2 for a constant β_1. As observed, PF_1 varies within a range $[0.5, 1]$ while CF_1 varies within $[0, 1]$. Moreover, the change in CF_1 is much more steeper then the change in PF_1 and both the changes are not linear. Fig. 5.5(b) illustrates the variation of the PF as the number of consequent classes c increases. It is observed that the range of learning for PF approaches that of CF as the number of consequent classes increase.



(a)

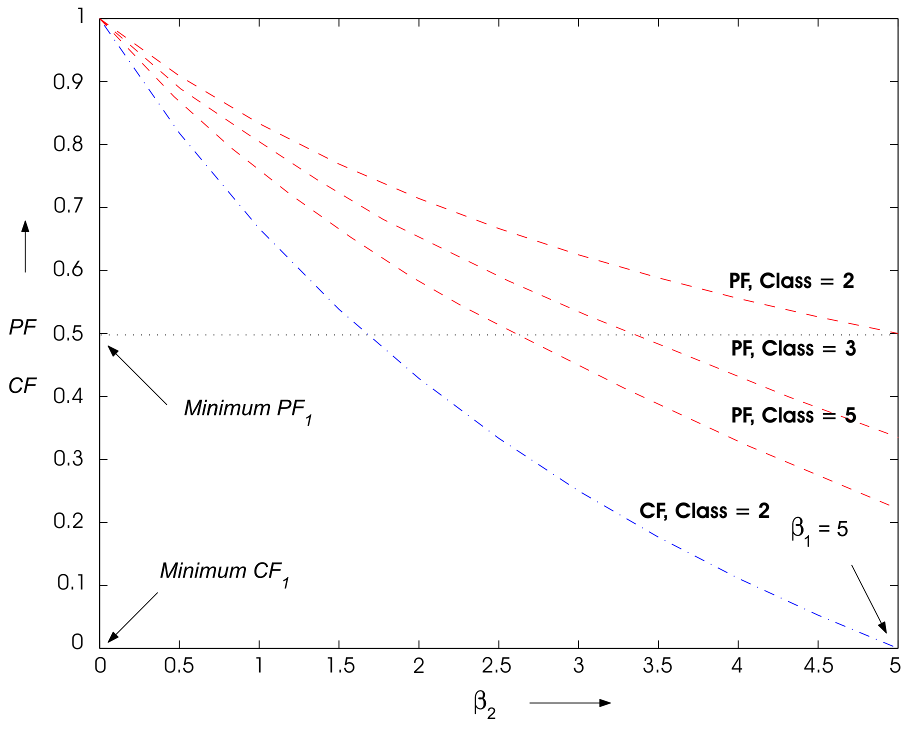


Figure 5.5: Variation of PF and CF

Consider a set of sample rules for the same two class problem, where the single attribute has been discretized into three fuzzy intervals defined by *small*, *medium* and *large* as in Fig. 5.6(a).

If x is *small*, then Class 1
If x is *medium*, then Class 2
If x is *large*, then Class 1

If CF or PF is not used in the fuzzy rule, the classification boundaries for optimal classification is obtained by modification of the membership functions as in Fig. 5.6(b).

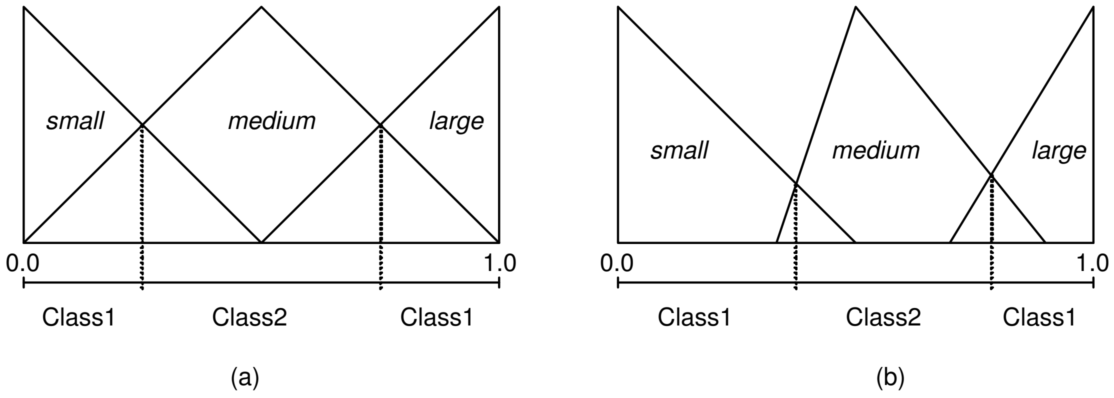


Figure 5.6: Adjusting classification boundaries by membership function modification

When CF or PF is used in the fuzzy rule, the classification boundaries get adjusted by variation of the respective factors, while the membership function shape, remains constant. This is illustrated in Fig. 5.7, where the dashed and dotted lines indicate the product of the respective factors CF and PF with the membership attained by samples for each fuzzy if-then rule.

Assume that a similar set of data exists for Class 2 as in Table. 5.1, with the aggregate membership values reversed. Fig. 5.7(a) illustrates a case where aggregate membership of Class 2 is 5.0 while that of Class 1 is 1.0. The classification boundary range assigned for Class 2 is larger for PF as compared to CF.

For case 2, where aggregate memberships are 5.0 and 2.5 respectively for Class 2 and Class 1, the classification boundaries are indicated in Fig. 5.7(b). In this case, CF_2 provides the same boundaries as that achieved by modifying the membership function in Fig. 5.6(b), while PF_2 continues to provide a larger boundary range. It is impossible to achieve this boundary with PF_2, as the PF_2 value required is 0.33 which is less than $1/c = 0.5$. While the boundary range assigned for Class 2 is larger for PF_2 as compared to CF_2, the right boundary values provided by CF_2 has overshot the one provided by PF_2.

