Knowledge Discovery in Databases Processing Using Improved Data Mining Techniques

A Thesis Submitted in Partial Fulfillment of the Requirements for the Award of the Degree of

Doctor of Philosophy
in
Computer Science and Applications

By

Sanjeev Manchanda
Registration No. 9041454

May 2010

SCHOOL OF MATHEMATICS AND COMPUTER APPLICATIONS
THAPAR UNIVERSITY
PATIALA-147 001 (INDIA)

Knowledge Discovery in Databases Processing Using Improved Data Mining Techniques

Ph. D. Thesis

Submitted by

Sanjeev Manchanda

Registration No. 9041454

May 2010

SCHOOL OF MATHEMATICS AND COMPUTER APPLICATIONS
THAPAR UNIVERSITY
PATIALA-147 001 (INDIA)

Dedication

This thesis is dedicated to

Late Prof. H. S. Kasana,

whose guidance motivated me to work hard and sincerely, without compromising the quality of work. He will always be alive in my memories and will always motivate me to move ahead with same spirit.

Acknowledgements

First and foremost, I would like to thank my two supervisors, Dr. S. B. Singh and Dr. Mayank Dave, for their support and guidance during the progress of present research work and while writing this Ph.D. thesis. I would also like to thank Ph.D. Doctoral committee members, Prof. Sushil Mittal, Prof. S. S. Bhatia, Prof. Seema Bawa, Dr. Rajesh Kumar and Dr. A. K. Lal, whose comments and advice helped to improve the quality of this thesis. I would also like to thank to the former members of Doctoral committee, Prof. K. K. Raina and Prof. R. K. Sharma for their valuable suggestions in the development of this research work.

I would also like to acknowledge the colleagues from university for their support in all respects.

I would like to extend a very special thanks to people around the globe, who contributed directly or indirectly in the development of present research work.

Lastly, and most importantly, I acknowledge that it is impossible for me to express in words alone the full extent of my appreciation and gratitude owed to my family for their unconditional love and support, and their many willing sacrifices made, so that I could successfully complete my Ph.D.

Sanjeev Manchanda
Thapar University, INDIA,
May, 2009.

Declaration

I hereby declare that the work presented in this thesis was carried out by me, except as acknowledged in the customary manner. It has not been presented previously for any degree, nor is it at present under consideration by any other degree awarding body.

3/5/2010

(Sanjeev Manchanda)
Thapar University, INDIA

I hereby certify that above statement is true to the best of my knowledge and belief.

(Dr. S. B. Singh)
Deptt. of Mathematics,
Punjabi University, Patiala.

(Dr. Mayank Dave)
Deptt. of Computer Engg.,
NIT, Kurukshetra.

Abstract

This thesis focuses on problems related to supervised learning for data mining and system development for knowledge based systems. These problems have been analyzed and an effort have been made to find solutions for the problems related to earlier processes and models for supervised learning and knowledge based system development respectively.

Initially this research work concentrates upon investigating past developments in the areas of data mining, supervised learning as well as software and knowledge engineering to find the past developments through literature survey. Then this research work concentrates upon enumerating different processing methods used for supervised learning. These different processes are devised through a general Knowledge Discovery in Databases (KDD) process. From this general process, six specific processes were identified to be widely practiced worldwide. After investigating different processes for supervised learning, limitations of these methods are identified. After identifying problems related to different supervised learning processes, investigations are directed towards finding the solutions for these problems. Search for finding solutions of these problems motivated to develop a new process for supervised learning. This motivation resulted in the form of a new process named as Fuzzy Boundaries of Regression Based Clusters process.

Proposed process is based on parallel processing of classification as well as regression algorithms. Proposed process allows pruning the training data without compromising the performance of outcome. It reduces the size of the train set significantly, while analyzing each record of data qualitatively as well as quantitatively through classification and regression algorithms respectively. Proposed process as well as previously known processes are applied on twenty classification algorithms and five regression algorithms over ten datasets gathered from internationally renowned organizations. After investigating six previously used processes and a new proposed process for supervised learning, question arose about suitability of previously known system development models for knowledge based system that involves the processing of two or more of these processes simultaneously. All previously known models are investigated for this purpose, but none of

the system development model is recognized to be satisfactory for such a dynamic knowledge based system development.

This necessity to find satisfactory model led to development of model for developing and maintaining dynamic knowledge based systems. This led to the construction of 'Genetic Information System Development and Maintenance' model. This model postulates the need of creating and maintaining a team of software developers within the organization for whom system is to be developed. Further this model advocates the need of system development and maintenance activities to run simultaneously throughout the life of an organization. Different advantages of proposed model are enumerated. Proposed process for supervised learning and model for knowledge based system is investigated through quantitative evaluation. So, a case study was used to hypothesize and prove different aspects of this model.

Massive experimentation work is performed for evaluating and comparing previously known supervised learning processes as well as proposed process. Experimentation setup is divided into two sub categories viz. Major Study and Minor Study. Major study includes twenty classification algorithms, five regression algorithms and ten datasets for evaluating eight processes, whereas Minor study includes five classification algorithms, five algorithms and ten datasets for evaluating eleven processes. Results of proposed and previously practiced processes are compared through tables and graphs. 'Genetic Information System Development and Maintenance' model is evaluated through a case study of Pepsi Foods Ltd., India. Proposed model is implemented in this organization and data was gathered through this implementation. Gathered data is compared with data collected from earlier system development of same organization. Hypothesized results have been presented for this model.

Contents

Front Page i
Dedication ii
Acknowledgements iii
Declaration iv
Abstract . . . . . v-vi
Contents . . . . . vii-xiii
List of Figures . . . . . xiv-xv
List of Tables xvi

Chapter 1 Introduction..... 1-7

1.1 Motivation 3
1.2 Objective of Present Research 4
1.3 Scope of the Thesis 4
1.4 Approach 5

1.4.1 Experimentation 5
1.4.2 Case Study 6

1.5 Contributions 6

1.5.1 Contribution to Data Mining 6
1.5.2 Contribution to Software and Knowledge Engineering 6

1.6 Thesis Organization 7

PART I 8-34

Chapter 2 Literature Review..... 9-24

2.1 Data Mining 10
2.2 Supervised Learning 18
2.3 Software Engineering and Knowledge Engineering 21
2.4 Conclusion 24

Chapter 3 Supervised Learning Processes..... 25-34

3.1 Various Processes for Supervised Learning 25

3.1.1 Simple Supervised Learning 25
3.1.2 Preprocessing of the data 26

3.1.2.1 Attribute Transformation 26
3.1.2.2 Data Sampling 27
3.1.2.3 Validation. 27

3.1.3 Post processing of the knowledge derived 28

3.1.3.1 Calibration 28
3.1.3.2 Thresholding 29

3.1.4 Stacking 30
3.1.5 Complex Processing 31

3.2 Description of Data Mining Techniques and Algorithms used for study 31

3.2.1 Decision Trees 31
3.2.2 Support Vector Machine 31
3.2.3 Genetic Algorithms 32

3.2.4 Neural Networks 32
3.2.5 K-nearest neighbor 32
3.2.6 Rule Induction 33
3.2.7 Logistic Regression 33
3.2.8 Bayesian Approach 33
3.2.9 Boosting/Bagging 33

3.3 Conclusion 34

PART II 35-65

Chapter 4 Supervised Learning Through Fuzzy Boundaries of Regression Based Clusters..... 36-50

4.1 Critical Analysis of Previously known Preprocessing and Post Processing Methods 36
4.2 Conceptual Foundation for Proposed Method 38

4.2.1 Supervised Learning and Classification 38
4.2.2 Regression Analysis 38
4.2.3 Fuzzy sets 39
4.2.4 Clustering 39

4.3 Issues/Problems related to Supervised Learning, Classification, Regression and their Fusion 40

4.3.1 Issues/Problems Related to Supervised learning processes . . 40
4.3.2 Issues/Problems related to Classification of data 40
4.3.3 Issues/Problems related to Classification through Regression based algorithms 40
4.3.4 Issues/Problems related to Fusion of Classification and Regression Algorithms 41

4.4 Conceptual Foundation of Fuzzy Boundaries of Regression Based Clusters 41
4.5 Proposed Improved Supervised Learning Algorithm based on fuzzy boundaries of regression based clusters 45
4.6 Composite effect of Classification algorithms and Regression based Clusters 48

4.6.1 Data common to Correctly Classified and under strong confidence range through regression 48
4.6.2 Data common to misclassified instances 48
4.6.3 Unique data from misclassified instances 48
4.6.4 Rightly classified data but with Poor confidence 49
4.6.5 Strongly known but poorly defined by regression 49
4.6.6 Other shifted data 49
4.6.7 Issue related to eliminated data 49

4.7 Conclusion 50

Chapter 5 'Genetic Information System Development and Maintenance' Model..... 51-65

5.1 Relationship between Organization and Information system 52
5.2 'Genetic Information System Development and Maintenance' Model 53

5.2.1 Team Organization for System Development and Maintenance 53
5.2.2 Development and Maintenance of the system 54

5.3 Proposed model for system development 59
5.4 Effectiveness of proposed model 63
5.5 Conclusion 65

PART III..... 66-140

Chapter 6 Experimentation..... 67-82

6.1 Experimental Setup for Supervised Learning Processes 67

6.1.1 Datasets 68
6.1.2 Processes Used for Experimentation 70

6.1.2.1 Fixed Split 70
6.1.2.2 Cross Validation 71
6.1.2.3 Calibration Methods 71
6.1.2.4 Fuzzy Boundaries of Regression Based Clusters . . . 72

6.1.3 Algorithms used for Experimentation 73

6.1.3.1 Classification Algorithms 73
6.1.3.2 Regression Algorithms. 76

6.1.4 Metrics for evaluation 77
6.1.5 Bootstrap analysis for ranking of problems/metrics 79

6.2 Case Studies for 'Genetic Information System Development and Maintenance' Model 80
6.3 Approach for the evaluation of 'Genetic Information System Development and Maintenance' Model 81
6.4 Conclusion 82

Chapter 7 Results, Interpretations and Evaluation..... 83-130

7.1 Experimental Results of Supervised Learning Processes 83

7.1.1 Major Study 84

7.1.1.1 Performance by Problem 84
7.1.1.2 Percentage of Datasets Used for Proposed Model . . . 85
7.1.1.3 Performance by metrics 85
7.1.1.4 Critical analysis of Results generated from supervised learning processes of Major study 96

7.1.2 Minor Study 105

7.1.2.1 Performance by Problem 106
7.1.2.2 Performance by metrics 106
7.1.2.3 Critical analysis of Results generated from supervised learning processes of Minor study 111
7.1.2.4 Comparison of Performance from Regression Algorithms of Fuzzy Boundaries of Regression Based Clusters process with Stacked Algorithms 113

7.1.3 Graphical Comparison 115

7.1.3.1 ROC Curves 115
7.1.3.2 Precision/Recall Curves 115

7.2 Hypothesis Testing for ‘Genetic Information System Development and Maintenance’ Model 122

7.2.1 Reliability testing through numbers of faults received after initial implementation of information systems 122
7.2.2 Reliable corrective maintenance testing through action taken within same week against the faults received after initial implementation of information system 124
7.2.3 Testing the difference in cumulative costs for the systems based on other models and proposed model 125
7.2.4 Comparative analysis to check the effect of proposed model over software reuse 127
7.2.5 Generalized comparison of proposed model with other models 128

7.3 Conclusion 129

Chapter 8 Conclusions and Future Research..... 131-140

8.1 Summary of Present Research 131
8.2 Contributions of Present Research 134
8.3 Limitations of Present Research 136
8.4 Future Research Work 137
8.5 Conclusion 139

References 141-159
List of Publications 160-161
Annexure A: Glimpses of Datasets 162-164
Appendix B: List of Abbreviations and Symbols 165-167

List of Figures

Figure 1.1: The general KDD process 2
Figure 3.1: Simple Supervised Learning Process 26
Figure 3.2: Discretized Supervised Learning Process 27
Figure 3.3: Cross-Validated Supervised Learning Process 28
Figure 3.4: Post-Processed Supervised Learning Process 29
Figure 3.5: Stacked Supervised Learning Process 30
Figure 4.1: Ideal Results for Classification 38
Figure 4.2: Actual Classification Results 38
Figure 4.3: Frequency chart for Class Probability Estimate (CPE) of Classification algorithm 42
Figure 4.4: Frequency Chart from Continuous outcome from regression algorithm . . 42
Figure 4.5: Combined Frequency charts of Class Probability Estimate and Continuous Outcomes of Classification and Regression algorithms respectively . . . 43
Figure 4.6: Center of Regression output tilted to the left of Classification Output . . . 44
Figure 4.7: Center of Regression output tilted to the right of Classification Output . . 44
Figure 4.8: Flowchart for fuzzy boundaries of Regression Based clusters processing . 47
Figure 5.1: Flow Chart for Genetic Information System Development and Maintenance Model 58
Figure 5.2: Accessing User account through Internet or Intranet 60
Figure 5.3: Connectivity of user to database through process/ sub process 61
Figure 5.4: Transition of processes from current generation to next generation 62
Figure 7.1: Relationship of size of models with increasing sizes of datasets 98
Figure 7.2: Sizes of models from all algorithms for different processes 99
Figure 7.3: Performance of Different algorithms for different processes 99
Figure 7.4: Performance of different Processes for different algorithms 99
Figure 7.5(a and b): Performances of Individual algorithms for different processes over different datasets/problems 101-102
Figure 7.6: Performance of Different algorithms for different processes 111
Figure 7.7: Performance of different Processes for different algorithms 112
Figure 7.8(a and b): Performances of Individual algorithms for different processes over different datasets/problems 112-113

Figure 7.9: Comparison of performances from different algorithms from different Fuzzy theory based processes and from stacking of same algorithms 114
Figure 7.10: ROC graphs of twenty algorithms for Adult problem (X Axis-False Positive Rate, Y-Axis True Positive Rate) 116
Figure 7.11: Precision/Recall graphs of twenty algorithms for Adult problem (X Axis-Recall, Y Axis-Precision) 116
Figure 7.12(a, b and c): ROC graphs of twenty algorithms from all five fuzzy processes for Adult problem (X Axis-False Positive Rate, Y-Axis True Positive Rate) 117-119
Figure 7.13(a, b and c): Precision/Recall graphs of twenty algorithms from all five fuzzy processes for Adult problem (X Axis-Recall, Y Axis-Precision) 119-121
Figure 7.14: Reliability Trend: Faults received per week after initial implementation . . 123
Figure 7.15: Dealing with failures: Faults Received and Action taken in Same Week . 124
Figure 7.16: Percentage of faults corrected within same week of identification 125
Figure 7.17: Comparison of System Development and Maintenance Costs 126
Figure 7.18: Comparison of Cumulative Costs 126
Figure 7.19: Comparison of Software Reused in Percentage of Lines of Code reused 127

List of Tables

Table 6.1: Description of Problems/Datasets Used for present research 70

Table 6.2: A contingency table for a binary class problem 78

Table 7.1(a, b and c): Accuracy of all algorithms over ten problems and their mean performances in descending order 86-88

Table 7.2(a and b): Percentage of the train set used for generating results from Fuzzy boundaries of regression based clusters process 89-90

Table 7.3(a, b, c, d and e): Average performances for each learning algorithm by metric (average over ten problems) 91-95

Table 7.4: Sizes of models in Kilobytes for different algorithms with increasing sizes of the train set 97

Table 7.5: Sizes of models in Kilobytes for different algorithms over different processes 98

Table 7.6: Bootstrap Analysis of Overall Rank by Mean Performance across Problems and Metrics 103

Table 7.7: Comparison of previous and current study through Average Accuracy . . 107

Table 7.8: Accuracy of algorithms from minor study over ten problems and their mean performances in descending order 108

Table 7.9(a and b): Average performances for each learning algorithm of minor study by metric (average over ten problems) 109-110

Table 7.10: Comparison of results from Fuzzy process and stacking processes . . . 114

Table 7.11: Comparison of different models on the basis with cost, reliability, maintainability and adaptability of information systems 128

Table B.1(a and b): List of Abbreviations used in this research 165-166

Table B.2: List of Symbols used in this research 167

Chapter 1

Introduction

Knowledge discovery in databases (KDD) is the theme of many discussions for last two decades. A large number of techniques and algorithms have been developed for mining the knowledge from large databases. The techniques used earlier to search data in response to queries having fixed domains has undergone substantial change with searching unknown patterns or vaguely defined queries.

A lot of research efforts are performed around the world to find unknown facts hidden in millions of records, stored in huge databases. As databases are growing in size, there is a need to explore new patterns from existing database(s) and there is a need to develop new techniques that are efficient enough to find such patterns and should also be able to fulfil future needs of the companies demanding such solutions. There are different perspectives, why we search for such new patterns e.g. sales/marketing, customer retention, buyer behaviour, cost/utilization, quality control, inventory and fraud detection *etc.* Most important among them is marketing department(s) of the organizations, who are always looking for such patterns, which can furnish new prospects for marketing managers to think upon new trends, aspects and emerging scopes to take decisions at different managerial levels. Data mining techniques can be categorized in different ways. One of the most important

categorization is based on kind of problems solved. Classification, Clustering and Association rules are most important kind of problems that are solved by data mining. Present work concentrates upon classification tasks. A large number of techniques and algorithms have been devised for extracting knowledge from databases. A large part of these techniques and algorithms concentrate upon classification problems. Classification task comprises of assigning objects to classes (groups) on the basis of measurements made on the objects. Classes of data are predefined, a set (the train set) of labeled objects are used to form a model through classifier for classification of future observations (or the test set). Such a processing is named as supervised learning. Supervised learning techniques are usually used for the solution of classification problems. Usually a general process (Fayyad et al. [53]) is recommended for supervised learning as shown in Figure 1.1. But practical implementation of a general process becomes difficult, when we need to implement this general process for some specific problem solving.

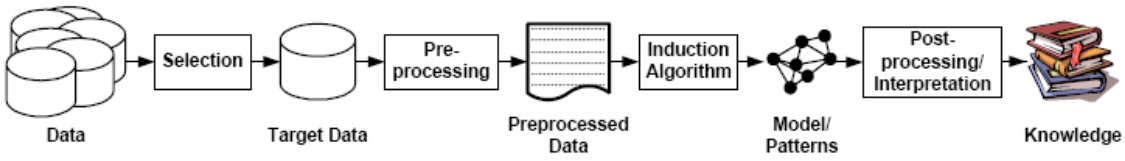

Figure 1.1: The general KDD process

Supervised learning processes can vary from simple to very complex processing. Different techniques and algorithms are used to extract knowledge from data. These algorithms involve certain parameters and procedures to extract knowledge. Different techniques and algorithms are suitable for different types of problems. There is no unique technique/algorithm to solve all types of problems. Enormous efforts have been made to develop and implement such techniques to furnish required knowledge. There is a need to analyze these developments critically and to enhance the achievements and to remove pitfalls and to design such processes, which can improve these shortcomings significantly. Future expectations are very high from these techniques. This is mainly due to the reason that databases are massively increasing in size with time and so is the need for information requirements, along with expectations for better results. Present research work is an effort to explore relationship between types of problems with specific technique/algorithm as well as with type of processing required for extracting knowledge, finding improved processing procedure(s) and for finding the way to handle the processing of these processes in

information system development environment. A variety of datasets were collected from a variety of internationally renowned organizations. These datasets were used for drawing results for previously known processes for supervised learning. Thereafter a new process has been proposed for supervised learning. Results of Proposed process are compared with the results generated from previously known processes. Results are compared for fifteen different metrics out of the results generated for twenty three metrics. Results are also compared with the available benchmark results of the problems involved for study. One model has also been proposed for developing information/knowledge based system for knowledge extraction.

1.1 Motivation

There are several algorithms and processes that are used for supervised learning. Problem arises with finding a suitable algorithm and process for extracting knowledge for a problem at hand. This type of dilemma motivated us to analyze the environmental factors that affect the selection of a suitable algorithm as well as process of supervised learning and to handle potential issues involved in such processing.

Another issue is handling these processes for supervised learning in a real life system development environment. This motivated us to find solution to the problem of processing many knowledge extraction processes and to maintain them suitably. Knowledge extraction is a task of understanding problem domain and to explore new knowledge. For creating knowledge extraction system, usual system development strategies are not that impressive due to the fact that knowledge extraction is largely dependent upon domain knowledge of problem. A lot of dynamism is involved in knowledge processing. Technical expertise is very much required for any system development. Above that, knowledge extraction requires domain expertise and experience based on knowledge extraction tasks. This motivated us to find a suitable strategy to develop knowledge based systems by analyzing the problems with earlier system development strategies.

This research work, first of all, focuses on analyzing different processes of supervised learning. Secondly, it proposes a new improved process for developing knowledge extraction. Massive experimentation work is carried out to compare the results obtained from

all processes. Thirdly, this research work focuses on developing knowledge based system development process.

1.2 Objective of Present Research

Objectives of this thesis are stated below:

“To propose new algorithms that will be guided by information requirements and suitable database(s) to achieve improved KDD (Knowledge Discovery in Databases) processing”.

To justify the objective of thesis, different processes for supervised learning will be analyzed and new process shall be proposed with experimental results. Above that, a model will be proposed for developing knowledge based systems capable of handling dynamism involved in developing knowledge based systems.

1.3 Scope of the thesis

Data mining is the subject area of present thesis. This thesis primarily includes investigations of supervised learning techniques and algorithms. Classification tasks are included in the majority of applications of supervised learning. This thesis concentrates upon classification tasks through supervised learning. We have included applications from various domains for investigations. Different processes practised for supervised learning are investigated closely and experimented under current environment. After analyzing limitations of earlier processes a new process is proposed. Experiments were performed to investigate different processes of supervised learning used globally as well as for the proposed process. This thesis also investigates different system development models used worldwide and proposes a new model for the development of knowledge based systems. Different important attributes of new models are hypothesized and are proven through quantitative as well qualitative parameters. This thesis presents and evaluates results with various metrics and through different tabular as well as graphical representations.

1.4 Approach

This thesis is fundamentally an empirical study. Massive experimentation work is performed for comparing different processes of supervised learning. It also includes a case study based on quantitative as well as qualitative data. Case study is used for model that is proposed for developing knowledge based systems and is justified with hypothesized results.

The word experiment is generated from Latin word *ex-periri*, which means "to try out". An experiment is defined, in science, as a method of investigating less known fields, solving practical problems and proving theoretical assumptions. An experiment (Kish [113]) can be thought of as a specific type of method used in scientific inquiries, and personal questioning, usually to study causality. Often the objective is to test a hypothesis: i.e. a tentative explanation of a phenomenon or mechanism of causality. The essence of an experiment is to introduce a change in a system (the independent variable) and to study the effect of this change (the dependent variable).

The case study (Stake [176]) can be characterized as a study of the specific case to learn something. The method applied is to define a “topic of concern”, identify the “case” to be studied, investigate how the “topic of concern” relates to the specific “case” and on this background make assertions about the “topic of concern”. Rigorously performed, one can state hypothesis about the “topic of concern”, one can design the investigation, gather data in a structured way and can evaluate the hypothesis.

Experimentation and case study used in this thesis are described as follows:

1.4.1 Experimentation

Present study compares supervised learning processes on ten binary classification problems. Data is collected from internationally renowned organizations. Datasets are applied on seven different processes. Six processes are used for experimentation worldwide and seventh process is contributed by this thesis. Results generated from these experiments are compared for different metrics based on confusion matrix.

1.4.2 Case Study

Case study used in this thesis is based on a renowned soft drink company. As soft drink industry is marketing operations intensive, its information requirements are highly dynamic. Changing information needs has created a lot of difficulty for the software developers, because maintenance costs for software rise beyond estimates and cause unreliability even after spending sufficient amounts on the system development and maintenance. Earlier software development methodologies and models have not been able to meet the dynamic information requirements. Present thesis introduces a model that is used for developing system and helps improving the complexities involved in system development and maintenance.

1.5 Contributions

Contributions of this thesis can be categorized into following two categories.

1.5.1 Contribution to Data Mining

In this thesis different processes used worldwide for supervised learning, were investigated and experiments were performed for ten different problems. An attempt was made to investigate different factors related to these problems as well as different processes based on different metrics generated from massive experimentation. Present research work proposes a new process for experimentation. Results of proposed process are compared with results of earlier processes.

1.5.2 Contribution to Software and Knowledge Engineering

This thesis also investigates the problems related to knowledge based system development and presents a model for system development. Different aspects of this model are hypothesized and proven quantitatively.

1.6 Thesis Organization

The remaining contents of this thesis are organized in three parts.

The first part is composed of Chapter 2 and Chapter 3. Chapter 2 includes literature review of related work. Chapter 3 includes detailed investigations of previously known processes for supervised learning.

The second part is composed of Chapter 4 and Chapter 5. Chapter 4 proposes a new process for supervised learning with its causes, effects and justifications. Chapter 5 proposes a new model for system development and maintenance and justifies its suitability for knowledge based system development and maintenance.

The third part is composed of Chapter 6, Chapter 7 and Chapter 8. Chapter 6 includes the details of experimental setup. Chapter 7 includes experimental results and their evaluations. It also compares the results with results generated from current environment as well as benchmark results. Chapter 8 concludes the research work included in this thesis with future directions.

PART I

Chapter 2 **Literature Review 9-24**

Chapter 3 **Supervised Learning Processes …… 25-34**

Chapter 2

Literature Review

Though data mining is the evolution of a field with a long history, the term itself was only introduced relatively recently, in the 1990s. Data mining roots are traced back along three family lines viz. statistics, artificial intelligence and machine learning. Data mining, in many ways, is fundamentally the adaptation of machine learning techniques to business applications. It is best described as the union of historical and recent developments in statistics, artificial intelligence and machine learning. These techniques are then used together to study data and find hidden trends or patterns. Data mining is finding increasing acceptance in science and business areas that need to analyze large amounts of data to discover trends, which they could not otherwise find. In present study literature was reviewed in three major directions. All three directions are related to each other but can be distinguished according to their nature of study. First of all, literature was reviewed in the direction of data mining and is presented as general review of development in the area. Section 2.1 investigates the evolution of data mining field. Secondly, previous developments were investigated related to supervised learning. As present work concentrates upon supervised learning, so supervised learning was reviewed more closely. Section 2.2 investigates the developments related to supervised learning. Thirdly, different information development models were investigated as present

work concentrated upon developing a model suitable for knowledge based system development. Section 2.3 reviews the developments related to previously know information system development models.

2.1 Data Mining Evolution

Data mining term is known to be coined first time when Piatetsky [157] emphasized on the need of data mining due to growth in amount of databases. He introduced the idea of Knowledge Discovery in Databases and sated that time had come for mining databases. Han et al. [88] emphasized on concept based classification in Relational database. He devised a method for the classification of data in relational databases by concept-based generalization. Bocca [22] discussed the design and engineering of the enabling technologies for building Knowledge Based Management Systems (KBMS). He demonstrated the solutions of existing technologies commercially available *i.e.*, relational database systems and logic programming. Aggarwal et al. [1] discussed the problems occurring while classifying data populations and samples into 'm' group intervals. This study proposed a tree based interval classifier, which generated a classification function for each group that could be used to efficiently retrieve all instances of the specified group from the population database. Frawley et al. [61] gave an overview on the need of concentration of artificial intelligence society on knowledge discovery in databases. They presented the issues involved in development of data mining concept and its need. Han et al. [89] devised an attribute-oriented concept tree ascension technique, which was applied in generalization and substantially reduced the computational complexity of database learning processes. They showed that their method substantially reduced the computational complexity of the database learning processes. Hearst [91] emphasized on the need of in-depth information availability by the text based intelligent systems by avoiding the complexity and resource-consuming behavior of detailed text understanders. He devised a method for a general class of queries without engaging the complexity required by natural language processing techniques that attempt to generate "all plausible" inferences.

Piatetsky et al. [156] in a workshop on KDD at Washington summarized the increasing interest and researches in the area of knowledge discovery in databases and explored on-going work in the field and the systems developed for this purpose. Han et al. [85] emphasized on the need of knowledge discovery of databases in object-oriented and active databases. They discussed the extensions of mechanisms for knowledge discovery in relational databases towards new generation databases. Agarwal et al. [3,

4] raised the issue of performance in data mining as the confluence of machine learning and its emphasis on database technology due to problems involving classification, associations and sequences and emphasized on finding association rules in large databases. This study presented solution by using a model and four basic solutions e.g., generate and measure, combine, filter and select and presented an algorithm for finding rules. Mobasher et al. [146] presented a logic programming language, which used a four-valued bi-lattice as the underlying framework for semantics of programs. Ng et al. [149] tried to explore whether clustering methods had a role to play in spatial data mining. They developed a clustering algorithm called CLARANS, which is based on randomized search and two data mining algorithms Spatial Dominant Approach (SD) and Non-Spatial Dominant Approach (NSD) and showed the effectiveness of algorithms. Mannila et al. [137] revised the solution to the problem raised by Agarwal et al. [3] and proposed an algorithm to improve the solution. Han et al. [79] studied the construction of Multi Layered Databases (MLDBs) using generalization and knowledge discovery techniques and the application of MLDBs to cooperative/intelligent query answering in database systems. They proposed an MLDB model and examined in their study and showed the usefulness of MLDB in cooperative query answering, database browsing and query optimization.

Holsheimer et al. [92] surveyed the data mining researches of that time and presented the main underlying ideas behind data mining such as inductive learning, search strategies and knowledge representations used in data mine systems. This study also described important problems and suggested their solutions. Kawano et al. [108] integrated knowledge sampling and active database techniques to discover interesting behaviors of dynamic environments that react intelligently to the environment changes. This study showed firstly that data sampling was necessary in the collection of information for regularity analysis and anomaly detection. Secondly, knowledge discovery was suggested to be important for generalizing low-level data to high-level information and detecting interesting patterns. Thirdly, active database technology was felt essential for the real time reaction to the changes in real time environment, it was concluded that the integration of three technologies forms a powerful tool for control and management of large dynamic environments in many applications. Zaïane et al. [196] proposed multiple layered database approach to handle the resource and knowledge discovery in global information base. The MLDB approach promoted a tight integration of database and data mining technologies with resource and knowledge discovery in global information systems. Cheung et al. [35] extended their concept of non-rule-based hierarchy introduced by Bocca [22] of knowledge representation to a rule-based concept

hierarchy. This study developed the rule-based attribute-oriented induction algorithm to facilitate learning with a rule-based concept graph. Han et al. [80] developed a prototyped data mining system called DBLearn, which could extract different kinds of knowledge rules from relational databases. Agrawal et al. [5] considered the problem of discovering association rules between items in a large database of sales transactions. They proposed two algorithms Apriori and AprioriTid for discovering all significant association rules between items in a large database of transactions and compared them with earlier available algorithms with better experimental results. Mobasher et al. [147] presented an operational semantics for bilattice based logic programs, which uses the join irreducible elements of the knowledge part of bilattice, as its basis. This study showed that restricting attention to the join-irreducible elements could reduce the overall complexity of the inference systems. Han et al. [81] stated that their research covered a large wide spectrum of knowledge discovery, which included the study of knowledge discovery in relational, object oriented, deductive, spatial, active databases and global information systems and development of various kinds of knowledge discovery methods including attribute-oriented induction, progressive deepening for mining multiple level rules, meta-guided knowledge mining *etc.* and also studied algorithms for the techniques of data mining.

Han et al. [78, 87] emphasized on mining knowledge at multiple levels, which could help database users to find some interesting rules, which were difficult to be discovered otherwise and view database contents at different abstraction levels and from different angles. They proposed many techniques and developed DBMiner (a revision of DBLearn [80]). Holsheimer et al. [93] discussed the use of database methods for data mining and studied the degree to which one could manage using database management systems. Han et al. [77] described an attribute-oriented induction technique for discovery of data evolution regularities in relational databases. They showed that attribute-oriented induction substantially reduced the computational complexity of the knowledge discovery process. Agrawal et al. [2] introduced an active data mining paradigm, in which data was continuously mined at a desired frequency, rules were added or updated as these were generated or some change in history of parameters was found. Park et al. [154] examined the issue of mining association rules among items in large database of sales transactions. They designed an algorithm and conducted extensive simulation to evaluate the performance of the proposed algorithm. Bayardo et al. [16] analyzed the effects of polynomial-space bounded learning on runtime complexity of backtrack search. They presented and theoretically evaluated backtrack based algorithms for tractable constraint satisfaction on structurally restricted instances. Han et al. [82] designed data

mining query language, DMQL, for mining different kinds of knowledge in relational databases. They implemented the portions of the language in their software DBMiner for interactive data mining. Cheung et al. [33, 34] discussed relationship between locally large and globally large itemsets and proposed a distributed association rule mining algorithm FDM (Fast Distributed Mining Of Association Rules) and showed better performance over direct application of a typical sequential algorithm. Kohavi et al. [115] focused on classification algorithms and reviewed the need for multiple classification algorithms and designed a system called MLC++, which was designed to help choose the appropriate classification algorithm for a given dataset by making it easy to compare the utility of different algorithms on a specific dataset of interest. Chen et al. [32] surveyed data mining techniques from database researcher's point of view and provided a classification and comparative study of these techniques. Cheung et al. [36] proposed algorithm for mining association rules for distributed databases named DMA. They tested this algorithm on an experimental test bed and studied its performance, which was found to be superior when compared with direct application of popular sequential algorithm.

Klemettinen et al. [114] proposed a methodology for knowledge discovery in databases. In this study, a large collection of patterns was discovered at once and also retrieved subset of collection of patterns. Micheline et al. [142] addressed to the efficiency and scalability issues by proposing a data classification method, which integrated attribute-oriented induction, relevance analysis, and the induction of decision trees. This kind of integration lead to efficient, high-quality, multiple-level classification of large amounts of data, the relaxation of the requirement of the perfect train sets and the elegant handling of continuous and noisy data. Micheline et al. [141] designed approach called Meta rule-Guided Mining under where users could probe the data under analysis by specifying the hypothesis in the form of meta rules or pattern templates. Mannila et al. [136] showed strong connections between the verification problem and the hyper graph transversal problem. Shrikant et al. [174] considered the problem of integrating constraints that are Boolean expressions over the presence or absence of items into the item discovery algorithm. They proposed three algorithms for mining association rules with item constraints and discussed their trade-offs. Dimitrios et al. [45] showed that the problem of finding maximally specific sentences that are interesting in a database had close relationship with the hypergraph transversal problem. This study analyzed two algorithms earlier used in data mining and proved upper bound on their complexity. Mobasher et al. [145] proposed a method for clustering of data in a high dimensional space based on a hypergraph model. The hypergraph model mapped the relationship present in the original data in high dimensional space into a hypergraph. Han et al. [83] integrated data

cube and Apriori Data mining techniques for mining segment wise periodicity with regard to a fixed length period and showed that data cube could provide an efficient structure and a convenient way for interactive mining of multiple-level periodicity.