Fig. 5.7(c) illustrates the case when CF and PF have attained their respective lowest values of 0.0 and 0.5. In this case, though the fuzzy rule employing CF gets activated for instances of Class 2, the final result provided by the CF weighted membership shows that the rule is not active for Class 2 instances. This is indicative that the rule can be removed from the final rule set. The fuzzy rule employing PF returns a non-zero PF weighted membership, which is exactly half of the original membership. In this case also, the rule can be safely removed from the final rule set.

The range of learning induced by PF is the lowest for a two class problem. However, as the number of classes c increases, the range of learning induced by $PF = [1/c, 1]$ approaches the range induced by CF. The range of learning while employing PF is much better for a multi-class problem as compared to a two class problem

In conclusion, while PF shows learning capabilities similar to CF, it does so on a restricted range. However, PF has a meaning in terms of rough set, since it measures the membership ratio of members belonging to the partition induced by class c_j, with the fuzzy upper approximation of rule R_j. The PF can only be

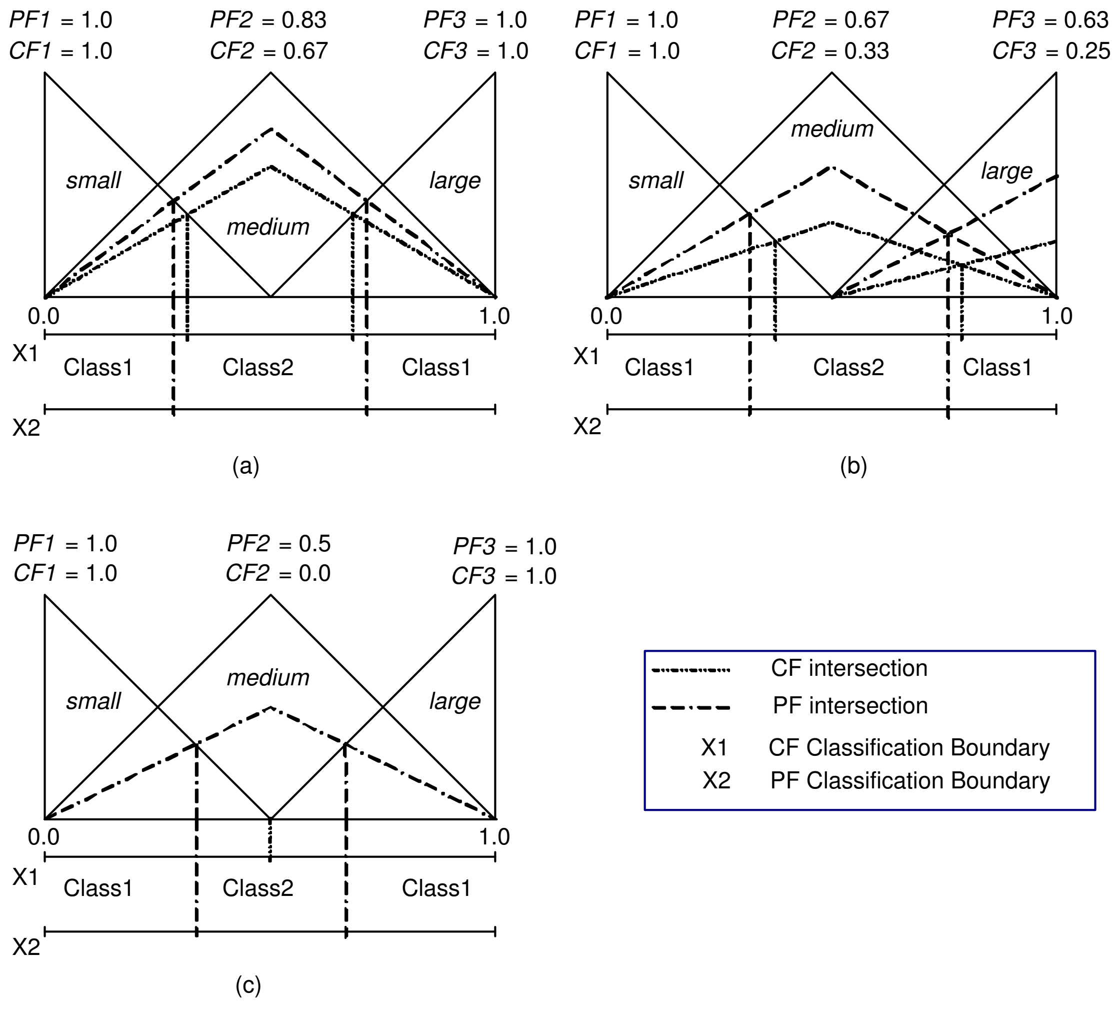


Figure 5.7: Adjusting classification boundaries by certainty and plausibility factors

derived from the properties of the training set and cannot be modified. However, CF has been defined as a heuristic quantity and can be modified for improving the performance of the rules, as has been done in [60]. However, performance evaluation of the proposed technique indicate that the usage of PF in combination with a certain and possible rule set can provide results comparable to well known classifiers in the literature, if the approximate data clusters are known before hand.

5.7 Performance Evaluation and discussion

The performance of the proposed system is evaluated on some well known data sets from the UCI database [89]. Two types of evaluation are performed on each data set. In the first type, the entire data set is taken as the training set. In the second type, predictive accuracy is measured by a ten-fold cross-validation procedure [123]. In essence, the data set is divided into ten mutually exclusive and exhaustive partitions. Then a classification algorithm is run ten times. Each time a different partition is used as the test set and the other nine partitions are used as the training set. The results of the ten runs (accuracy rate on the test set) are then averaged and reported as the accuracy rate of the discovered rule set. The attribute values are linearly normalized in the range [0,1] for all the data sets.

A crossover rate of 0.9 and a mutation rate of 0.1 are considered for the GA. The GA is allowed to run for a maximum of 50 generations before selecting the best rule. The number of newly created chromosomes in each generation is limited to 90% of the population. In each generation, a new population is formed based on the fitness value obtained from a combined parent and child pool. While the rule generation can be invoked until all samples are classified, the error rate starts converging after 20-25 rules. The rule generation is then stopped, though there are misclassified instances.

5.7.1 Performance Evaluation: Iris Data Set

The iris flower data set (SAS Institute, 1988) was originally published by Fisher (1936) for examples in discriminant and cluster analysis. The four attributes associated with Iris flower: sepal length, sepal width, petal length and petal width, are measured on Fifty iris specimens from each of three species: Iris setosa, Iris versicolor, and Iris virginica. The problem is to classify the specimens into three classes based on the four attributes.

Initially, the entire data set is used as the training set. It is observed that the majority of the samples in the data set can be classified with the certain rule set. Only the unclassified samples are used as training data, for the possible rule set. The population size is fixed to 20 chromosomes. For comparison, the famous rule-based machine learning algorithm C4.5 [124] and the statistical classifier Naive Bayes have been applied to the data set. Shi *et al.* [2] proposed a Pittsburgh approach based fuzzy learning system and reported performance of the system on the Iris data set. Table. 5.2 compares the performance of the proposed fuzzy rough classifier (FRC) technique with all the three classifier systems, when the entire data set is taken as training data, over ten runs. The fuzzy rough classifier outperformed the other three classifier techniques, though the best classification rate of 100% is also achieved by C4.5 and Naive Bayes.

Table 5.2: Performance comparison for Iris

Algorithm	Avg Accuracy %	Best Accuracy %
FRC	98.7	100
C4.5	93.7	100
Naive Bayes	95.5	100
Pittsburgh Approach [2]	98.5	98.5

5.7.2 Performance Evaluation: Wine Data Set

The wine data set consists of 178 samples from 3 classes with 13 continuous attributes. The wine data set is known to be fully classifiable, in the literature. However, the difficulty in classifying this data set is due to the relatively large number of attributes. This data set has been chosen to observe the classification ability of the proposed classifier for the large number of attributes. Ishibuchi *et al.* [60] compared the performance of the CF based fuzzy classifier with the results published by Corcoran *et al.* [90]. Table. 5.3 compares the performance of the proposed classifier with that of Ishibuchi [60] and Corcoran [90]. The performance of the fuzzy rough classifier is similar to that of the classifier with CF modification, reported by Ishibuchi *et al.* [60].

Table 5.3: Performance comparison for Wine

Algorithm	Avg Accuracy %	Best Accuracy %
FRC	100	100
Ishibuchi without CF modification [60]	98.5	99.4
Ishibuchi with CF modification [60]	100	100
Corcoran [90]	99.5	100

5.7.3 Performance Evaluation: Glass Data Set

The glass data set consists of 214 samples with 9 continuous attributes spread over 6 classes. It is observed that the data pockets in this data set are well mixed and very narrow. The proposed classifier generated around 14 certain rules of which only 4 rules covered a large number of samples. The other certain rules covered only 2 or 3 instances and are pruned from the data set. Hence, the number of certain rules generated is very less as compared to the other two data sets.

The CF based fuzzy classifier is compared by Ishibuchi *et al.* [60] with the results published for 1R [92] and C4 [93] algorithms. In this comparison, 2/3 of the data samples are used as training data with the rest being considered as test data. The same testing technique is adopted and results are compared in Table. 5.4. The performance is almost the same as that reported for CF based fuzzy classifier [60] and C4 [93] algorithm.

Table 5.4: Performance comparison for Glass

Algorithm	Avg Accuracy %
FRC	63.1
Ishibuchi [60]	64.4
1R [92]	53.8
C4 [93]	63.2

5.7.4 Performance Evaluation: Summary

In Table. 5.5, the performance comparison between the 'test with entire data set' and the 'ten fold test' is provided for all the data sets.

Fig. 5.8(a) shows a bar chart of the number of certain and possible rules, when the entire data set is used for training and during the ten fold test. Fig. 5.8(b), Fig. 5.8(c) and Fig. 5.8(d) show the relative percentage of the total classification performed, by the certain and possible rules for each of the data sets, when the training set comprises the entire data.

When the majority of the instances are well separated or at least form distinctive data clusters as in the case of Iris and Wine data sets, more number of certain rules are generated and the bulk of the classification is performed by the certain rules. However, when the data clusters are very narrow as in the case of glass data

Table 5.5: Comparison of training with entire data set and ten fold test

	Data set	Avg # certain rules	Avg # possible rules	Avg Accuracy %	Best Accuracy %
Training with entire data set	Iris	12	5	98.7	100
Ten Fold Test	Iris	10	6	96.7	98.4
Training with entire data set	Wine	14	8	100	100
Ten Fold Test	Wine	11	10	98.3	99.5
Training with entire data set	Glass	4	19	67.2	69.3
Ten Fold Test	Glass	3	17	61.7	63.4