Han et al. [86] investigated the issues on generalization-based data mining in object oriented databases in three aspects generalization of complex objects, class based generalization and extraction of different kinds of rules. Their studies showed that set of sophisticated generalization operators could be constructed for generalization of complex objects, dimension based class generalization mechanism could be developed for class based generalization and sophisticated rules formation method could be developed for extraction of different kinds of rules. Han et al. [84] Integrated both constraint-based and multidimensional mining into one Framework that provided an interactive, exploratory environment for effective and efficient data analysis and mining. Fu et al. [65] developed a top-down progressive deepening method for efficient mining of multiple-level association rules from large transaction databases based on the Apriori principle. They developed a group of algorithm and showed that efficient algorithms can be developed from large databases for the discovery of interesting and strong multiple-level association rules. Garofalakis et al. [66] proposed the use of Regular Expressions (REs) as a flexible constraint specification tool that enables user-controlled focus to be incorporated into the pattern mining process. They developed a novel family of algorithms termed SPIRIT (Sequential Pattern Mining with Regular Expression Constraints) for mining frequent sequential patterns that also satisfied user-specified RE constraints. The solutions provided valuable insights into the tradeoffs that arose when constraints that do not subscribe to nice properties (like anti-monotonicity) were integrated into the mining process, which was conducted through an extensive experimental study on synthetic and real-life data sets. Toivonen et al. [185] introduced a new method for linkage disequilibrium mapping: haplotype pattern mining (HPM). The method was based on discovery of recurrent patterns. They defined a class of useful haplotype patterns in genetic case-control data and used the algorithm for finding disease-associated haplotypes. Dhar et al. [44] described and evaluated several variations of a new genetic learning algorithm (GLOWER) on a variety of data sets. They examined the performance of several GLOWER variants on two UCI data sets as well as on a standard financial prediction problem (S&P500 stock returns), used the results to identify one of the better variants for further comparisons.

Maniatty et al. [135] explored a migration path out of limited capacity and performance of data mining tools by considering an integrated hardware and software approach to

parallelized data mining. Their analysis showed that parallel data mining solutions required parallel data mining algorithms, parallel and distributed databases, parallel file systems, parallel input/output, tertiary storage, management of online data, support for heterogeneous data representations, security, quality of service and pricing metrics, which was surveyed with an eye towards an integration strategy leading to a complete solution. Han et al. [76] developed a new frequent pattern growth for mining frequent patterns without candidate generation. Divide and conquer philosophy was adopted for that method to project and partition databases based on discovered frequent patterns and grew such patterns to longer ones in projected databases. Zaki [199] presented efficient algorithms for the discovery of frequent itemsets, which forms the compute intensive phase of the Association rule discovery. They experimentally compared the new algorithms against the previous approaches and obtained improvements of more than an order of magnitude for its test databases. Garofolakis et al. [67] addressed the problem of constructing “simple” decision trees with few nodes that was easy for humans to interpret. They permitted users to specify constraints on tree size or accuracy and then built the ‘best’ that satisfied the constraints. They assured that final tree was easy to understand and had good accuracy.

Zaniolo et al. [201] proposed a technique where, by keeping histograms on attribute pairs, they achieved (i) a significant speed-up over traditional classifiers based on single attribute splitting, and (ii) the ability of building classifiers that use linear combinations of values from non-categorical attribute pairs as split criterion. Indeed, by keeping two-dimensional histogram. Zaki [198] presented pSPADE, a parallel algorithm for fast discovery of frequent sequences in large databases. The algorithm used to decompose the original search space into smaller suffix-based classes so that each class could be solved in main-memory using efficient search techniques and simple join operations. Kumar et. al. [123] demonstrated the necessity of a two-phase rule induction framework for learning effective rule based model with small dataset. Li et al. [130] proposed a new associative classification method, CMAR, *i.e.* Classification based on Multiple Association Rules. The method extended an efficient frequent pattern mining method, FP-growth, which involved to construct a class distribution-associated FP-tree, and could mine large database efficiently. Moreover, it could apply a CR-tree structure to store, retrieve mined association rules efficiently, and prune rules effectively based on confidence, correlation and database coverage. Kumar et al. [124] compared three representative categories of most popular boosting variants in the context of rare classes and results showed better results for two of these three variants. Gupta et al. [74] presented an overview of the field of KDD and data mining. They discussed various

techniques of data mining along with some related issues. Zaki et al. [197] discussed about Implementation of data mining ideas in high-performance parallel and distributed computing environments was thus becoming crucial for ensuring system scalability and interactivity as data continues to grow inexorably in size and complexity. Evfimievski et al. [50] presented a framework for mining association rules from transactions consisting of categorical items where the data had been randomized to preserve privacy of individual transactions. They derived formulae for an unbiased support estimator and its variance, which allowed recovering itemset supports from randomized datasets and showed how to incorporate those formulae into mining algorithms. Mannila [138] discussed about emergence of data mining as an interesting area in the boundary between algorithms, probabilistic modeling, statistics and databases. He divided the data mining researches into two approaches. The first approach introduced was Global approach, which tried to model the whole data and the second one was local approach, which tried to find useful patterns occurring in the data. He discussed few examples of both, reviewed two attempts to combine both approaches and listed open problems with algorithmic flavor. Shasha et al. [172] proposed an algorithm for finding patterns in 3D graphs. They found patterns as a rigid substructure that could occur in a graph after allowing for an arbitrary number of rotations and translations as well as a small number of edit operations in the pattern or in the graph.

Kovalerchuk [118] introduced a hierarchical relational clustering method (Φ-method) applicable to discover hidden patterns in distributed heterogeneous textual databases and unstructured data. Clustering produced by the method was claimed to be invariant and had a clear meaning and natural visualization. Park et al. [153] proposed a new algorithm for sub dimensional clustering of high dimensional data, each of the three major steps of which partitioned the input points into several candidate clusters with proper numbers of points, filtered the clusters that could not be useful in the next steps and then merged the remaining clusters into the predefined number of clusters using a closeness function. The result of extensive experiments showed that the proposed algorithm exhibited better performance than the other existent clustering algorithms. Li et al. [129] proposed a similarity based distributed data mining(SBDDM) framework which explicitly took the differences among distributed sources into consideration. They introduced a new similarity measure and its effectiveness was then evaluated and validated. Zaniolo et al. [200] proposed a new extensibility mechanism for SQL-compliant DBMSs to support efficiently database-centric data mining applications and demonstrated its power in supporting decision support applications. Liu et al. [134] identified the key factors that influence the performance of the pattern growth approach

and optimized them to further improve the performance. Their algorithm used a simple while compact data structure ascending frequency ordered Prefix Tree (AFOPT) to store conditional databases, in which they used arrays to store single branches to further save space and AFOPT traversed structure in top-down depth-first order. Analysis and experiment results showed that it achieved significant performance improvement over previous works. Cheung et al. [37] handled a problem to mine N k-itemsets with the highest supports for k up to a certain k_{max} value and called the results the N-most interesting itemsets. They proposed two new algorithms, LOOPBACK and BOMO and Experiments showed that their methods outperformed the previously proposed Itemset-Loop algorithm, and the performance of BOMO could be an order of magnitude better than the original FP-tree algorithm, even with the assumption of an optimally chosen support threshold. Yip et al. [191] analyzed the major challenges of projected clustering and suggested why other algorithms needed to depend heavily on user parameters. Based on this analysis, they proposed a new algorithm that exploited the clustering status to adjust the internal thresholds dynamically without the assistance of user parameters. The results of extensive experiments on real and synthetic data, the new method was found to have excellent accuracy and usability. Kovalerchuk [119] demonstrated that neural networks to be useful when there a priori knowledge was not available about the analyzed data. They created a distributed computing architecture as well. Liu et al. [133] aimed to mine and to summarize all the customer reviews of a product. They performed the task by mining product features that have been commented on by customers, identified opinion sentences in each review and deciding whether each opinion sentence was positive or negative and summarizing the results. They proposed many techniques to perform these tasks.

Shasha et al. [171] extended the ideas of the most successful best match retrieval data structures, such as M-Tree, MPV-Tree, FQ-Tree, and List of Clusters, by using pivots based on the farthest pairs (Antipoles) in data sets. The resulting Antipole Tree was a bisector tree using pivot-based clustering with bounded diameter. Joshi et al. [106] proposed a polynomial time algorithm for combining phylogenetic trees, which makes use of least common ancestor information as optimality criterion. The algorithm satisfies the desirable properties of a phylogenetic supertree method and constructed a single phylogenetic supertree even for incompatible input trees, which was difficult to solve. Stasko et al. [177] identified the focus of current information visualization systems on representational primacy or the overriding pursuit of faithful data replication and comprehension. Gupta et al. [73] presented a user-centric model (I-MIN model) for KDD

processing and an architecture for a data mining management system based on it. This I-MIN system reinforced an iterative, interactive and operator-based approach to facilitate a dynamically changing search focus during exploration, at the end user level. Data mining has registered phenomenal growth in only two decades of its existence. With time need for better performance is increasing. Present research is another effort towards the contributions to this field.

2.2 Supervised Learning and Classification

Different studies have been conducted from time to time for accounting the development of supervised learning techniques and algorithms. Data mining itself has emerged from other disciplines like Machine Learning, Artificial Intelligence and Statistics etc. Hence, it is obvious to get initial references related to comparisons of algorithms from its parent disciplines. Many researches were being performed before the time data mining was coined as a separate discipline for study. Fahrmeir et al. [52] compared the results of a point awarding approach with the results obtained by the linear discriminant. Gorman et al. [72] reported that back-propagation outperformed nearest neighbor for classifying sonar targets, whereas Shadmehr et al. [170] showed that some Bayes algorithms were better on other tasks. Kirkwood et al. [112] developed a symbolic algorithm ID3, which performed better than discriminant analysis for classifying the gait cycle of artificial limbs. Breiman et al. [27] developed CART (Classification and Regression Trees) method and used it to analyze consumer credit granting. It concluded that CART had major advantages over discriminant analysis and emphasized the ability of CART to deal with mixed datasets containing both qualitative and quantitative attributes. However, on different tasks Spikvoska et al. [175] found that a higher order neural network (HONN) performed better than ID3 and Atlas et al. [11] showed that back-propagation did better than CART. Mitchell et al. [144] conducted a study for a coordinated comparison of many algorithms on the MONK's problem. Ripley [163] compared a diverse set of statistical methods, neural networks and a decision tree classifier, on the Tsetse fly data. After many small comparative studies, STATLOG by King et al. [111], is known as first comprehensive study that analyzed different data mining algorithms. LeCun et al. [127] compared several learning algorithms (including SVMs) on a handwriting recognition problem using three performance criteria: accuracy, rejection rate, and computational cost.

Cooper et al. [40] evaluated nearly a dozen learning methods on a real medical data set using both accuracy and an ROC-like metric. Lim et al. [132] performed an empirical comparison of decision trees and other classification methods using accuracy as the main criterion. Perlich et al. [155] conducted comparison between decision trees and logistic regression. Provost et al. [162] examined the issue of predicting probabilities with decision trees, including smoothed and bagged trees. Witten et al. [189] presented the comparison of different tools and techniques of data mining. Ferri et al. [55] presented an extension to the Area Under the ROC Curve in the form of the Volume. Under the ROC Surface (VUS), they showed the way to compute the polytope that corresponded to the absence of classifiers (given only by the trivial classifiers), to the best classifier and to whatever set of classifiers. They compared the VUS with “approximations” or “extensions” of the AUC for more than two classes. Johannes et al. [104] analyzed the most popular evaluation metrics for separate-and-conquer rule learning algorithms. Their results showed that all commonly used heuristics, including accuracy, weighted relative accuracy, entropy, Gini index and information gain, are equivalent to one of two fundamental prototypes: precision, which tries to optimize the area under the ROC curve for unknown costs, and a cost-weighted difference between covered positive and negative examples, which tries to find the optimal point under known or assumed costs.

Ronald et al. [167] introduced various distance measures. Some of these distance measures were metrics, while others were not. They identified a large and robust equivalence class of distance measures. Besides the applications to the task of identifying good notions of (dis)similarity between two top *k* lists, their results implied polynomial-time constant-factor approximation algorithms for the *rank aggregation problem* with respect to a large class of distance measures. Di-Rong et al. [46] provided a PAC error analysis for the *q*-norm soft margin classifier, a support vector machine classification algorithm that consisted of two parts: regularization error and sample error. They introduced a projection operator, which lead to better sample error estimates especially for small complexity kernels. Thomas et al. [182] presented an algorithm that analyzed interactions between various ROC dimensions, identifying independent classes, and groups of interacting classes, allowing the ROC to be decomposed. The resultant decomposed ROC hyper surface could be interrogated in a similar fashion to the ideal case, allowing for approaches such as cost-sensitive and Neyman-Pearson optimization, as well as the volume under the ROC. Thomas et al. [181] formalized multiclass ROC and exposed the computational complexities. A pair wise approach was proposed that approximated the multidimensional operating characteristic by discounting some interactions, resulting in an algorithm that was tractable, and extensible to large

numbers of classes. Two additional multiclass optimization techniques were also proposed that provided a benchmark for the pairwise algorithm and Experimental results demonstrated the efficacy of their pairwise approach.

Yair et al. [190] suggested a general approach based on a sequential learning model that utilized classifiers to sequentially restrict the number of competing classes while maintaining, with high probability, the presence of the true outcome in the candidates set. Jason et al. [103] compared Naive Bayes and Support Vector Machines on the task of multiclass text classification. Franc et al. [59] proposed a transformation from the multi-class SVM classification problem to the single-class SVM problem which was more convenient for optimization. Aik et al. [7] proposed a novel ensemble machine learning method that improved the coverage of the classifiers under the multi-class imbalanced sample sets by integrating knowledge induced from different base classifiers and illustrated their idea in classifying multi-class SCOP protein fold data. Tao et al. [179] discussed Discriminant analysis for learning discriminative feature transformations. This study investigated its use in multiclass classification problem s and tested performance on a large collection of benchmark datasets. Ting-Fan et al. [184] presented two approaches for obtaining class probabilities. They showed conceptually and experimentally that the proposed approaches were more stable than two existing popular methods voting etc. Hong et al. [94] discussed about fundamental power of microarrays lies in the ability to conduct parallel surveys of gene expression patterns for tens of thousands of genes across a wide range of cellular responses, phenotypes and conditions. Chen et al. [32] presented an approach among a number of strategies for multiple classifier combination is to calculate the beliefs of confidence based on confusion matrices from individual classifiers. To achieve precise belief computation, based on previous researchers' work, presented a better algorithm with a more generic capability and showed improved performance.

Zhong-Hui et al. [203] evaluated ensemble methods for creating ensemble classifier, such as Bagging and Boosting, etc. on two data sets. Eugene et al. [49] developed a novel multi-class classification method based on *output codes* for the problem of classifying a sequence of amino acids into one of many known protein structural classes, called *folds*. Fabio et al. [51] proposed multi-prototype SVM that extended multiclass SVM to multiple prototypes per class. They reported experiments on a number of datasets. Zhu et al. [204] proposed a new algorithm that naturally extends the original AdaBoost algorithm to the multiclass case without reducing it to multiple two-class problems. They provided a statistical justification for the new algorithm using a novel

multi-class exponential loss function and forward stage-wise additive modeling. Kun et al. [125] studied the problem of optimizing a multiclass classifier based on its ROC hypersurface and a matrix describing the costs of each type of prediction error. This study presented several heuristics for this problem, including linear and nonlinear programming formulations, genetic algorithms, and a customized algorithm. Their empirical results suggested that genetic algorithms fare the best overall, improving performance most often. Hsieh et al. [95] proposed a parametric linear feature extraction method for multiclass classification. It consisted of two types of schemes that were complementary to each other with regard to the discriminant information used. Matthew et al. [140] proposed automatically constructing ontologies, merging ontologies with different structures and optimal mechanisms for ontology building. In one other study, Bauer et al. [15] presented an impressive empirical analysis about different ensemble methods such as bagging and boosting. Recently, Caruana et al. [30] conducted a study to rank different many learning algorithms, based on binary problems. Present research work is dedicated to guide the ranking and selection of different supervised learning algorithms based on different metrics used to analyze all type of classification techniques and algorithms on a variety of problems and to compare the results with earlier studies. This research also proposes an improved process for supervised learning and compares results generated from earlier studies.

2.3 Software Engineering and Knowledge Engineering

In this section, the historical developments related to the System development models/methodologies, Process modeling, Change management, Reliability, Goal orientation in modeling and use of Internet etc. are examined. System development models/methodologies, Process modeling, Change management, Reliability etc. are clearly related to the domain of this paper, whereas goal orientation is essential for all the sections of any organization. Goal orientation is substantial for guiding organizational efforts in unique direction and information system should possibly be guided towards the organizational goals. Internet helps in developing globally operating systems. So Internet developments need to be revealed here.