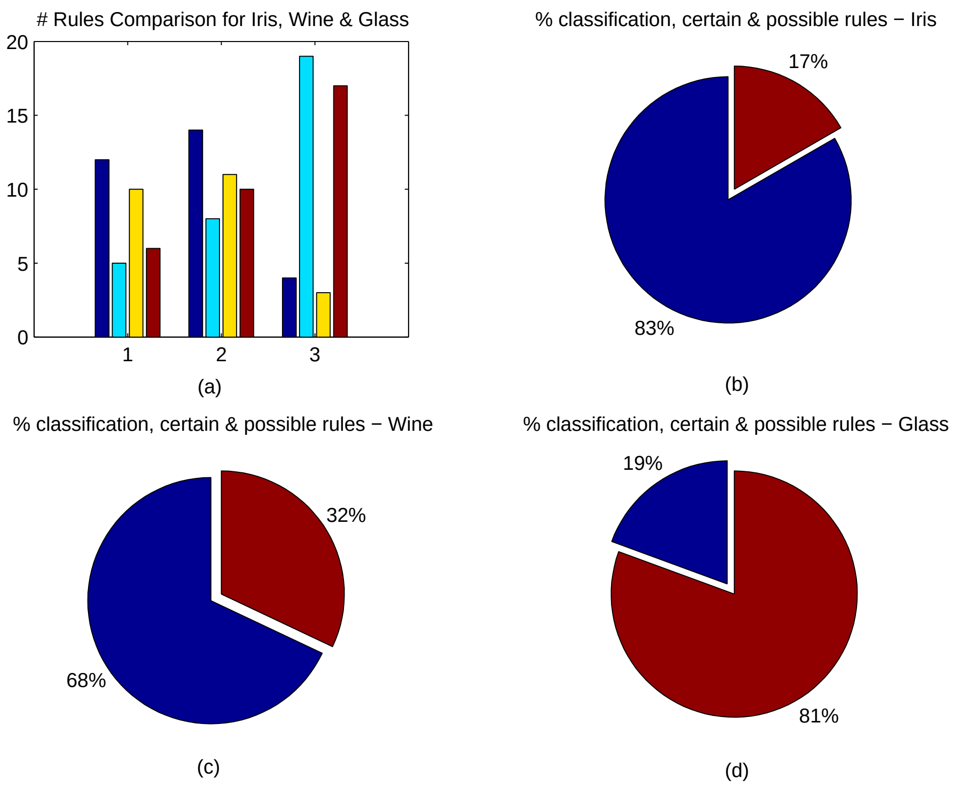


Figure 5.8: Comparison of the performance by certain and possible rules

set, only a few certain rules are obtained, with the classification being performed mainly by the possible rule set. This shows that the classifier is able to handle data which are clearly separable as well as inter-mixed, with a trade-off between the certain and possible rule sets.

5.8 Conclusion

A classifier based on the concepts of fuzzy equivalence relation is proposed. A certain rule set is generated based on the fuzzy lower approximation. A possible rule set based on fuzzy upper approximation is generated by a boosting enhanced genetic algorithm. The proposed fuzzy rules are weighted with a plausibility factor. The effect of the plausibility factor in learning classification boundaries is studied and compared with that of the heuristic certainty grade available in the literature. The performance of the proposed classifier has been evaluated with some of the well known data sets and compared with standard classification techniques.

Chapter 6

Future Directions

This chapter provides a brief outline of some of the open questions and possible future directions for research.

- Reducts from rough sets provide an efficient way to extract the core knowledge from a data set and result in a compact rule set. This rule set can be used for pattern classification. However, these rules tend to be exact rules, which have to be made more generic to cater to real world classification problems. Another potential area for improvement, is the formation of reduct for continuous data. Chapter 3 outlined a technique for handling continuous attributes by discretizing them. Fuzzy Systems are pretty good at representing continuous attributes by membership functions. Additionally, the introduction of fuzzy logic in reducts, will make the resultant rule set more generic. While Some research has already been done, there remains lot of scope for improvement.

- While the reduct rule-based models have potential for offering traceable or explainable classification, this approach is sometimes hampered by the size of the rule set. Schemes to make large rule-based models more manageable are clearly of interest to develop.

- Real life problems often lack a complete data set and have missing attributes. Many a times, records with missing attributes are rejected and the classifier is developed with the rest of the data set. Rough set theory provides a way to estimate dependency and the impact of missing attributes. More research in this area of missing attributes is of importance to develop robust classifiers for real world problems.

- A lot of research has been going on for integrating GAs with fuzzy systems. Chapter 4 outlined a technique for improving the performance of a GA driven fuzzy classifier. There remains scope for improvement in this technique as has been outlined in that chapter. Particular attention will have to be given to reduce the time and computational complexity of the Pittsburgh approach which has been adopted. Performance can also be improved by the usage of co-evolutionary algorithms for dimensionality reduction while evolving the fuzzy system.

- While the performance of the proposed classifier in Chapter 5 is good, the number of rules generated seems to be on the higher side as compared to the purely fuzzy techniques (as opposed to rough set techniques) in the literature. This is mainly due to the fact that the certain rule set identifies singleton attribute rules. If a simpler technique for obtaining certain rules with multiple attributes can be integrated in Stage2, this will considerably reduce the number of certain rules.

Bibliography

[1] Lotfi A. Zadeh. Fuzzy logic, neural networks and soft computing, November 1992. One-page course announcement of CS 294-4, Spring 1993, University of California, Berkeley.

[2] Yuhui Shi, Russell Eberhart, and Yaobin Chen. Implementation of evolutionary fuzzy systems. *IEEE Transactions on Fuzzy Systems*, 7(2):109–119, April, 1999.

[3] Z. Pawlak. Rough sets. *International Journal of Computer and Information Science*, 11:341–356, 1982.

[4] Z. Pawlak. *Rough Sets: Theoretical Aspects of Reasoning about Data.* Kluwer Academic Publishers, Dordrecht, 1991.

[5] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[6] G. Shafer. *A Mathematical Theory of Evidence.* Princeton University Press, Princeton, New York, 1976.

[7] D. Dubois and H. Prade. Rough fuzzy sets and fuzzy rough sets. *International Journal of General Systems*, 17:191–209, 1990.

[8] D. Dubois and H. Prade. Putting rough sets and fuzzy sets together. In Roman Slowinski, editor, *Intelligent Decision Support: Handbook of Applications and Advances in Rough Sets Theory*, volume 11 of *Series D: System Theory, Knowledge Engineering and Problem Solving*, pages 203–232. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

[9] J. W. Grzymala-Busse. Dempster-shafer theory interpretation of rough set approach to knowledge acquisition under uncertainty. Technical report, Department of Computer Science, Uninversity of Kansas, 1988.

[10] A. Skowron. The relationship between the rough set theory and evidence theory. *Bulletin of Polish Academy of Sciences*, 37:87–90, 1989.

[11] Andrzej Skowron and Cecylia Rauszer. The discernibility matrices and functions in information systems. In Roman Slowinski, editor, *Intelligent Decision Support: Handbook of Applications and Advances in Rough Sets Theory*, volume 11 of *Series D: System Theory, Knowledge Engineering and Problem Solving*, pages 331–362. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

[12] Zdzislaw Pawlak and Andrzej Skowron. Rough membership functions. In R. Yager, M. Fedrizzi, and J. Kacprzyk, editors, *Advances in the Dempster Shafer Theory of Evidence*, pages 251–271. John Wiley and Sons, 1994.

[13] Aleksander Ohrn. *Discernibility and Rough Sets in Medicine: Tools and Applications*. PhD thesis, Norwegian University of Science and Technology, Department of Computer and Information Science, 1999.

[14] Torulf Mollestad. *A Rough Set Approach to Data Mining: Extracting a Logic of Default Rules from Data*. PhD thesis, Norwegian University of Science and Technology, Department of Computer and Information Science, February, 1997.

[15] R. S. Michalski. A theory and methodology of inductive learning. *Artificial Intelligence*, 20:111–161, 1983.

[16] J. W. Shavlik and T. G. Dietterich, editors. *Readings in machine learning.* Morgan Kaufmann Publishers, 1990.

[17] J. W. Grzymala-Busse. Knowledge acquisition under uncertainty: a rough set approach. *Journal of Intelligent Robotic Systems*, 1:3–16, 1988.

[18] J. W. Grzymala-Busse. Lers - a system for learning from examples based on rough sets. In R. Slowinski, editor, *Intelligent Decision Support- Handbook of Applications and Advances of the Rough Set Theory*, pages 3–18. Kluwer Academic, Dordrecht, 1992.

[19] A. Skowron. Boolean reasoning for decision rules generation. In J. Komorowski and R. W. Ras, editors, *Methodologies for Intelligent Systems*, pages 295–305. Springer-Verlag, Berlin, 1993.

[20] R. Slowinski and J. Stefanowski. Roughdas and roughclass software implementation of the rough sets approach. In Roman Slowinski, editor, *Intelligent Decision Support: Handbook of Applications and Advances in Rough Sets Theory*, Series D: System Theory, Knowledge Engineering and Problem Solving, pages 445–456. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

[21] J. Stefanowski and D. Vanderpooten. A general two-stage approach to inducing rules from examples. In W. Ziarko, editor, *Rough Sets, Fuzzy Sets and Knowledge Discovery*, pages 317–325. Springer-Verlag, Berlin, 1994.

[22] W. Ziarko. Variable precision rough set model. *Journal of Computer and System Sciences*, 46:39–59, 1993.

[23] J. W. Grzymala-Busse and X. Zou. Classification strategies using certain and possible rules. In *Rough Sets and Current Trends in Computing, Proc. RSCTC'98 Conf.*, pages 37–44, Warsaw, 1998.

[24] J. Stefanowski. On rough set based approaches to induction of decision rules. In L. Polkowski and A. Skowron, editors, *Rough Sets in Knowledge Discovery*, volume 1, pages 500–529. Physica Verlag, Heidelberg, 1998.

[25] T. Y. Lin. Rough set theory in very large database. In *Proceedings of Symposium on Modeling, Analysis and Simulation, Computational Engineering in Systems Applications (CESA'96), IMACS Multi Conference*, volume 2, pages 936–941, Lille, France, July, 1996.

[26] T. Y. Lin and Y. Y. Yao. Mining soft rules using rough sets and neighbourhoods. In *Proceedings of Symposium on Modeling, Analysis and Simulation, Computational Engineering in Systems Applications (CESA'96), IMACS Multi Conference*, volume 2, pages 1095–1100, Lille, France, July, 1996.

[27] J. W. Grzymala-Busse and X. Zou. Lers - a knowledge discovery system. In L. Polkowski and A. Skowron, editors, *Rough Sets in Knowledge Discovery 2*, pages 562–565. Physica-Verlag, Heidelberg, 1998.

[28] R. Slowinski and J. Stefanowski. Rough family - software implementation of the rough set theory. In L. Polkowski and A. Skowron, editors, *Rough Sets in*

Knowledge Discovery, volume 2, pages 581–586. Physica Verlag, Heidelberg, 1998.

[29] J. G. Bazan, A. Skowron, and P. Synak. Market data analysis: A rough set approach, ics research reports 6/94. Technical report, Warsaw Uninversity of Technology, 1994.

[30] J. G. Bazan and M. Szczuka. Rses and rseslib - a collection of tools for rough set computation. In *Proceedings of second International Conference on Rough Sets and Current Trends in Computing, RSCTC'2000*, pages 74–81, Banff, Canada, 2000.

[31] J. G. Bazan. A comparison of dynamic and non-dynamic rough set methods for extracting laws from decision table. In L. Polkowski and A. Skowron, editors, *Rough Sets in Knowledge Discovery*, volume 1, pages 321–365. Physica Verlag, Heidelberg, 1998.

[32] W. Ziarko, R. Golan, and D. Edwards. An application of datalogic/r knowledge discovery tool to identify strong predictive rules in stock market data. In *Proceedings of AAAI Workshop on Knowledge Discovery in Databases*, pages 93–101, Washington D.C., USA, 1993.