The documented collection of policies, processes and procedures used by a development team or organization to practice software engineering, is known as software development methodology (SDM) or system development life cycle (SDLC). Development of system development methodologies originated from the waterfall model proposed by Royce [168], in which development was supposed to proceed linearly

through the phases of the requirement analysis, design, implementation, testing (validation), integration and maintenance. Researchers criticized Waterfall model for its excessive separation of different phases. In an attempt to overcome the shortcomings of the waterfall model many new software development approaches such as spiral model by Boehm [24], iterative enhancement by Basili et al. [14], rapid prototyping by Gomaa [69], evolutionary prototyping and incremental development by Floyd [56] had been suggested. In spiral development process, a desired capability is identified, but the end-state requirements are not known at program initiation. Those requirements are refined through demonstration and risk management, there is continuous user feedback and each increment provides the user the best possible capability. The requirements for future increments depend on feedback from users and technology maturation, whereas the basic idea behind the iterative enhancement was to develop a software system incrementally, allowing the developer to take advantage of what was being learnt during the development of earlier, incremental, deliverable versions of the system. Software development approaches incorporating prototyping have gained respectability as they have proved to be able to dynamically respond to changes in user requirements, reduce the amount of rework required and help control the risk of incomplete requirements. Currently reuse model has achieved tremendous success in information system development developed by Frakes [58]. The aim of Component-based software development (CBSD) by Aoyama [9] is to develop new software by widely reusing pre-fabricated software components. Many other researches (Kaushaar et al. [107], Boehm et al. [23], Gordon [71], Alavi [8], Naumann et al. [148], Tate et al. [180] and Palvia et al. [152] etc.) contributed in development and growth of these methodologies.

Many researchers have contributed to business process development. A large number of process models were developed (Armenise et al. [10], Bandinelli et al. [13], Bubenko, [28], Decker et al. [43], Jarzabek et al. [102], Jacobson et al. [100], Rumbaugh et al. [169] and Marca et al. [139] etc.).

Software configuration (Change) management (SCM) is the discipline discussed by Tichy [183] enables us to keep evolving software products under control and thus contributes to satisfy quality and delay constraints. Bersoff et al. [18] explained the purpose of SCM to manage change throughout the software development process. Change is a very natural and intrinsic aspect of software development process. SCM has been the focus of software engineering research and a great amount of research has been carried out on SCM. From the researches by Burrows et al. [29] and Dart [41], eight areas of functionality of SCM systems were found: (i) Version control, (ii) Configuration

support, (iii) Team support, (iv) Change control, (v) Build support, (vi) Process control, (vii) Status reporting and (viii) Audit control. These functional areas mainly cover the management issues of software development. To provide these eight areas of functionality, different SCM systems use different models, such as the checkout/check by Feller [54] in model, the composition model, the long transaction model, and the change set model. In recent years, the focus of SCM research is on software process support in SCM systems (Estublier et al. [48] and Leblang [126]), distributed configuration management systems (Hunt et al. [96] and Milewski [143]) and unified version models (Conradi et al. [39]) etc.

Goals are essence of management. Drucker [47] advocated management by objectives to be one of the most important motivation factors for the success of any organization. Attempts have been made to incorporate goals into process modeling Kueng et al. [122] suggested that an informal approach in which goals provide a basis for process definition. Khomyakov et al. [110] presented a model based on mathematical systems theory. This set of concepts was extended by Bider et al. [20] and used for defining a process pattern, allowing the design of generic processes that can be specialized for specific situations. The goals addressed by this approach are operational goals only, termed “functional goals”. Rolland [165, 166] applied goals and soft-goals for requirements elicitation in combination with scenarios and others contributions to relate goals with process models and its impact on strategic success.

The potential of the Internet to reach a large and growing body of customers, coupled with low communication costs, makes it a very attractive business medium to many organizations. Although there is significant interest in the use of the Internet for business purposes, studies articulating issues that can guide business managers in its use are lacking. Use of Internet and related issues are the hottest research areas. Different studies conducted and contributed (Brandtweiner et al. [25], Cho [38], Gonsalves et al. [70], Hamill [75], Weill [187] and Lee [128] etc.) to the aspects related to the use of Internet for business, marketing, performance and modeling of systems etc.

Therefore, a great amount of active research has been carried out in data mining, supervised learning as well as for developing system development methodologies, business process development, change management, goal orientation and its organizational implementation in modeling and Internet usage. In spite of such developments, we are facing many problems in supervised learning as well as in

development and maintenance of knowledge based information systems. This research work is an effort to identify those problems and to find practical solutions for them.

2.4 Conclusion

In this Chapter, detailed literature survey is conducted for examining previous developments in the areas of data mining, supervised learning as well as of software and knowledge engineering. These areas are in vision to be the major concentration of present research work. As data mining is popularized in last two decades, so literature survey is conducted from the day, the term data mining was coined to till date. The main orientation of present research is towards improvements in knowledge discovery in database (KDD) process or supervised learning process, so supervised learning process is examined more closely. Research investigations are involved in improving KDD process as well as for developing a system development model to support knowledge based system development, so previous developments of software and knowledge engineering are investigated more closely and are presented in this Chapter.

Chapter 3

Supervised Learning Processes, Techniques and Algorithms

3.1 Various Processes for Supervised Learning

Supervised learning processes can vary from simple processing to very complex processing. Different techniques and algorithms are used to extract knowledge from data. These algorithms involve certain criteria to extract knowledge. A particular technique or algorithm may be suitable for a segment of problems only, but may not be suitable for others. There is no unique technique or algorithm that solves all types of problems. Supervised learning involves the use of the train set to train algorithms for the creation of a model of that technique or algorithm. This model is then applied on the test set to generate and compare results. Different supervised learning processes as discussed by Witten et al. [188] are as follows:

3.1.1 Simple Supervised Learning

In its simplest form, input data is applied to classification algorithm and result is generated. It is also called Fixed Split experimentation as data is divided into the train and the test set. Such experimentation suffers with over fitting and under fitting model and results may not fulfill the reliability criteria. So there is a need for preprocessing and post-processing of data.

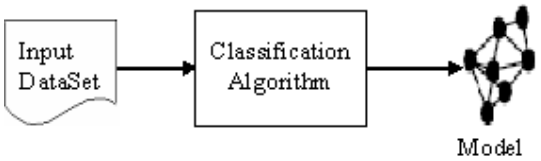

Figure 3.1: Simple Supervised Learning Process.

3.1.2 Preprocessing of the data

A data set collected is not directly suitable for induction (knowledge acquisition); it comprises in most cases noise, missing values, the data are not consistent, the data set is too large, and so on. Therefore, we need to minimize the noise in data, choose a strategy for handling missing (unknown) attribute values, use any suitable method for selecting and ordering attributes (features) according to their informativity (so-called attribute mining), discretize/fuzzify numerical (continuous) attributes and eventually, process continuous classes.

3.1.2.1 Attribute Transformation

Input data may be nominal or numerical. Few classification algorithms like ID3 and Naïve Bayes operate only on discrete data, whereas regression based algorithms operate only on numerical data. So there may a requirement of transformation of data from one form to another to match the data with algorithmic requirements.

(a) **Categorical Attribute Transformation:** Nominal or categorical data may be transformed into binary or scale values as follows:

(i) **Categorical to Binary:** Problems having more than two categories of class attribute are converted into Binary class problems. We have converted our datasets into binary class treating first half of class categories as negative class and last half as positive class.

(ii) **Dual Scaling:** Dual scaling (Nishisato [151]) is a multivariate method for assigning scale values to the rows and columns of a table of data, with certain optimal properties.

(b) **Continuous Attribute Transformation**

(i) **Class-based discretization:** Class-Attribute relationship is used to define discretization of any attribute, each attribute is discretized independently. Such

discretization is useful for small number of attributes, but becomes complex for large number of attribute.

(ii) Fixed-bin discretization: All the attributes to be discretized are considered collectively and a fixed number of bins are used for discretizing all attributes. We have used fixed bin discretization, so that the future researchers can utilize the results of this paper for their comparative analysis and it also helps in maintaining consistency of experimentation.

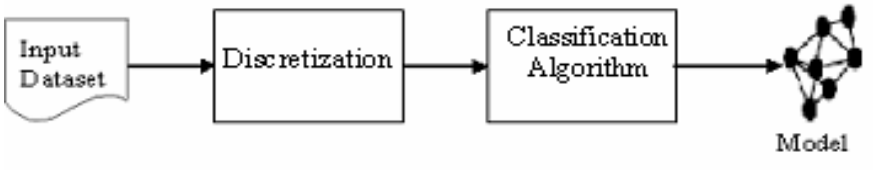

Figure 3.2: Discretized Supervised Learning Process.

(iii) Principle component analysis: Principal component analysis is a useful tool for categorization of data, it separates the dominating features in the data set.

3.1.2.2 Data Sampling

There are three types of sampling methods as follows:

(a) Progressive sampling: Progressive Sampling (PS) (Provost et al. [160]) incrementally constructs the train set from a larger dataset without decreasing the classification performance and without altering the initial format of the examples.

(b) Random sampling: Samples are selected randomly for experimentation. Such a sampling makes the experimentation results unreliable as different sampling algorithms may select samples differently and results may vary significantly.

(c) Stratified sampling: Stratified sampling is based on re-sampling the original datasets in different ways: under-sampling the majority class or over-sampling the minority class.

3.1.2.3 Validation

(a) Fixed Split Validation: Simplest form of experimentation is to divide dataset into two fixed length datasets of the train set and the test set to perform experiments directly. This kind of experimentation is meant for simple testing of algorithms.

Biggest problem with fixed split is over fitting of the training data e.g. tree based techniques may have too many branches that may reflect anomalies and result in poor accuracy of unseen samples. To overcome the limitations of fixed split, two approaches used are prepruning and post pruning. Prepruning is performed through cross-validation, whereas many calibration methods have been proposed for post pruning. Following sections include the discussion about these methods.

(b) Cross Validation: To evaluate the robustness of the classifier, the normal methodology is to perform cross validation on the classifier. Ten fold cross validation has been proved to be statistically good enough in evaluating the performance of the classifier. For the present study, datasets are divided into the train and the test sets. Then this train set is equally divided into 10 different subsets for ten fold cross validation. Nine out of ten of the train subsets are used to train the learner and the tenth subset is used as the test set. The procedure is repeated ten times, with a different subset being used as the test set. In this way cross validation is performed to calibrate the models and select the best parameters and then models are applied on the large final test set.

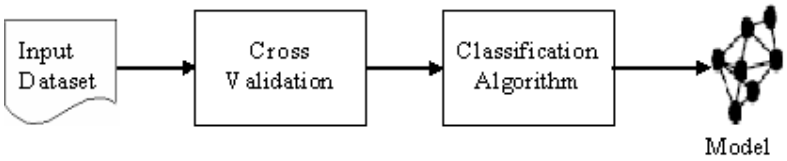

Figure 3.3: Cross-Validated Supervised Learning Process.

3.1.3 Post processing of the knowledge derived

The pieces of knowledge extracted in the previous step could be further processed. One option is to simplify the extracted knowledge. Also, we can evaluate the extracted knowledge, visualize it, or merely document it for the end user. There are various techniques to do this. Next, we may interpret the knowledge and incorporate it into an existing system, and check for potential conflicts with previously induced knowledge.

3.1.3.1 Calibration

Many learning algorithms do not predict probabilities. For example the outputs of an SVM are normalized distances to the decision boundary, whereas naive bayes models

are known to predict poorly calibrated probabilities, because of the unrealistic independence assumption.

A number of methods have been proposed for mapping predictions to posterior probabilities. Platt Scaling (Platt [158]) is used for transforming SVM predictions to posterior probabilities by passing them through a sigmoid. Platt scaling also works well for boosted trees and boosted stumps (Niculescu-Mizil et al. [150]). A sigmoid is also not the correct transformation for all learning algorithms.

Second method used for calibration is Logistic regression. Logit Boost algorithm is used for performing additive logistic regression. This algorithm performs classification using a regression scheme as the base learner and can handle multi-class problems (Friedman et al. [63]) and it can also do efficient internal cross-validation to determine appropriate number of iterations.

Other method generally used for calibration is Isotonic Regression (Zadrozny et al. [194,195]; Robertson et al. [164]). It is used to calibrate predictions from SVMs, naive bayes, boosted naive bayes, and decision trees. Isotonic Regression is more general method, but its only restriction is that the mapping function used is isotonic (monotonically increasing). A standard algorithm for Isotonic Regression that finds a piecewise constant solution in linear time, is the pair-adjacent violators (PAV) algorithm (Ayer et al. [12]).

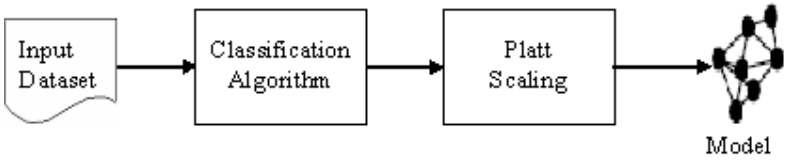

Figure 3.4: Post-Processed Supervised Learning Process.

3.1.3.2 Thresholding

This is the minimum acceptable value which, in the user's judgment, is necessary to satisfy the need. If threshold values are not achieved, program performance is seriously degraded, the program may be too costly, or the program may no longer be timely.

(a) **Class Probability Estimators (CPE) thresholding:** For a decision maker to act optimally it is necessary to estimate the probability of success. Because training information is costly, we would like to reduce the cost of inducing an estimation model that will render decisions of a given quality. One approach to reducing the cost of learning accurate CPEs is via traditional active learning methods, which are designed to improve the model's average performance over the instance space. if the probability of a successful outcome exceeds the threshold.

(b) **Regression Thresholding:** Threshold regression refers to first-hitting-time models with regression structures that accommodate covariate data. The parameters of the process, threshold state and time scale may depend on the covariates.

3.1.4 Stacking

Stacking combines the output of a number of classifiers. Stacked generalization, also known as Stacking in the literature, is a method that combines multiple classifiers by learning the way that their output correlates with the true class on an independent set of instances. At a first step, N classifiers Ci, i = 1..N are induced from each of N data sets Di, i = 1..N. Then, for every instance e_j , j = 1..L of an evaluation set E, independent of the Di data sets, the output of the classifiers Ci(e_j) along with the true class of the instance class(e_j) is used to form an instance m_j , j = 1..L of a new data set M, which will then serve as the meta-level train set. Each instance will be of the form: C1(e_j), C2(e_j), . . . , CN(e_j), class(e_j). Finally, a global classifier GC is induced directly from M. If a new instance appears for classification, the output of all local models is first calculated and then propagated to the global model, which outputs the final result. Any algorithm suitable for classification problems can be used for learning the Ci and GC classifiers. Independence of the actual algorithm used for learning Ci, is actually one of the advantages of Stacking, as not every algorithm might be available for each data set and not the same algorithm performs best for every data set. We have applied stacking of isotonic regression with other classification algorithms.

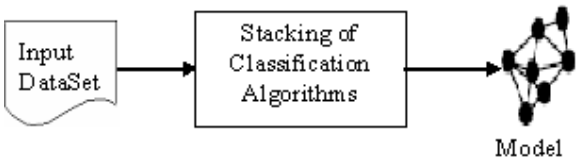

Figure 3.5: Stacked Supervised Learning Process.

3.1.5 Complex Processing

Different preprocessing, Post-processing and stacking of different algorithms may be combined to extract knowledge from databases. Such complex criteria may involve parallel processing of different algorithms as well.

Out of these five broader categories six specific processes are practiced worldwide. These six processes are Fixed Split, Cross Validation Stacking Procedure and Three Calibration methods. In present study these six processes are used for experimentation. Present research contributes a new process based on preprocessing of data and can be categorized under Complex processing. So, present research included seven processes for experimentation.

3.2 Description of Data Mining Techniques and Algorithms used for study

A variety of classification algorithms were used for study. Different techniques included for study with their specific algorithms are described as follows:

3.2.1 Decision Trees

Decision trees are tree-shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset. Decision trees represent a series of IF…THEN type rules which are linked together and can be used to predict properties for observations based upon the values of various features. These are able to produce human-readable descriptions of trends in the underlying relationships of a dataset and can be used for classification and prediction tasks. The algorithms used for experimentation were ADTree, Decision Stump and REPTree etc. Different parameters were set as follows: maximum tree depth was allowed to be infinite, minimum number of instance per leaf were set to 2, Confidence threshold was set to 0.25 and numbers of trees were allowed to be infinite.

3.2.2 Support Vector Machine

These are methods for creating functions from a set of labeled training data. These functions can be a classification function (the output is binary: is the input in a category) or the function can be a general regression function. For classification, SVMs operate by finding a hyper-surface in the space of possible inputs. This hyper-surface will attempt to

split the positive examples from the negative examples. The split will be chosen to have the largest distance from the hyper-surface to the nearest of the positive and negative examples. Intuitively, this makes the classification correct for testing data that is near, but not identical to the training data. We have included LibSVM and SMO algorithms for study. Different parameters were set as follows: Different types of kernel functions were tried like linear, polynomial, radial basis function etc., Degree of kernel function set to 3 and Tolerance parameter set to 0.001.

3.2.3 Genetic Algorithms

Optimization techniques that use process such as genetic combination, mutation, and natural selection in a design based on the concepts of evolution. Genetic algorithms should be used, when no other option is left. We have not included any genetic algorithm, but learning processes are always based on genetic processing, so indirect contribution of genetic processing can not be neglected.

3.2.4 Neural Networks

A neural network consists of a set of highly interconnected entities, called *nodes* or *units*. Each unit is designed to mimic its biological counterpart, the neuron. Each neuron accepts a weighted set of inputs and responds with an output. An Artificial Neural Network (ANN) technique is based upon the model of biological nervous system. The key element of this technique is the novel structure of the information processing system. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. We have included Multi Layer Perceptron (MLP) algorithm for study. Different parameters were set as follows: Learning rate of back propagation set to be 0.3, Momentum rate 0.2 etc.

3.2.5 K-nearest neighbor

Among the various methods of supervised statistical pattern recognition, the Nearest Neighbor rule achieves consistently high performance, without *a priori* assumptions about the distributions from which training examples are drawn. It involves a train set of both positive and negative cases. A new sample is classified by calculating the distance to ‘k’ nearest training cases, sign of that point then determines the classification of the sample. The IBk classifier included in present study extends this idea by taking the *k* nearest points and assigning the sign of the majority. It is common to select *k* small and

odd to break ties (typically 1, 3 or 5). Larger *k* values help reduce the effects of noisy points within the training dataset and the choice of *k* is often performed through cross-validation. Different values for k were tried ranging between 1 and 10.

3.2.6 Rule Induction

This technique is the extraction of useful if-then rules from data, based on statistical significance. We have included Decision Table and ZeroR algorithms for study. We have taken Confidence threshold to be 0.25.

3.2.7 Logistic Regression

This method is directed to estimate a probability rather than estimating the value for variables and can be used as a memory-based classification method. We have included Simple Logistic algorithm in this study. Different parameters were set as follows: Ridge in the log-likelihood set to 1.0e-8, Maximum number of iterations allowed being infinite.

3.2.8 Bayesian Approach

Given the probability distribution, Bayes classifier can provably achieve the optimal result. Bayesian method is based on the probability theory. We have included Naïve Bayes Updatable and Bayes Net Generator algorithms for study.

3.2.9 Boosting/Bagging

These methods create a set or ensemble of classifiers from a given dataset. Each classifier is generated with a different train set obtained from the original using re-sampling techniques. The final output is obtained by voting.

Boosting: The idea of Boosting is to combine simple rules to form an ensemble such that the performance of the single ensemble member is improved i.e. Boosted. AdaBoostM1 algorithm was used for boosting trees (Yoav et al. [192]). Different parameters were set as follows: Number of iterations allowed was 10 and 100 percentage of weight mass being used.

Bagging (**B**ootstrap A**gg**regat**ing**): It produces replications of the train set by sampling with replacement. Each replacement of the train set has the same size as the original

set, but some examples can appear more than once while other don't appear at all. A classifier is generated from each replication. All classifiers are used to classify each sample from the test set using a vote scheme (Breiman [27]). We have applied Bagging and Boosting on ADTree, Decision Stump and Random Forest algorithms. Experimental results of both Boosting and Bagging are really enthusiastic. Different parameters were set as follows: Size of each bag being set to 100 and number of iterations allowed were 10.

In present study twenty algorithms were identified for experimentation. These algorithms are ADTree, BayesNetGenerator, Decision-Stump, Decision-Table, IB-k, LibSVM, MultiLayerPerceptron, Naïve-Bayes-Updatable, PART, Random-Forest, REPTree, SimpleLogistic, SMO, ZeroR. Bagging and Boosting were applied on decision tree based algorithms that is ADTree, Decision Stump and Random Forest algorithms. As stated earlier present research proposes a new process for supervised learning described in the next chapter.

3.3 Conclusion

In this chapter, different processes, algorithms used for supervised learning are discussed. Different processes related to simple, pre processing and post processing as variants of general KDD processes are identified and are examined closely. Out of different processes discussed in this chapter, six processes used extensively worldwide were identified for experimentation in present research. Different techniques and algorithms practiced worldwide are also described in present chapter. Twenty algorithms out of these algorithms were identified to be included in this investigation for experimentation.

PART II

Chapter 4 Supervised Learning Through Fuzzy Boundaries of Regression Based Clusters ……. 36-50

Chapter 5 'Genetic Information System Development and Maintenance' Model 51-65

Chapter 4

Supervised Learning Through Fuzzy Boundaries of Regression Based Clusters

In Chapter 3 different preprocessing and post processing methods were analyzed. In this chapter a new preprocessing method is proposed for supervised learning. Proposed method is based on pruning parts of the train set incrementally through clusters formed from the output of classification and regression based algorithms. Proposed process targets to fill the pitfalls left by most popular preprocessing methods like Cross Validation etc.

4.1 Critical Analysis of Previously known Preprocessing and Post Processing Methods

As Simple/Fixed Split experimentation suffers with over fitting and under fitting model and results may not fulfill the reliability criteria. So, there is a need for preprocessing and post-processing of data. Attribute transformation, Data Sampling and Validation methods are used for preprocessing data. Among these Cross Validation is most popular method that is based on dividing the train set into many (usually 10) parts and then using nine parts as the train set of algorithm and tenth part as a test set. Results are generated through their experimentation. Now tenth part is replaced with one of the parts from remaining nine parts and then again experimentation is performed and like this ten results are generated. Best performing set of nine subsets of the train set are selected for generating model for future predictions. Tenth part of the train set is left unanalyzed

only due to the reason that rest of the nine parts performed well on this data and is never used for learning through algorithm. This remaining tenth part never becomes a part of the learning process. Here objection arises that if a part of the train set performed well for a small test set, it does not promise to predict well for a large test set also. Leaving a part of the train set without utilizing it for learning can never be called a wise idea. These objections are countered by post processing algorithms as these allow whole of the train set to help in growing the model completely and then to reduce size of the model by removing less promising parts of the model.

In case of a decision tree, we can say that in preprocessing, less promising branches of decision tree are cut while developing model whereas in post-processing, fully grown tree is reduced by cutting less promising branches. One can say that in preprocessing top down approach is adopted, whereas in post-processing bottom-up approach is adopted. Post processing algorithms suffer a drawback to find the basis on which less promising branches can be cut. If model generated from an algorithm is applied over the train set for post pruning then it can not be much reliable as same data is used for training as well as for pruning. On the other hand if model is generated from the train set and post pruning is done on separate data then post pruning will vary with data used for post pruning. In practice, the train set is fed back for post pruning leaving objection over defining poorly performing branches of a model. A model is well worse with most of the aspects related to the train set and it leaves very less cushion for post pruning aspects. Above that deciding the threshold value for inclusion or exclusion of certain branch is a tedious job.

Whenever top down and bottom approaches are discussed, possibility of hybrid approach arises naturally. Cross validation limits its reliability before post processing algorithms as one part of the train data is never analyzed at all. Present research work concentrates on developing a process that analyzes whole data and reduces the size of the train set incrementally. Proposed process involves classification and regression based algorithms to be processed in parallel and reduces the size of the train set suitably. Present research work analyzes the effectiveness of proposed method for preprocessing and compares the results for all previously known processes with results of proposed process. Present research concentrates on preprocessing of data and to reduce the size of the train set but it also provides foundation for developing hybrid approaches in future.

4.2 Conceptual Foundation for Proposed Process

Proposed process is a combination of many different concepts. It involves regression, fuzzy set theory and clustering for classification tasks through supervised learning. We briefly revisit all these concepts again.

4.2.1 Supervised Learning and Classification

Supervised learning is fairly common in classification problems because the goal is often to get the computer to learn a classification system based on the learning through training data. Classification learning is appropriate for any problem where deducing a classification is useful and the classification is easy to determine. Supervised learning techniques are highly dependent on the information given by the pre-determined classifications. For a classification problem, the goal of the learning algorithm is to minimize the error with respect to the given inputs. These inputs, often called the train set, are the examples from which the agent tries to learn. Also not all of the train sets have the inputs classified correctly. This can lead to problems if the algorithm used, is powerful enough to memorize even the apparently "special cases" that don't fit the more general principles. This can lead to over fitting and it is a challenge to find algorithms that are both powerful enough to learn complex functions and robust enough to produce generalized results.

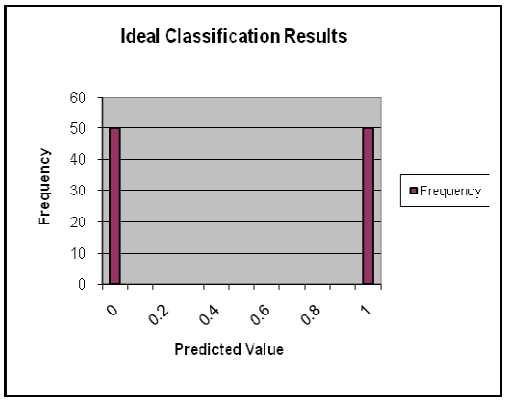

Figure 4.1: Ideal Results for Classification

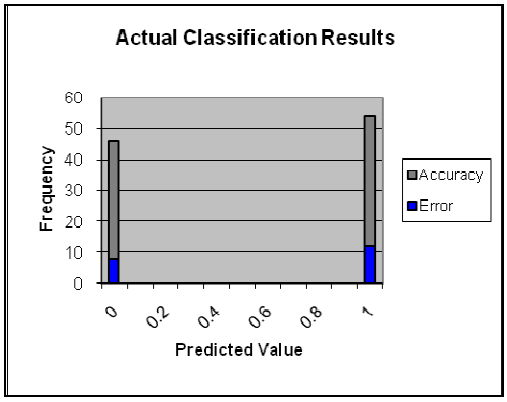

Figure 4.2: Actual Classification Results

4.2.2 Regression Analysis

Regression analysis involves the prediction of continuous outcomes. Building a regression model involves collecting predictor and response values for common samples

and then fitting a predefined mathematical relationship to the collected data. More generally, classical indications for regression as a predictive tool are when there is a requirement to use cheap, easy-to-perform measurements as a substitute for more expensive or time-consuming ones and also when it is required to build a response surface model from the results of some experimental design. In a regression problem the dependent variable Y is numerical. A prediction rule is defined analogous to a classifier. It is a function '*d* ' such that:

$$d : \mathrm{dom}(X1) \times \cdots \times \mathrm{dom}(Xm) \rightarrow \mathrm{dom}(Y). \quad \ldots\ldots (4.1)$$

4.2.3 Fuzzy sets

For any crisp set A, it is possible to define a characteristic function $\mu_X = \{0, 1\}$, i.e. the characteristic function takes either of the values 0 or 1 in the classical set. For a fuzzy set (Zadeh [193]) the characteristic function can take any value between zero and one.

The membership function $\mu_A(x)$ of a fuzzy set A is a function $\mu_A(x): X \rightarrow [0,1]$ .... (4.2)
So every element in x in X has membership degree: $\mu_A(x) \in [0,1]$ ..... (4.3)
A is completely determined by the set of tuples: $A = \{(x, \mu_A(x)) \mid x \in X\}$ ..... (4.4)

$\mu_A(x) = 0$ if the element x | X does not belong to A, $\mu_A(x) = 1$ if x completely belongs to A and $0 < \mu_A(x) < 1$ if x partially belongs to A.

4.2.4 Clustering

Clustering (Jain et al. [101]) is the unsupervised classification of patterns (observations, data items or feature vectors) into groups (clusters). Cluster analysis is the organization of a collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity. Intuitively, patterns within a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster. The difference between clustering (unsupervised classification) and supervised classification is that in supervised classification we are provided with a collection of *labeled* (pre-classified) patterns, but the problem is to label a newly encountered, yet unlabeled, pattern. Typically, given labeled (*training*) patterns are used to learn the descriptions of classes which in turn are used to label a new pattern, whereas in case of clustering, the problem is to group a given collection of unlabeled patterns into meaningful clusters. In a sense, labels are associated with clusters also, but these category labels are *data driven*; that is, they are obtained solely from the data.

4.3 Issues/Problems related to Supervised Learning, Classification, Regression and their Fusion

Supervised learning process is applied with certain modifications for different problems and their environments. This type of customization demands big deal of domain knowledge and experience in implementing suitable process for problem at hand. Different problems related to supervised learning, classification, regression and their fusion are discussed as follows.

4.3.1 Issues/Problems Related to Supervised learning processes

Data mining researches has explored sufficiently large number of algorithms for supervised learning. Different algorithms classify data according to their own criteria and usually differ in their results. Different algorithms accompany many parameters to be adjusted for the improvement of their performance. Supervised processing involves many processes as well however finding a suitable process is a difficult task. These processes target to optimize size of the model with respect to learning. All these factors create difficulty in applying a suitable process for problem at hand.

4.3.2 Issues/Problems related to Classification of data

The need for classification task appears in a variety of applications, from industry to research and from common life to professional life, where there is need to select one among many available options. Training of computers for such decision making tasks has gained huge attention. Different applications use data mining solutions for their classification problems. Varying applications demand varying modification of supervised learning process. These applications also demand variety of metrics for their analysis of data. Varying applications, different mining processes and need for different metrics makes classification process complex.

4.3.3 Issues/Problems related to Classification through Regression based algorithms

Regression based algorithms can be used to classify quantitative data. Qualitative data need to be coded into quantitative data. Experience with regression based algorithms for classification is not so much impressive as compared to the techniques like decision

trees and neural networks etc. Above that different data coding practices may influence the classification results as well.

4.3.4 Issues/Problems related to Fusion of Classification and Regression Algorithms

Most of the classification tasks involve qualitative as well as quantitative data, whereas regression based algorithms strictly work for quantitative data. Fusion of classification and regression algorithms with the help of techniques like Stacking is a difficult task, as many classification techniques can't be incorporated on numeric data. Results obtained from stacking of classification and regression algorithms performed poorly when combined with other algorithms. So fusion of regression with classification task was not being explored so well till now. Here we target to explore this possibility through parallel execution of these algorithms, where results should help in improving the size and performance of model generated by different algorithms.

4.4 Conceptual Foundation of Fuzzy Boundaries of Regression Based Clusters process

Ideally classification algorithm must classify whole data accurately. But data is rarely classified with complete accuracy. When classification algorithm is applied over data, it classifies data with certain accuracy as well as with certain inaccuracy. Figure 4.1 demonstrates ideal classification of data, whereas Figure 4.2 demonstrates practical outcome, where misclassified data is represented as error. Classified data is shown to be having discrete values. These discrete values can be replaced with class probability estimator (CPE). A CPE is built to estimate the probability of output of classification algorithm. It represents the probable confidence in classified data. If we plot different CPEs over a frequency chart, then frequency of data will get distributed continuously as depicted in Figure 4.3. CPE values may be expected to be having more frequency near extremes and we can expect decrease in confidence as we move towards center of both extremes. Beyond center there will be misclassified data with very less confidence in favour of right prediction value and more confidence in favour of wrong prediction value. Figure 4.3 visualizes frequency chart of CPEs.

Density of data is expected to be more at extremes as compared to the data scattered towards center. Areas marked with labels 1 and 2 represent correctly classified data. Densities of these areas are shown to be decreasing as we move from extremes to

towards center. Regression based algorithms generate continuous outcomes. Their results are real numbers extending values over a range, rather discrete values as shown in Figure 4.4.

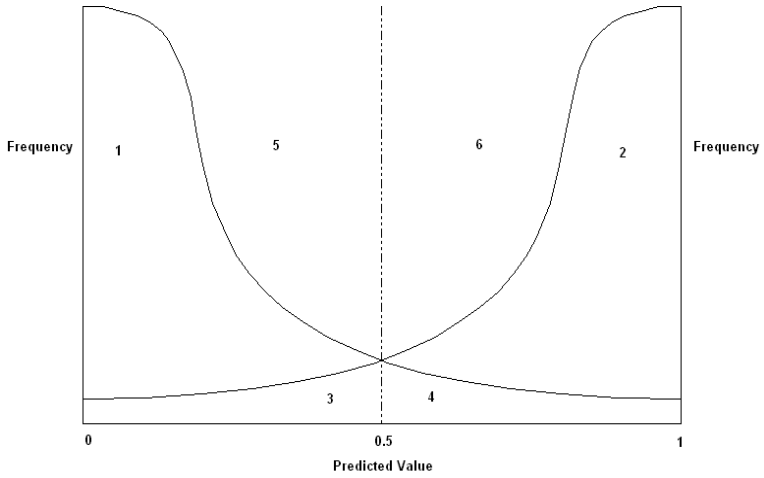

Figure 4.3: Frequency chart for Class Probability Estimate (CPE) of Classification algorithm

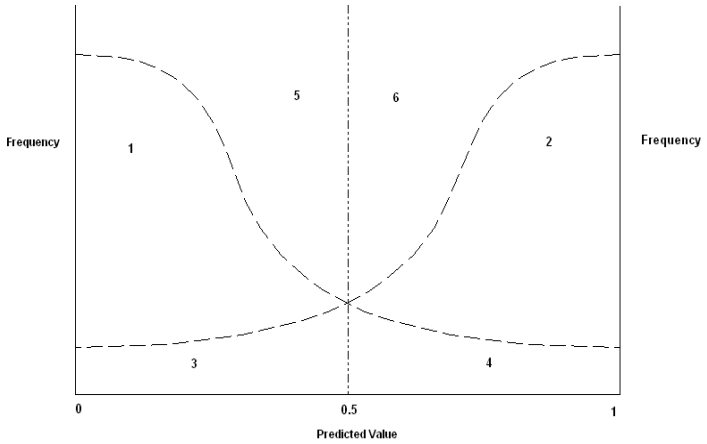

Figure 4.4: Frequency Chart from Continuous outcome from regression algorithm

Classification algorithms generate discrete outcomes, but CPE generates a range of data like the output of regression algorithms. Performance of classification algorithms is usually better than regression algorithms. When frequency charts of all classification algorithms are prepared these are expected to have larger frequencies toward extremes and less frequencies toward center as shown in Figure 4.3 for output from a binary classification, whereas regression based algorithms are supposed to have relatively less frequencies near extremes and little bulgy frequencies in centers as shown in Figure 4.3 for same binary classification data as used for classification. Regions labeled 1 and 2 in Figures 4.3 and 4.4 represent accurately classified data, whereas regions labeled 3 and 4 represent misclassified data. If same binary data is used with regression as well as classification algorithms, where regression algorithms' output as a real numbered output

and classification algorithms' output as CPE and frequency charts of both kinds of outputs are combined, then combined frequency chart will appear like Figure 4.5.