[33] A. Lenarcik and Z. Piasta. Rough classifiers. In W. Ziarko, editor, *Rough Sets, Fuzzy Sets and Knowledge Discovery*, pages 298–316. Springer-Verlag, London, 1994.

[34] W. Kowalczyk. A tool for rough data analysis, classification and clustering. In L. Polkowski and A. Skowron, editors, *Rough Sets in Knowledge Discovery*, volume 2, pages 566–568. Physica Verlag, Heidelberg, 1998.

[35] A. Lenarcik and Z. Piasta. Probrough - a system for probabilistic rough classifiers generation. In L. Polkowski and A. Skowron, editors, *Rough Sets in*

Knowledge Discovery, volume 2, pages 569–571. Physica Verlag, Heidelberg, 1998.

[36] L. Polkowski and A. Skowron, editors. *Rough Sets in Knowledge Discovery 1: Methodology and Applications*, volume 18 of *Studies in Fuzziness and Soft Computing*. Physica-Verlag, Heidelberg, Germany, 1998.

[37] J. Komorowski, A. Ohrn, and A. Skowron. Rosetta and other software systems for rough sets. In W. Klosgen and J. Zytkow, editors, *Handbook of Data Mining and Knowledge Discovery*. Oxford University Press, 2000.

[38] H. Ishibuchi, K. Nozaki, and H. Tanaka. Distributed representation of fuzzy rules and its application to pattern classification. *Fuzzy Sets and Systems*, 52(1):21–32, November, 1992.

[39] H. Ishibuchi, K. Nozaki, and H. Tanaka. Efficient fuzzy partition of pattern space for classification problems. *Fuzzy Sets and Systems*, 59:295–304, 1993.

[40] L. X. Wang and J. M. Mendel. Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(6):1414–1427, 1992.

[41] R. Rovatti and R. Guerrieri. Fuzzy sets of rules for system identification. *IEEE Transactions on Fuzzy Systems*, 4:89–102, 1996.

[42] S. Abe and M. S. Lan. A method for fuzzy rules extraction directly from numerical data and its application to pattern classification. *IEEE Transactions on Fuzzy Systems*, 3:18–28, 1995.

[43] S. Abe and M. S. Lan. Fuzzy rules extraction directly from numerical data for function approximation. *IEEE Transactions on Systems, Man and Cybernetics*, 25:119–129, 1995.

[44] T. Hong and C. Lee. Induction of fuzzy rules and membership functions from training examples. *Fuzzy Sets and Systems*, 84:33–37, 1996.

[45] R. Fuller. *Introduction to Neuro-Fuzzy Systems.* Physica-Verlag, Wurzburg, 1999.

[46] J.S.R. Jang, C.T. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing.* Prentice Hall, NJ, 1997.

[47] D. Nauck, F. Klawoon, and R. Kruse. *Foundations of Neuro-Fuzzy Systems.* Wiley, New York, 1997.

[48] P.P. Angelov. *Evolving Rule-Based Models. A Tool for Design of Flexible Adaptive Systems.* Physica-Verlag, Wurzburg, 2002.

[49] O. Cordon, F. Herrera, F. Hoffmann, and L. Magdalena. *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases.* World Scientific, Singapore, 2001.

[50] F. Herrera and J. L. Verdegay, editors. *Genetic Algorithms and Soft Computing.* Physica-Verlag, Wurzburg, 1996.

[51] W. Pedrycz, editor. *Fuzzy Evolutionary Computation.* Kluwer Academic Publishers, Dordrecht, 1997.

[52] E. Sanchez, T. Shibata, and L. Zadeh, editors. *Genetic Algorithms and Fuzzy Logic Systems: Soft Computing Perspectives.* World Scientific, Singapore, 1997.

[53] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

[54] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, 1975.

[55] M. Valenzuela-Rendon. The fuzzy classifier system: A classifier system for continuously varying variables. In *Proceedings of fourth International Conference on Genetic Algorithms (ICGA)*, pages 346–353, San Diego, CA, July, 1991.

[56] A. Parodi and P. Bonelli. A new approach to fuzzy classifier systems. In *Proceedings of fifth International Conference on Genetic Algorithms*, pages 223–230, Urbana-Champaign, IL, July, 1993.

[57] T. Furuhashi, K. Nakaoka, and Y. Uchikawa. Suppression of excessive fuzziness using multiple fuzzy classifier systems. In *Proceedings of third IEEE International Conference on Fuzzy Systems*, pages 411–414, Orlando, FL, June, 1994.

[58] K. Nakaoka, T. Furuhashi, and Y. Uchikawa. A study of apportionment of credits of fuzzy classifier system for knowledge acquisition of large scale systems. In *Proceedings of third IEEE International Conference on Fuzzy Systems*, pages 1797–1800, Orlando, FL, June, 1994.

[59] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka. Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 3(2):260–270, August, 1995.

[60] H. Ishibuchi, T. Nakashima, and T. Murata. Performance evaluation of fuzzy classifier systems for multi-dimensional pattern classification problems. *IEEE Trans. on Systems, Man, and Cybernetics- Part B: Cybernetics*, 29(5):601–618, October, 1999.

[61] T. Nakashima, G. Nakai, and H. Ishibuchi. Improving the performance of fuzzy classifier systems by membership function learning and feature selection. In *Proceedings of IEEE International Conference on Fuzzy systems*, pages 488–493, Honolulu, USA, May, 2002.

[62] W. R. Hwang and W. E. Thompson. Design of intelligent fuzzy logic controllers using genetic algorithms. In *Proceedings of Third IEEE International Conference on Fuzzy Systems*, pages 1383–1388, Orlando, June, 1994.

[63] P. Thrift. Fuzzy logic synthesis with genetic algorithms. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 509–513, San Diego, CA, July, 1991.