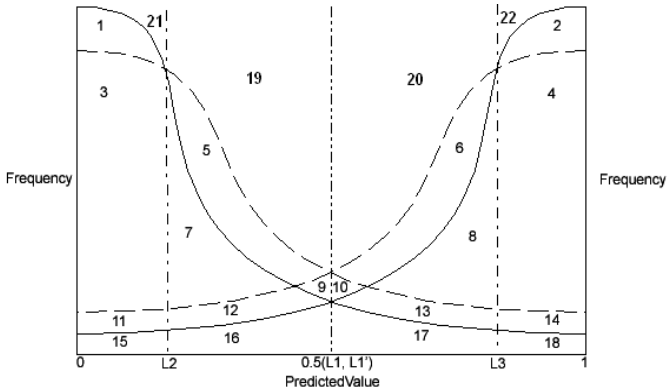

Figure 4.5: Combined Frequency charts of Class Probability Estimate (CPE) and Continuous Outcomes of Classification and Regression algorithms respectively

Regions marked with labels 1, 2, 3 and 4 represent accuracy from both frequency charts. Regions with labels 3 and 4 represent common accuracy for combined frequency chart. As classification algorithms usually perform better than regression algorithms, so at extremes frequencies of output from classification algorithms will have bigger peaks as compared to output from regression algorithms. Vertical lines L1 and L2 represent the crossover of the curves of classification and regression where frequencies of outputs from regression algorithms will have more values as compared to the frequencies of outputs from classification algorithms.

For binary problems curves from both extremes will stretch like this up to other extreme. As low performance algorithms have more misclassified data, so regions labeled from 11 to 18 will contain misclassified data and regression curves will continue to be more bulgy. Region extending from vertical line L2 to L3 will include this crossover data. As, we can be quite confident about the results related to common region with correctly classified data as both classification and regression algorithms have reasonably good confidence in these regions about their outcome. But regions extending from vertical line L2 to L3 contain data that is under suspicion, this is a fuzzy region that need to be invested more so that more confidence can be generated for this region. Misclassified data also needs to be investigated more closely, so that model generated by classification/regression algorithm is trained for this misclassified data. In this way, we can conclude that data belonging to regions labeled 3 and 4 is verified qualitatively and quantitatively by classification and regression algorithms respectively. Model generated

through training of the train set is quite confident about correctly classified data within regions labeled 3 and 4 in Figure 4.5. Data belonging to regions 3 and 4 can be excluded from the train set, as it is clearly understood by previous training of algorithm and is verified to be understood well qualitatively as well as quantitatively. As size of dataset affects size of the model, reducing the size of the train set will keep the size of model at minimum level. When well understood data is excluded from the train set, one can expect to reduce the size of the train set and model generated from training without affecting the quality of outcome.

Fuzzy boundaries formed in Figure 4.5 represent the situation, when centers of both curves of classification as well as regression output meet at same point. Meeting points of these curves may be different from each other. Figures 4.6 and 4.7 represent such a situation. In Figure 4.6 meeting point of regression output has tilted to the left of curves from classification algorithm‘s output, whereas in Figure 4.7 it has tilted to right. In both these curves strongly confident region is labeled with regions 3 and 4 again. Rest of the data labeled from 1 to 22 excluding regions 3 and 4 need to be analyzed more closely.

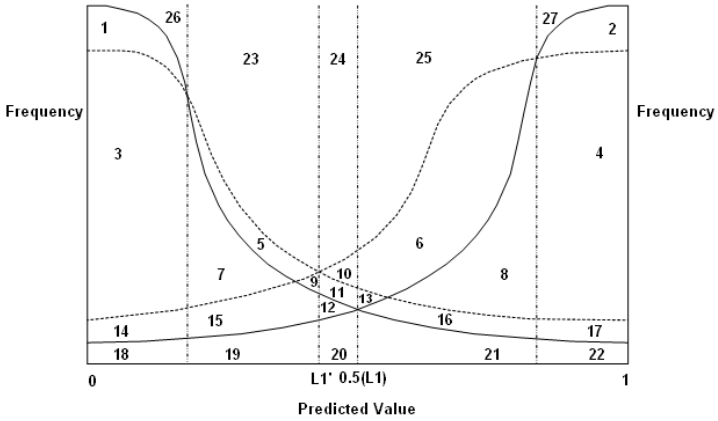

Figure 4.6: Center of Regression output tilted to the left of Classification Output

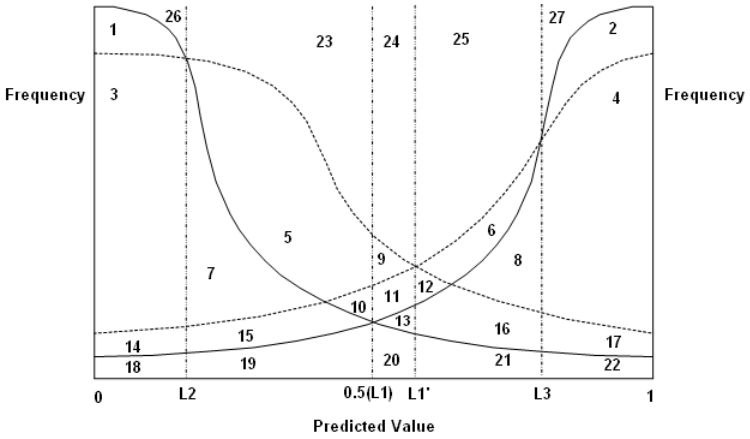

Figure 4.7: Center of Regression output tilted to the right of Classification Output

There may be certain other issues related to proposed theory, so section 4.6 will investigate these issues more closely. The following section presents an algorithm for supervised learning based on concept presented in this section.

4.5 Proposed Improved Supervised Learning Algorithm based on fuzzy boundaries of regression based clusters

Using the concept presented in previous section an algorithm is prepared for supervised learning. In proposed process, the train set is divided into ten parts (i. e. Δ_1, Δ_2 …., Δ_{10}). First subset of the train set (Δ_1) is used to train classification and regression algorithms in parallel. Models generated from these algorithms are applied in parallel, on second subset of the train set (Δ_2) to generate outputs. Classification algorithm's outputs are generated in the form of class probability estimate (CPE), whereas regression algorithm's output is always a set of floating values scattered over a range. Frequency charts are generated for the outputs of both kinds of algorithms. Frequency charts of both algorithms are plotted on single graph. Combined frequency chart is used to identify data near extremes with high confidence through fuzzy boundaries of regression based clusters as explained in section 4.4. Data with higher degree of confidence is excluded from the second subset (Δ_2) of dataset that is used as a test set in this iteration. Remaining data of second subset of the train set (δ), after excluding data with higher degrees of confidence, is added into first subset of data i. e. $\Delta_2' = \Delta_1 + \delta$. This set ($\Delta_2'$) of data will have less than twenty percent of the train set. Now this data (Δ_2') is used to train classification and regression algorithms in parallel and third subset (Δ_3) of data is used to test the model generated from these algorithms. Data with greater confidence is excluded from the third subset of data and is added into previous data having data from first two subsets. Like this we continue to include data in the train set incrementally and ultimately we get a train set (Δ_{10}') having less than hundred percent of the actual train set. Now this train set is applied on the actual test set (ψ) to generate output.

Algorithm 4.1: Algorithm for fuzzy boundaries of Regression Based clusters processing

Step 1: Input The Train Set ξ and The Test Set ψ.

Step 2: Convert The Train Set ξ into Ten Incremental subsets Δ_1, Δ_2 …., Δ_{10}. Set i=1, Pick Δ_1 as a Train Set and Δ_2 as a Test Set.

Step 3: Use Δ_i as a Train Set and Δ_{i+1} as a Test Set

Parallel Begin

Step 3.1.a: Apply Classifier through training of The Train Set Δ_i over The Test Set Δ_{i+1} (Treating the Incremental Train Set as a Test Set).

Step 3.1.b: Generate Output in the form of Class Probability Estimate (CPE).

Step 3.1.c: Separate δ' records from the Incremental Train Set guided by CPE output after applying proposed Theory.

Step3.2.a: Apply Regression Algorithm through training of The Train Set Δ_i over The Test Set Δ_{i+1} (Treating the Incremental Train Set as a Test Set).

Step 3.2.b: Generated Output Continuous Estimates.

Step 3.2.c: Separate δ" records from the Incremental Train Set guided by continuous output after applying proposed Theory.

Parallel End

Step 4: Compute $\delta = \delta' \cup \delta''$.

Step 5: Compute $\Delta' = \Delta_i \cup \delta$.

Step 6: Condition: IF more incremental train subset exists, THEN Increment i= i+1, Set $\Delta_i = \Delta'$ and Go to Step 3, ELSE set $\xi' = \Delta'$ and go to step 7.

Step 7: Apply Classifier through training of The Train Set ξ' over The Test Set ψ.

Step 8: Generate Output with different metrics.

Step 9: Stop.

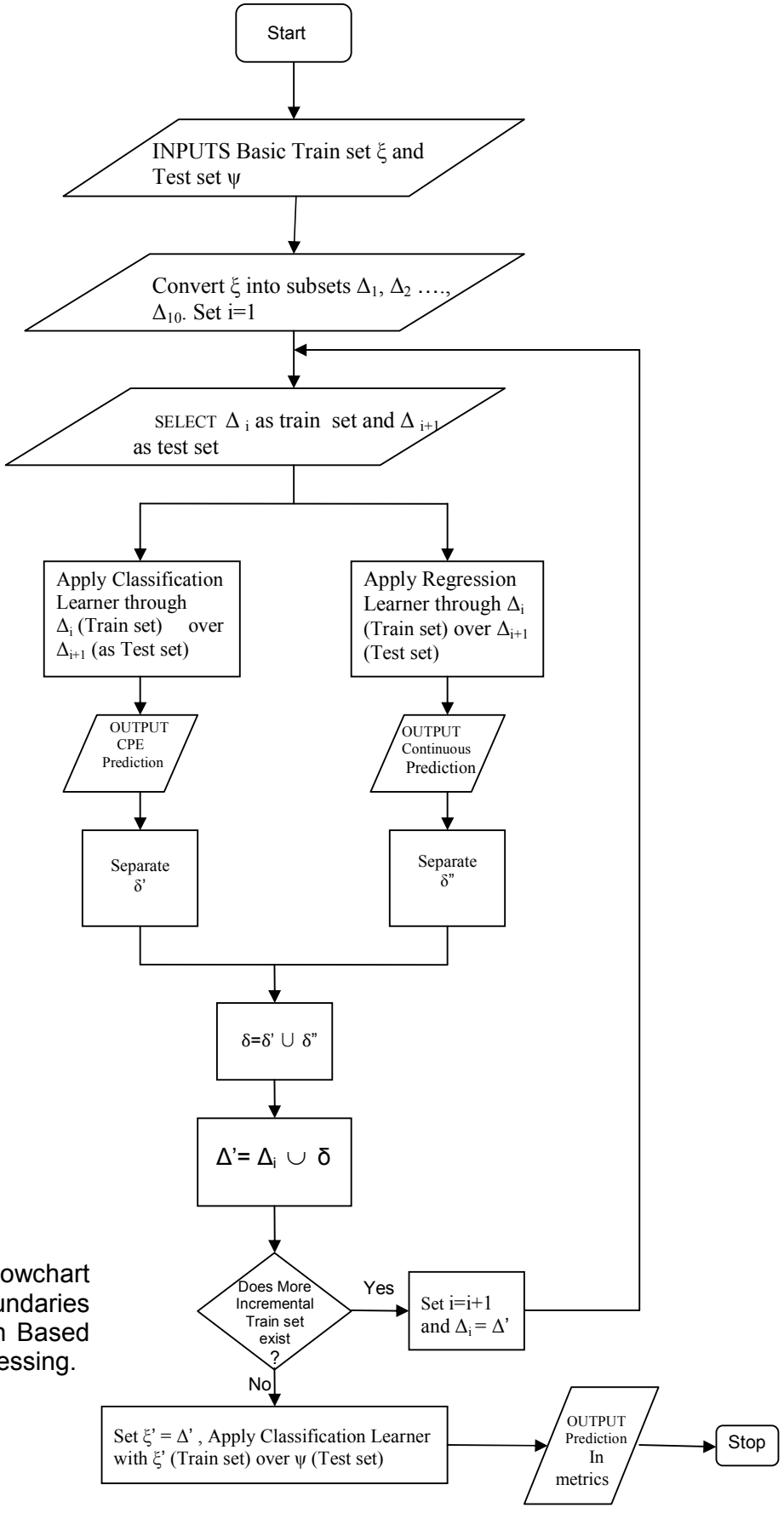

Figure 4.8: Flowchart for 'fuzzy boundaries of Regression Based clusters' processing.

4.6 Composite effect of Classification algorithms and Regression based Clusters

Proposed algorithm prunes data on the basis of overlapping clusters formed through the output of classification and regression algorithms. Data scattered on frequency charts through this clustering is represented in Figures 4.5, 4.6 and 4.7. There is a need to analyze data scattered in different regions of these frequency charts more closely. More close analysis of this scattered data is as follows:

4.6.1 Data common to Correctly Classified and under strong confidence range through regression

Data related to clusters formed by classification output and regression output towards extremes of combined frequency graph includes data predicted with great confidence by the model trained with training data used. This data is identified to be confidently predicted correctly by classification and regression algorithms implying their confidence to be tested qualitatively as well as quantitatively. This data is very confidently classified with current learning, so may be excluded from training of data through such data.

4.6.2 Data common to misclassified instances

Misclassified data need to be investigated more, so that model can be trained with such data to classify accurately. Misclassified data contains anomalies as well. So training of algorithm with anomalies can be disastrous as well. Incremental investigations can be quite helpful in identifying anomalies as well.

4.6.3 Unique data from misclassified instances

Using classification as well as regression algorithms may analyze a part of the test data differently. As classification algorithms are better classifiers than regression algorithms, so classification algorithms generate relatively less misclassified data as compared to regression algorithm. Certain part of misclassified data will be common to both outcomes. Common misclassified data undoubtedly demands further investigation, but misclassified output individually related to each algorithm is also needed to be investigated closely. As misclassified data of classification algorithm must help classification algorithm to be trained well, so this data is needed to be included in the train set, whereas misclassified data of regression algorithm is not misclassified by classification algorithm, so this data is also needed to be investigated more closely as

confidence in such data is not sufficient quantitatively. It implies that union of misclassified data from classification as well as regression algorithms need to be investigated more closely.

4.6.4 Rightly classified data but with Poor confidence

Data rightly classified but lying within fuzzy boundaries of combined graph from the outputs of classification and regression algorithms needs to be investigated more closely. As data within fuzzy boundaries and classified accurately is rightly classified, but confidence about its outcome is not so strong. So this data needs to be investigated more closely.

4.6.5 Strongly known but poorly defined by regression

There will be certain part of data for which classification algorithm provides strong probability estimate but regression algorithm is unable to predict it so confidently. Such data needs to be investigated more closely. There is a need to investigate lacking of such data to be predicted with greater confidence quantitatively by regression algorithm.

4.6.6 Other shifted data

There may be a lot of movement of data from different regions of data in frequency chart. Entire such data needs to be investigated closely as confidence in such data differs significantly as it is tested for classification or regression algorithm.

4.6.7 Issue related to eliminated data

With proposed process, data with strong confidence is eliminated during different iterations, because confidence of learning in current iteration is very strong. But issue arises when rest of the data is used for training, trains algorithm for itself. It may lead to weakening of confidence in data being eliminated from the train set and in later iterations same data may not be having that much confidence. Such issue can be justified with the argument that if such data weakens in later stages then such data will be included in the train set as this data will jump to fuzzy range and will be included in the forthcoming train set. So this issue will not arise up to last iteration. In last iteration, data close to eliminated data will not get chance to be included in the train set, but elimination of this data is based upon confident training of algorithm and algorithm will not loose its

confidence so quickly. So such issues should not affect the performance of algorithm even after reducing the size of the train set.

If we compare the categorization of proposed process, it can be placed under complex processing as explained in section 3.1.5 of chapter 3. Results of experimental work related to six previously known processes and proposed process will be presented in Chapter 6. Experimental work will include twenty algorithms for processing of all the processes. Now question arises that there are so many processes as well as algorithms for experimentation. It is previously known that no individual algorithm or processes can fulfill all the needs of a problem at hand. There is a need to analyze more than one process/algorithm to justify the outcome. It creates a big problem before knowledge engineers about development and maintenance of knowledge based information systems. Present research work also concentrates for finding a system development model that can help to meet the dynamism involved in developing knowledge based systems, is discussed in next chapter.

4.7 Conclusion

In this chapter, a new process is proposed for supervised learning based on classification algorithm, regression algorithm, fuzzy sets and clustering for classification tasks. Limitations of previously known processes are identified and this new process is developed to overcome the limitations of earlier processes. Algorithm of proposed process is included in this chapter and different issues and their justifications are also included.

Chapter 5

‘Genetic Information System Development and Maintenance’ Model

Knowledge based information system are need of current era. Increasing sizes of databases has generated this need for efficient development of such systems. In previous two chapters many processes were discussed for supervised learning and twenty algorithms will be applied on these processes over ten databases collected from internationally renowned organizations. These many processes are highly dynamic and demand a support from software engineering in development and maintenance of such system in real practice. In this chapter a model is presented to support software and knowledge engineers in developing and maintaining such complex systems efficiently. Before presenting model relationship between an organization and information systems is explored in next section followed by proposed model for system development and maintenance of knowledge based information systems. Presented model is not limited to knowledge based systems, but is also applicable to software application development and maintenance of other organizational information systems, so proposed model will explore all corners of its utility in real world.

5.1 Relationship between Organization and Information System

Information systems are cores of business organizations. Flow of Information in an organization is as if blood floats in veins of human body. Information system takes birth with the birth of an organization, it grows as the organization grows and stays alive forever with the organization or up to the death of the organization. Growth of organization directly affects the growth of information systems, changes in organizations create the need for changes in information systems as well, e.g. when a firm is incorporated, its operations are limited to small areas and a small number of people. Consequently, information system is also small and can be handled manually or with little automation by small number of people. Once a company starts expanding its operations from one place to other places and from one country to many other countries, information system is also expanded concurrently to handle the organizational information needs. Even after many developments, software development and maintenance are very tedious tasks. Many problems generated due to information systems, faced by the organizations, can be listed as budget of information system, selection of technology, selection of system development companies, duration of system development project, change identification, vision for future changes, flexibility of information system to incorporate future changes, level of changes, maintenance terms of the systems, system improvement policies, control over system development and maintenance, Data security, Fulfillment of right kind of user needs, Connectivity of different users etc. Proposed model is an effort to find the solution of many of the problems listed above. First of all, there is a need to change the view about requirement of change(s) in information systems. Business organizations view change requirements of information system as burden for their organization. Organizational members especially need to adopt change requirements of information systems as inherent necessity of their information system. Secondly vision of organizational members should be global and the initial selections of technology need to support global expansion of information systems. Nevertheless to remark that Internet based technologies are the future of information systems. All organizations today need to adopt online technologies that support information exchange worldwide. Thirdly the selection of system development model and processes should support frequent change requirements, software reuse and controlled maintainability of information systems. An information system survives with organization, changes occurring in organization need to change information systems. If information systems reflect the organization, then how information needs of an information system may be understood and

implemented in a short span of system development project. There is a fundamental need to develop information systems continuously throughout the life of organization, so that required changes can be implemented effectively and system architecture must support such requirement. In following sections we propose a system development process and a model to develop as well as maintain the information systems continuously throughout the life of an organization.

5.2 'Genetic Information System Development and Maintenance' Model

System development and maintainability can be achieved efficiently, if it is under direct control of organization's management. Management of organization can control their own employees far better than the employees of other organizations. Organizing a team of organization's employees and flexible system development process are the ingredients of proposed 'Genetic Information System Development and Maintenance' model as explained below.

5.2.1 Team Organization for System Development and Maintenance

First of all, there is a need to have a group of personnel (optimum number of people) within organization to analyze, develop, test, implement and maintain the information system for their own organization throughout the life of an organization. This group of people will continuously look forward for the information/knowledge requirements of the management personnel continuously and build the system or incorporate the changes continuously, without affecting the earlier implemented system at large, but only affected parts will be required to be updated, implemented, documented and only concerned people will be informed about the changes. This group of people will not only look after the information requirements of the management personnel or the public concerned, but also look forward for the technological developments worldwide, so that their organization can be benefited through the involvement of latest technological developments. More people, expertise and organizations may be hired, contracted or outsourced to help the development, whenever it is necessary.

5.2.2 Development and Maintenance of the System

There is an additional need to develop the system by iterating the following system development steps throughout the life of an organization continuously:

STEP I: Genetic Creation of Processes/Services and Sub Processes

First of all there is a need to identify information requirements from scratch or from previously implemented major processes/services (P_k) and sub processes (SP_{kl}) of the organization and to program them to meet the requirements by formulating a library (L) of processes/services and sub processes i.e.

$$L = \{P_1, P_2, ..., P_n\} \quad ..... (5.1)$$

where $P_k = \{SP_{k1}, SP_{k2}, ..., SP_{kl}\}, \quad \forall k = 1 \; to \; n, \quad l \geq 1,$

$'n' \; and \; 'l' \; are \; \text{variables}.$ Number of processes/services and sub processes will be finite and almost different for all the processes, whereas maximum value of $'l'$ and 'n' for the number of sub processes and processes respectively will be dependent upon the decision of system analyst. Processes/Services and sub processes will work as genes for the genetic development process/service. The goal of design (from a low-level perspective) is to minimize coupling and maximize cohesion (Kramer et al. [120]), can only be achieved from the division of different information needs in independent categories of processes/services. A process/service is defined as a set of sub processes and will fulfill information requirements through execution of proper logic and will provide access to the authenticated database(s). Here term process is used as synonym to a service. Genetic development means the primary set or sequence of processes that lead to the formation and subsequent development of processes/sub processes. Development of processes is carried out at three different levels as follows:

(i) User Level: Information requirements are identified at this level, so system development process actually begins at this level. First of all, users' information/knowledge requirements need to be identified. Once users' information requirements are identified, further level can be sequentially explored. List of identified information/knowledge requirements needs to be reviewed successively.

(ii) Database level: Logical designs of schema/subschema need to be developed and implemented according to the user requirements and the organizational considerations. Schema / Subschema need to be updated according to the initially defined/emerging/modified information/knowledge requirements. Detailed design considerations need to be defined so that less frequent changes are required later. Technology used for System development needs to be adaptive for any kinds of changes from logical to physical memory levels.

(iii) Programming Level: Process development is completed by converting user requirements into programming code and connecting databases of significant concern by developing programs and logic to find practical solutions of user requirements/problems. Genetic development effort is constantly required to fulfill the user requirements from the databases through programming.

The set of processes or a library created during entire genetic creation of processes will require different processes to undergo fitness screening. The fitness screening of processes/sub processes need to be implemented during every review of processes/sub processes.

Function **f** for fitness of processes/sub processes (based on mutation, selection and reproduction etc.) (Darwin [42]; Forrest [57]) need to be applied on set $L^{(i)}$ to get $L^{(i+1)}$ i.e. Set of Processes after qualifying fitness criteria.

$$\mathbf{f}: L^{(i)} \longrightarrow L^{(i+1)} \qquad \ldots\ldots (5.2)$$

Where $i \geq 0$ represents (i+1)th generation/iteration/review of process/sub process development and $L^{(i+1)}$ will become a set

$$L^{(i+1)} = \{P_1, P_2, \ldots\ldots, P_{n'}\}^{(i+1)} \qquad \ldots\ldots (5.3)$$

Having n' number of processes with their sub processes, after review in current generation. At the time when new system is developed from scratch library L will be developed by newly developed processes, whereas in next generations this library will undergo development and fitness processing by revising earlier processes or by including new processes. After screening of the processes, n is assigned the value of n' and procedure enters to next step.

STEP II: Implementation of Quality Parameters

Each process and sub process needs to undergo quality screening based on parameters set for quality assurance and implementation as follows:

Function '**q'** Quality parameters need to be applied on the set L to get QL i.e. Quality Library.

$$\mathbf{q}: L \rightarrow QL \qquad \ldots\ldots (5.4)$$

Where QL will become a set $QL = \{QP_1, QP_2, ..., QP_n\}$ of Quality processes and sub processes. Parameters of quality may be based on quality standards, syntactical, logical, environmental and organizational factors etc. Quality parameters are usually dependent upon system analysts' vision about the quality and differ from system to system and place to place.

STEP III: Customization- Now Quality Library needs to be customized according to the needs of individual users as follows. Function '**c'** of Customization needs to be applied on QL to give a set CQL.

$$\mathbf{c}: QL \rightarrow CQL \qquad \ldots\ldots (5.5)$$

Where CQL will become a set $CQL = \{CQP_1, CQP_2, ..., CQP_n\}$ of Customized Quality processes and sub processes. Same module may be customized differently for different users, e.g. finance related data may be more detailed for financial experts, whereas summarized for the marketing analyst. Customization of processes depend upon type of user, user's level in organizational hierarchy, authority etc.

STEP IV: Security- Now security criteria need to be applied on Customized Quality Processes for the secure access of the organizational database(s).Function '**s'** of Security criteria need to be applied on CQL to generate a set CSQL.

$$\mathbf{s}: CQL \rightarrow SCQL \qquad \ldots\ldots (5.6)$$

Where SCQL will become a set $SCQL = \{SCQP_1, SCQP_2, ..., SCQP_n\}$ of Secure Customized Quality processes and sub processes, which may be assigned to the authorized users through their login accounts. Security requirements of different processes may be different, e.g. confidential organizational resources demand more secure access, rather public information resources.

STEP V: Total Quality Management (TQM): It is a philosophy toward continually improving your business and products (Ishikawa [99], Hyde [97]). Information systems need to be improved continuously. Organizational team of system development personnel needs to search for better options for the organizational information system and then to implement them in the system. These options include the comprehensive search for better technology, better processes, enhancement of earlier processes, search for new information needs, other problems related to earlier implementation.

These five steps need to be iterated continuously forever. Following flowchart includes the above discussed system development and maintenance procedure with a start and continuous development, but without any stopping criteria.

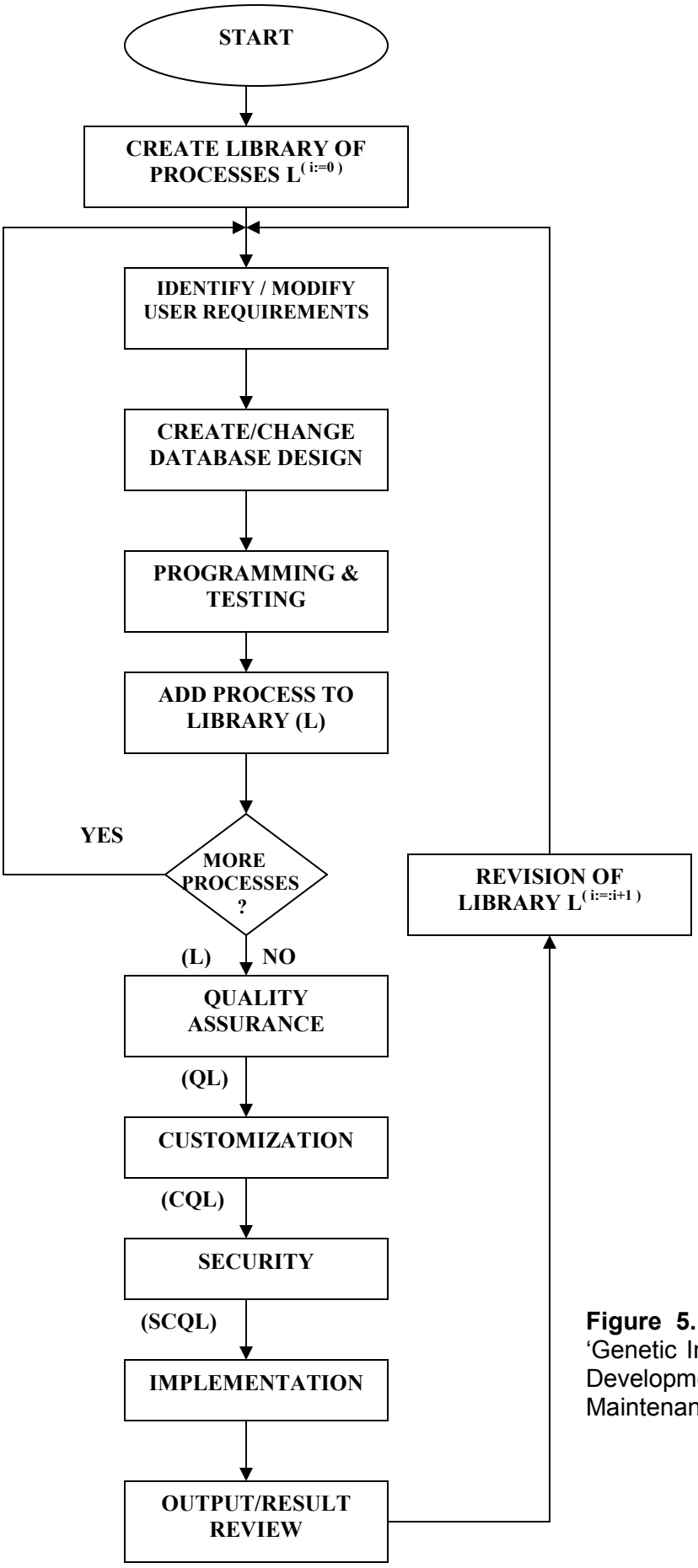

Figure 5.1: Flow Chart for 'Genetic Information System Development and Maintenance' model.

Proposed system development will be a combined and continuous effort of organization personnel (users), system analysts, programmers and Database Administrator (DBA) to develop, implement and maintain information system and business processes. System analyst need to identify organizational structure, personnel, subordinates and their information requirements etc. continuously, DBA need to be involved simultaneously for authorizing required database access to different personnel. Management is involved for identifying their personalized information requirements and its implementation in customized way.

5.3 Proposed model for system development

Different users have different type of information requirements. Type of user, user's level in organizational hierarchy, his/her authority, technical ability of system to fulfill users' information requirements etc. are the factors that play an important role in differentiating different users and their information needs. Categorizing users based on their common information requirements is always a better option, e.g. grouping users on departmental level, common work level etc. Users with common information needs may be grouped together to fulfill their information needs, whereas their authority in organizational hierarchy may help to distinguish their access to the information resources, e.g. marketing department may be grouped together to fulfill their information needs related to marketing aspects. Nevertheless, higher-level managers will have higher as well as summarized access to the databases as compared to the low level executives, who usually have lesser but detailed access to the databases. Processing of information may be distinguished by different type of processing set-ups like one-dimensional data processing, multi-dimensional data processing and knowledge discovery in databases (i.e. data mining or vaguely defined information requirements) etc. Different sets of information requirements demand different kind of processing set-ups and different databases to be connected, e.g. marketing executives may require many databases to be accessed worldwide to get latest information and different type of processing set-ups, so that right kind of information is accessed. On the other hand, financial people require information related to fund flow, sales and expenditures only. Such information requirements, processing set-ups and database connectivity necessitate the development of processes/sub processes for the extraction of required information. Each process is developed to fulfill a homogeneous/related set of users' requirements. Domain of Individual process need to be defined very precisely, so that each process is connected to the databases concerned with its own domain and client/server side logic, security guidelines, quality standards etc. may be settled according to these homogeneous information requirements. Different processes (i.e. heterogeneous set of

processes) are combined parallel to each other, with pre-defined accessibility. Implementation of whole system can be explained as a user account login/password authentication based access of a World Wide Web based information system. All management personnel are allocated their personal accounts, which is a page if we talk in terms of World Wide Web. Each user is authenticated through login for its access from the website of the organization. Each account page is a set of windows or drop down menus having some predefined templates for enquiries, access to the database(s) and information transfer (General and specific information) as shown in Figure 5.2.

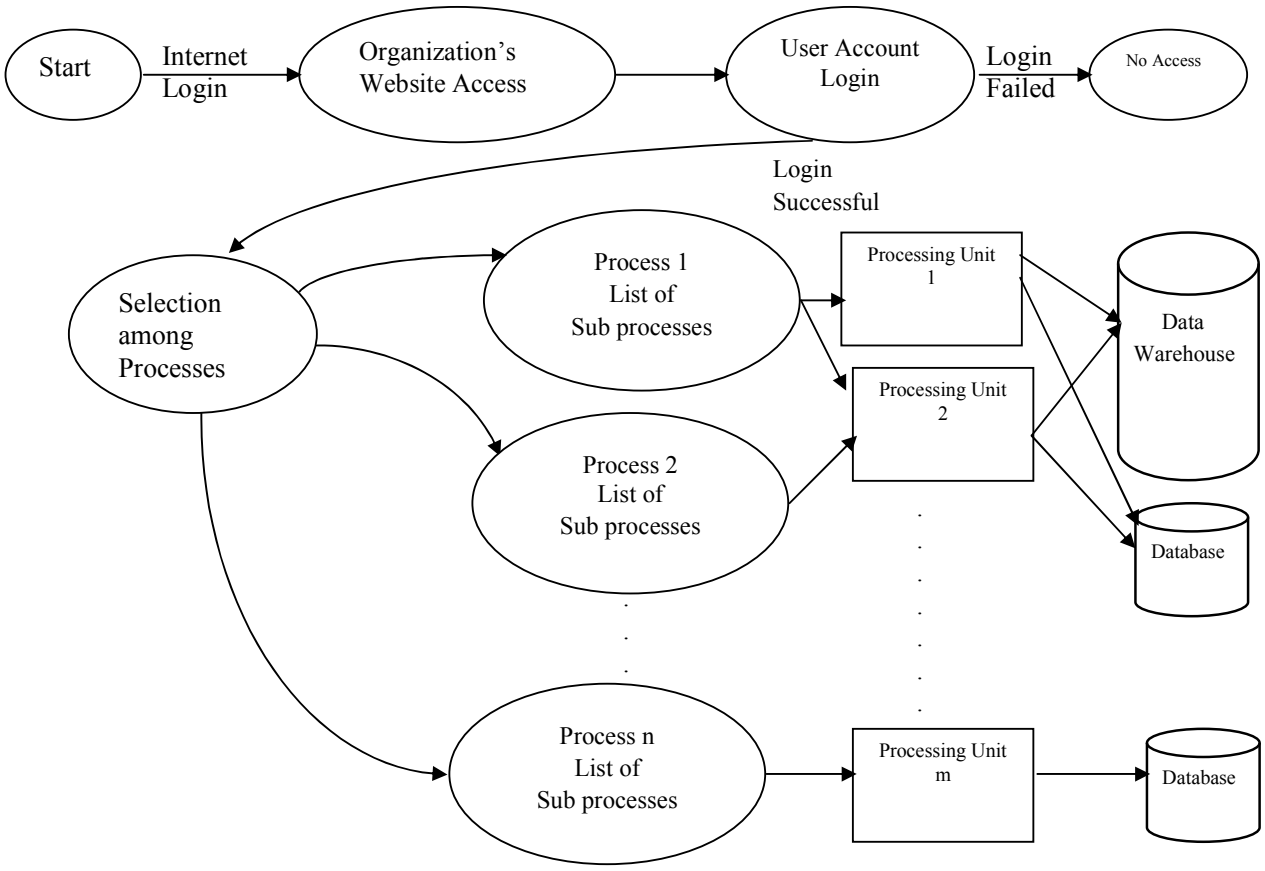

Figure 5.2: Accessing User account through Internet or Intranet

Programs developed by programmers, are maintained by DBA's library of processes and sub processes meant for the purpose. User may access, query, modify and update database according to the authority provided by DBA. Continually up gradation of library programs is required to fulfill the newly identified information requirements, facilities made available by innovations and new technology. Data warehouse is maintained at back end to collect the information from different sources scattered worldwide. User may access its account by connecting from any part of the world through Internet. Each user

is provided information from internal and external sources of information of the organization. Key process areas need to be identified and information is collected in central database to furnish the required information, so that one may acquire up-to-date information related to the organization. Different internal/external sources of information are identified and bundled together to furnish the information from external environment of the organization. Key process areas are those critical operations upon which success of any business is dependent. Key Process Areas of any organization may be among Sales, Advertisement, Logistics, Raw Material, Labour, Finance or Production etc., whereas external sources of Information Government Agencies, Competitors, Industry, Suppliers, Customers and many others. These key success factors can be identified and information related to them can be furnished to the user according to their authority. External sources can be identified and their information may be made available directly or after required processing to the user. Organization's Information System Development Team of Analysts, DBAs, Programmers need to work together for identifying emerging information needs of users, considers the matter of inclusion of new technology etc. continuously. Most important aspect of this model is its open behaviour. Processes and sub processes are combined with their logic and access. Individual process is a complete system in all respects having required accessibility, functionality and logic as shown in Figure 5.3.