[64] C. L. Karr and E. J. Gentry. Fuzzy control of ph using genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 1:46–53, February, 1993.

[65] A. Homaifar and E. McCormick. Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 3:129–139, May, 1995.

[66] M. A. Lee and H. Takagi. Dynamic control of genetic algorithms using fuzzy logic techniques. In *Proceedings of International Conference on Genetic Algorithms*, pages 76–83, Urbana-Champaign, IL, July, 1993.

[67] A. Gonzalez and F. Herrera. Multi-stage genetic fuzzy systems based on the iterative rule learning approach. *Mathware Soft Computing*, 4:233–249, 1997.

[68] O. Cordon and F. Herrera. A three-stage method for designing genetic fuzzy systems by learning from examples. In *4th International Conference on Parallel Problem Solving from Nature (PPSN IV)*, pages 720–729, Berlin, Germany, 1996.

[69] O. Cordon, M. J. del Jesus, F. Herrera, and E. Lopez. Selecting fuzzy rule-based classification systems with specific reasoning methods using genetic algorithms. In *Seventh International Fuzzy Systems Association World Congress (IFSA)*, volume 2, pages 424–429, Praga, Czech, 1997.

[70] A. Gonzalez and R. Perez. Slave: A genetic learning system based on an iterative approach. *IEEE Transactions on Fuzzy Systems*, 7(2):176–191, 1999.

[71] Frank Hoffmann. Combining boosting and evolutionary algorithms for learning of fuzzy classification rules. *Fuzzy Sets and Systems*, 141:47–58, 2004.

[72] L. Wang and J. Yen. Extracting fuzzy rules for system modeling using a hybrid of genetic algorithms and kalman filter. *Fuzzy Sets and Systems*, 101:353–362, 1999.

[73] D. Park, A. Kandel, and G. Langholz. Genetic-based new fuzzy reasoning models with application to fuzzy control. *IEEE Transactions on Systems, Man, and Cybernetics*, 24:39–47, January, 1994.

[74] P. P. Bonissone, R. Subbu, and K. S. Aggour. Evolutionary optimization of fuzzy decision systems for automated insurance underwriting. In *Proceedings of IEEE International Conference on Fuzzy Systems, FUZZ-IEEE02*, pages 1003–1008, 2002.

[75] I. G. Damousis and P. Dokopoulos. A fuzzy expert system for the forecasting of wind speed and power generation in wind farms. In *Proceedings of International Conference on Power Industry Computer Applications, PICA 2001*, pages 63–69, Sydney, Australia, June, 2001.

[76] T.Y. Lin and A.M. Wildberger, editors. *Soft Computing: the Third International Workshop on Rough Sets and Soft Computing.* The Society of Computer Simulation, San Diego, CA, 1995.

[77] W. Ziarko, editor. *Rough Sets, Fuzzy Sets and Knowledge Discovery.* Springer Verlag, London, 1994.

[78] K. Slowinski. Rough classification of hsv patients. In Roman Slowinski, editor, *Intelligent Decision Support: Handbook of Applications and Advances in Rough Sets Theory*, volume 11 of *Series D: System Theory, Knowledge Engineering and Problem Solving*, pages 373–389. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

[79] M. G. Kendall, A. Stuart, and J.K. Ord. *The advanced Theory of Statistics*, volume III of *Design and Analysis and Time Series.* Charles Griffin and Co., London, fourth edition, 1983.

[80] R. O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* John Wiley, New York, 1973.

[81] T. Y. Lin and N. Cercone, editors. *Rough Sets and Data Mining: Analysis of Imprecise Data.* Kluwer Academic Publishers, Norwell, MA, USA, 1996.

[82] J. C. Bezdek. On the relationship between neural networks, pattern recognition and intelligence. *International Journal of Approximate Reasoning*, 6:85–107, 1992.

[83] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man and Cybernetics*, 15(1):116–132, 1985.

[84] J. S. R. Jang. Fuzzy controller design without domain experts. In *Proceedings of IEEE International Conference on Fuzzy Systems*, pages 289–296, San Diego, CA, March, 1992.

[85] M. Sugeno and T. Yasukawa. A fuzzy-logic based approach to qualitative modeling. *IEEE Transactions on Fuzzy Systems*, 1:7–31, February, 1993.

[86] B. Carse, T. C. Fogarty, and A. Munro. Evolving fuzzy rule based controllers using genetic algorithms. *Fuzzy Sets and Systems*, 80:273–293, June, 1996.

[87] H. Inoue, K. Kamei, and K. Inoue. Automatic generation of fuzzy rules using hyper-elliptic cone membership functions by genetic algorithms. In *Proceedings of 7th IFCA World Congress*, volume II, pages 383–388, Prague, Czech Republic, June, 1997.

[88] K. Shimojima, T. Fukuda, and Y. Hasegawa. Rbf-fuzzy system with ga based unsupervised/supervised learning method. In *Proceedings of International Joint 4th IEEE Conference on Fuzzy Systems. / 2nd International. Fuzzy Eng. Symp. (FUZZ/IEEE-IFES)*, volume 1, pages 253–258, Yokohama, Japan, March, 1995.

[89] C. L. Blake and C. J. Merz. Uci repository of machine learning databases, 1998.

[90] A. L. Corcoran and S. Sen. Using real-valued genetic algorithms to evolve rule sets for classification. In *Proceedings of first IEEE International Conference on Evolutionary Computing*, pages 120–124, Orlando, FL, June, 1994.

[91] H. Ishibuchi and T. Murata. Minimizing the fuzzy rule base and maximizing its performance by a multi-objective genetic algorithm. In *Proceedings of 1997 IEEE International Conference on Fuzzy Systems*, pages 259–264, Barcelona, Spain, July, 1997.

[92] R. C. Holte. Very simple classification rules perform well on most commonly used dataset. *Machine Learning*, 11(1):63–90, 1993.

[93] J. R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27:221–234, September, 1987.

[94] D. Dubois and H. Prade. Epistemic entrenchment and possibilistic logic. *Artificial Intelligence*, 50:223–239, 1991.

[95] Z. Pawlak. Rough classification. *International Journal of Man-Machine Studies*, 20:469–483, 1984.

[96] Ewa Orlowska. Reasoning with incomplete information: Rough set based information logics. In V. S. Alagar, S. Bergler, and F. Q. Dong, editors, *Incompleteness and Uncertainty in Information Systems, Proceedings of the SOFT-EKS Workshop on Incompleteness and Uncertainty in Information Systems*, Workshops in Computing, pages 16–33, Montreal, Canada, October, 1993. Springer.

[97] L. T. Germano and P. Alexandre. Knowledge-base reduction based on rough set techniques. In *Canadian Conference on Electrical and Computer Engineering*, pages 278–281, Canada, 1996.

[98] P. J. Lingras and Y. Y. Yao. Data mining using extensions of the rough set model. *Journal of American Society for Information Science*, 49(5):415–422, 1998.

[99] Ning Zhong, Ju-Zhen Dong, S. Ohsuga, and Tsau Young Lin. An incremental probabilistic rough set approach to rule discovery. In *Proceedings of IEEE World Congress on Computational Intelligence (WCCI'98)*, volume 2, pages 933–938. IEEE Press, 1998.

[100] Jack David Katzberg and Wojciech Ziarko. Variable precision extension of rough sets. *Fundamenta Informaticae*, 27(2, 3):155–168, August, 1996.

[101] A. Lenarcik and Z. Piasta. Discretization of condition attributes space. In Roman Slowinski, editor, *Intelligent Decision Support: Handbook of Applications and Advances in Rough Sets Theory*, volume 11 of *Series D: System Theory, Knowledge Engineering and Problem Solving*, pages 373–389. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

[102] H. S. Nguyen and A. Skowron. Quantization of real valued attributes, rough set and boolean reasoning approaches. In *Proc. Second Annual Conf. Information Sciences*, pages 34–37, Wrightsville Beach, NC, USA, 1995.

[103] N. H. Nguyen and N. S. Nguyen. Discretization methods in data mining. In L. Polkowski and A. Skowron, editors, *Rough Sets in Knowledge Discovery*, volume 1, pages 451–452. Physica Verlag, Heidelberg, 1998.

[104] I. Graham and P. L. Jones. *Expert systems-knowledge, uncertainty and decision.* Chapman and Computing, Boston, 1988.

[105] H. J. Zimmermann. *Fuzzy set theory and its applications.* Kluwer Academic Publishers, Boston, 1991.

[106] T. P. Hong, T. T. Wang, and S. L. Wang. Knowledge acquisition from quantitative data using the rough-set theory. *Intelligent Data Analysis*, 4:289–304, 2000.

[107] T. P. Hong, T. T. Wang, S. L. Wang, and B. C. Chien. Learning a coverage set of maximally general fuzzy rules by rough sets. *Expert Systems with Applications*, 19:97–103, 2000.

[108] Qiang Shen and Alexios Chouchoulas. A rough-fuzzy approach for generating classification rules. *Pattern Recognition*, 35:2425–2438, 2002.

[109] Amitava Roy and K. P. Sankar. Fuzzy discretization of feature space for a rough set classifier. *Pattern Recognition Letters*, 24:895–902, 2003.

[110] H. Ishibuchi and T. Nakashima. Effect of rule weights in fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems*, 9(4):506–515, August, 2001.

[111] D. Nauck and R. Kruse. How the learning of rule weights affects the interpretability of fuzzy systems. In *Proceedings of 7th IEEE International Conference on Fuzzy Systems*, pages 1235–1240, Anchorage, AK, May, 1998.

[112] S. Chiu. Fuzzy model identification based on cluster estimation. *Journal of Intelligent and Fuzzy Systems*, 2(3), September, 1994.

[113] R. Yager and D. Filev. Generation of fuzzy rules by mountain clustering. *Journal of Intelligent and Fuzzy Systems*, 2(3):209–219, 1994.

[114] Yoav Freund and Robert E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, September, 1999.

[115] Robert E. Schapire. The boosting approach to machine learning - an overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

[116] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of 13th International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1996.

[117] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

[118] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, August, 1997.

[119] L. Junco and L. Sanchez. Using the adaboost algorithm to induce fuzzy rules in classification problems. In *Proceedings of Spanish Conference for Fuzzy Logic and Technologies*, pages 297–301, Sevilla, Spain, 2000.

[120] A. Gonzalez and R. Perez. Completeness and consistency conditions for learning fuzzy rules. *Fuzzy sets and Systems*, 96:37–51, 1998.

[121] J. Komorowski, L. Polkowski, and A. Skowron. Learning tolerance relations by boolean descriptors: Automatic feature extraction from data tables. In S. Tsumoto, S. Kobayashi, T. Yokomori, H. Tanaka, and A. Nakamura, editors, *RSFD'96: The Fourth International Workshop on Rough Sets, Fuzzy Sets, and Machine Discovery*, pages 11–17, University of Tokyo, November, 1996.

[122] J. Komorowski, Z. Pawlak, L. Polkowski, and A. Skowron. Rough sets: A tutorial. In S.K. Pal and A. Skowron, editors, *Rough fuzzy hybridization: A new trend in decision-making*, pages 3–98. Springer-Verlag, Singapore, 1999.

[123] S. M. Weiss and C. A. Kulikowski. *Computer Systems That Learn.* Morgan Kaufmann, San Francisco, 1991.

[124] J. R. Quinlan. *Neuro-Fuzzy and Soft Computing.* Morgan Kaufmann, San Mateo, CA, 1993.