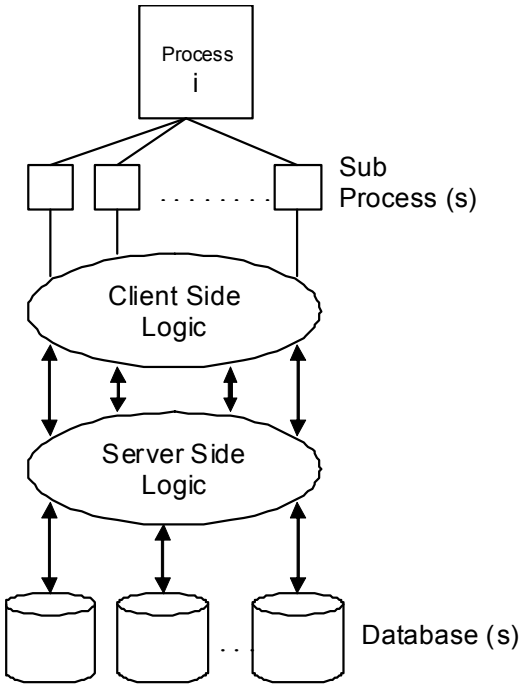

Figure 5.3: Connectivity of user to database through process/ sub process.

When a user is authorized to access any process or sub process, that process/ sub process is added to user's account, removal of process(s) or sub processes can be done by removing those from the user account. Different types of database environments may be combined to give access to the user. If data is to be accessed from two or more different type database environments and minimum access time is needed, then these may be combined in organizational data warehouse on regular basis. If real time access is required, then remote database(s) can be accessed with small delays as well. Providing heterogeneous access of database(s) to the user solves problem of adaptation of new technology and processes. Thorough documentation is needed for the development and updating the set of processes, so that software reuse is maximized. Whenever a change is implemented, affected areas may be rectified and concerned documentation is done. In this way, problems related to maintenance might be controlled. Desired results can only be achieved, if the development and maintenance is carried out continuously. The set of processes or library is carried forward from current generation to next generation, after implementing the required modifications. Processes may undergo minor modification of its sub-processes, major modification of processes/sub processes, exclusion of complete processes (i.e. discarding complete processes) or may require no change in earlier implementation as shown in Figure 5.4.

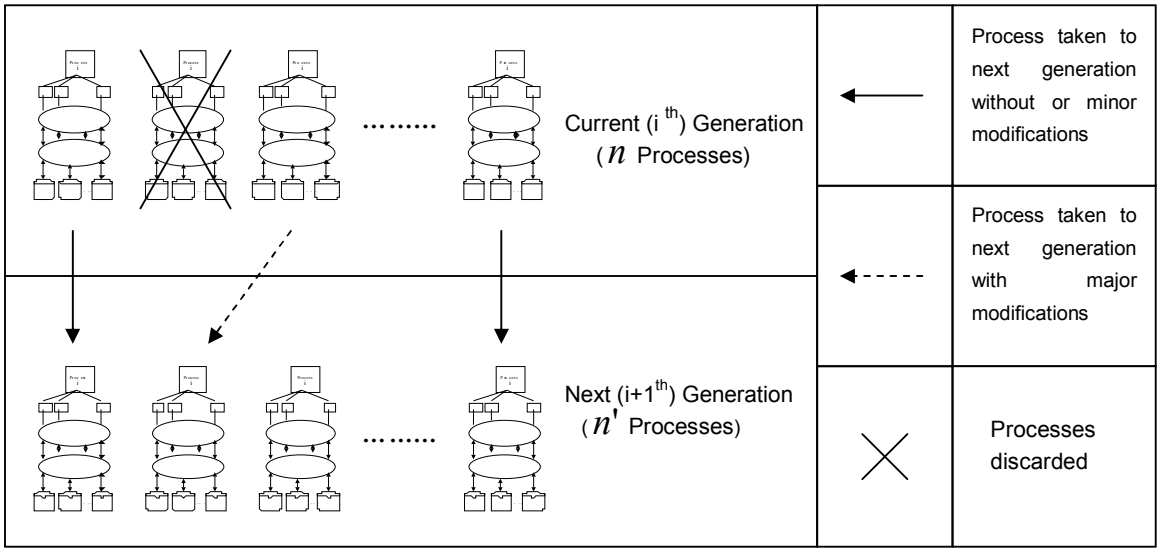

Figure 5.4: Transition of processes from current generation to next generation.

5.4 Effectiveness of proposed model

Organizational employees can understand and identify information requirements of their organization far better than the professionals hired for system development from outside organizations. People from outside agencies may be involved for technical expertise for software development, but Genetic Information System Development require involving more own organization's employees as system developers. Different problems faced during software development and support provided by proposed model is as follows:

(i) **Cost:** Software development is a process of major concern, but a major chunk of total software cost is consumed in maintenance. 'Genetic Information System Development and Maintenance' model cost almost same expenditure, but it helps in developing personalized software, enhancing software reuse and early identification as well as improvement of newly generated information needs on continuous basis, which is profitable in long run. It is due to the reason that the organizational people may understand their organization better than anyone else from outside agency and continuous review enhances the system performance.

(ii) **Functionality:** Customized or personalized system development helps in achieving higher user satisfaction, as user requirements are understood more closely and system is updated according to these requirements.

(iii) **Reliability:** When user needs are understood well and are improved whenever changes are required, testing of system in real environment gives maximum fruitful results, if processes are implemented in a well manner and reliability is enhanced.

(iv) **Maintainability:** 'Genetic Information System Development and Maintenance' model inherently concentrate on the maintainability of the system.

(v) **Adaptability:** System is opened to involve any type of technology and access of databases may be provided directly to the user or it may be connected to the centralized database and suitable changes may be implemented in concerned programs or logic of the system for different users.

(vi) **Efficiency:** Online/Internet based systems are as good as any offline systems. In case of Online Analytical Processing or Data Mining processing, a small delay in responses may be due to inability of state-of the-art technology to meet desired response time, but still current technology can meet all kinds of information requirements with negligible delays. Open connectivity of improved technology and continuous maintenance of system can help to achieve the desired efficiency.

(vii) **Portability and Reuse:** Internet connectivity is available with almost all type of technologies with little modifications, so portability is up to the mark for the system. Proposed model sufficiently assures the requirement of proper reuse of software.

(viii) **Usability:** Users may be trained by simple training programs to use the system and system needs to be developed more intuitively. Different activities of the user pages may be combined in separate windows or drop down menus and details can be accessed through in-depth navigation of the window or connected pages.

(ix) **Security:** Security of data is always a biggest question, while accessing Internet or online processing. Access to the database need to be secured and multiple levels of security layers may be involved, where authentication is required. Security levels may differ from process to process or database to database as per the requirement.

(x) **Quality:** If above discussed parameters are successfully achieved, then quality of information system is assured, not only from the outputs of the system, but also from quality standards defined for the purpose as well.

Proposed model is implemented in many organizations operating in a variety of business domains. 'Genetic Information System Development and Maintenance' model is helpful in finding the solution of many problems related to software development discussed above. But in present thesis, investigations for testing the model are guided towards reliability, maintainability and costs involved. Case studies are used to concentrate the investigations toward these parameters. The proposed model indicates improvement in adaptive and perfective maintenance, so further investigations need focus on corrective maintenance/change management at low costs. Chapter 6 will discuss these case studies for investigations in these directions.

5.5 Conclusion

In this chapter, a new model is proposed for highly dynamic knowledge based information system development. Dynamism involved in developing knowledge based systems is analyzed and accordingly the 'Genetic Information System Development and Maintenance' model is proposed. Proposed system advocates need for organizing and maintaining a team of software developers within the organization for which system is to be developed. Further, this model recommends the need of system development and maintenance activities to run simultaneously throughout the life of an organization. Effectiveness indicators of proposed model over different parameters are also included in this chapter.

PART III

Chapter 6 **Experimentation 67-83**

Chapter 7 **Results, Interpretations and Evaluation 84-131**

Chapter 8 **Conclusions and Future Research 132-141**

Chapter 6

Experimentation

This chapter presents the experimental setup of present work. As present research contributes a new process for supervised learning and a model for developing knowledge based information systems, so experimentation is also categorized in same manner. First section of this chapter explains the experimental setup for previously known six processes as well as for new process proposed by this research. It includes the description of all datasets, algorithms and metrics used for study. Second section of this chapter introduces the case studies used for investigating the effectiveness of proposed model for knowledge based 'Genetic Information System Development and Maintenance' model.

6.1 Experimental Setup for Supervised Learning Processes

In this section the experimental setup is explained for experimentation performed for present study. This section describes the datasets, algorithms, methods and metrics used for experimentation in present study. Reasons behind selecting these datasets, algorithms and metrics are also explained with their descriptions.

6.1.1 Datasets

Present research compares all supervised learning processes on ten binary classification problems. These datasets were selected very carefully such that these include a variety of applications and most of them are having benchmark results available for comparison with results of present research.

(i) **Adult:** The task is to predict if an individual's annual income exceeds $50,000 based on census data. This dataset is also known as "Census Income" dataset. Extraction of this dataset was done from 1994 United States Census database. Number of instances in this dataset are 48842 and number of attributes are 16.

(ii) **Covtype:** The forest cover type for 30 x 30 meter cells obtained from United States Forest Service (USFS) Region 2 Resource Information System (RIS) data. Independent variables were derived from data originally obtained from United States Geological Survey (USGS) and USFS data. Data is in raw form (not scaled) and contains binary (0 or 1) columns of data for qualitative independent variables (wilderness areas and soil types). Dataset has 54 attributes and numbers of instances in dataset were 581012, out of which 50000 instances were used for present research. This dataset had been converted to a binary problem by treating the largest four classes as positives (i.e. 1) and the rest three as negatives (i.e. 0).

(iii) **DS100:** The ds1.100 dataset is a compressed life sciences dataset. Each row of the original, expanded dataset represented a chemistry or biology experiment, and the output represented the reactivity of the compound observed in the experiment.

(iv) **Letter:** The task in this dataset is to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique instances and 17 attributes. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15. This dataset had been converted by replacing alphabets A-M as negatives and N-Z as positives.

(v) **Oracle:** This dataset is a collection of observations related to potential customer identification problem. Number of instances in this dataset are 8000 and number of attributes are 27. This dataset had been converted into binary problem by replacing top two classes (Very High and High) into positive class whereas lower two into negative class (Medium and Low).

(vi) **Pen Digits:** This dataset was created by collecting 250 samples from 44 writers. A pressure sensitive tablet with an integrated LCD display and a cordless stylus were used for collecting data. The input and display areas were located in the same place. It allowed the collection of handwriting samples. The tablet used to send x and y tablet coordinates and pressure level values of the pen at fixed time intervals (sampling rate) of 100 miliseconds. Number of instances in this dataset are 10992 and number of attributes are 16. This dataset had been converted by replacing top five digits (5 to 9) into positive class whereas lower five into negative class (0 to 5).

(vii) **Physics:** The goal in this task is to learn a classification rule that differentiates between two types of particles generated in high energy colliding experiments. It is a binary classification problem with 78 attributes and numbers of instances were 150000, out of which 50000 instances were used for present research.

(viii) **Satellite:** This dataset consists of the multi-spectral values of pixels in 3x3 neighborhoods in a satellite image and the classification associated with the central pixel in each neighborhood. The aim is to predict this classification, given the multi-spectral values. In the sample dataset, the class of a pixel is coded as a number. Number of instances in this dataset are 6435 and number of attributes are 36.

(ix) **Shuttle:** This dataset is a used to generate comprehendable rules for determining the conditions under which an auto landing would be preferable to manual control of the spacecraft. Number of instances in this dataset are and number of attributes are 16. This dataset had been converted to a binary problem by treating largest two classes as positives and rest three classes as negatives. This dataset had been converted into binary dataset by converting largest three classes (i. e. 4, 5, 7) as positives (class 6 was absent), whereas smallest three classes as negatives (i.e. 1, 2, 3).

(x) **Tic2000:** This data set contains information on customers of an insurance company. The data consists of 86 variables and includes product usage data and socio-demographic data derived from zip area codes. The data was collected to predict that who would be interested in buying a caravan insurance policy and give an explanation for that.

Adult, Cov_Type, Letter, Pen_Digits, Physics, Shuttle, Satellite and Tic2000 are the problems from UCI repositories (Blake et al. [21]). Oracle dataset is procured from Oracle data mining problems from Oracle website. DS1_100 is procured from (Komarek et al. [117]). Table 6.1 includes brief description about the datasets.

Problem	Number of Attributes	Size of Datasets		
		Train Set	Test Set	Total
Adult	14	10000	38842	48842
Cov_Type	54	10000	40000	50000
Ds1_100	100	10000	16734	26734
Letter	16	10000	10000	20000
Oracle	27	8000	7342	15342
Pen_Digits	16	5000	5992	10992
Physics	78	10000	40000	50000
Satellite	36	3000	3435	6435
Shuttle	9	10000	40000	50000
Tic2000	85	5000	4822	9822

Table 6.1: Description of Problems/Datasets Used for present research

6.1.2 Processes used for Experimentation

Experimentation is the most important part of any empirical study. In present research all well known ways of experimentation that are developed so far for supervised learning have been included for comparing the results with the outcome of proposed method. Previously known methods/processes can be categorized under Fixed Split, Cross Validation and Calibration.

6.1.2.1 Fixed Split

Simplest form of experimentation is to divide dataset into two fixed length datasets of the train set and the test set. This kind of experimentation is meant for simple testing of

algorithms. Biggest problem with fixed split is over fitting of training data e.g. tree based techniques may have too many branches that may reflect anomalies and result in poor accuracy of unseen samples. To overcome the limitations of fixed split two approaches used are prepruning and post pruning. Prepruning is performed through cross-validation, whereas many calibration methods have been proposed for post pruning. Following subsections include the discussion about these methods.

6.1.2.2 Cross Validation

To evaluate the robustness of the classifier, the normal methodology is to perform cross validation on the classifier. Ten fold cross validation has been proved to be statistically good enough in evaluating the performance of the classifier (Witten et al. [189]). For present study datasets are divided into the train and the test sets. Then this train set is equally divided into 10 different subsets for ten fold cross validation. Nine out of ten train subsets are used to train the learner and the tenth subset is used as the test set. The procedure is repeated ten times, with a different subset being used as the test set. In this way cross validation is performed to calibrate the models and select the best parameters and then models are applied on the large Final test set.

6.1.2.3 Calibration Methods

Many learning algorithms do not predict probabilities. For example the outputs of an SVM are normalized distances to the decision boundary, whereas naive bayes models are known to predict poorly calibrated probabilities, because of the unrealistic independence assumption.

A number of methods have been proposed for mapping predictions to posterior probabilities. Platt Scaling (Platt [158]) is used for transforming SVM predictions to posterior probabilities by passing them through a sigmoid. Platt scaling also works well for boosted trees and boosted stumps (Niculescu-Mizil et al. [150]). A sigmoid is also not the correct transformation for all learning algorithms.

Second method used for calibration is Logistic regression. Logit Boost algorithm is used for performing additive logistic regression. This algorithm performs classification using a

regression scheme as the base learner, and can handle multi-class problems (Friedman et al. [63]) and it can also do efficient internal cross-validation to determine appropriate number of iterations.

Other method generally used for calibration is Isotonic Regression (Zadrozny et al. [194,195]; Robertson et al. [164]). It is used to calibrate predictions from SVMs, naive bayes, boosted naive bayes, and decision trees. Isotonic Regression is more general method, but its only restriction is that the mapping function used is isotonic (monotonically increasing). A standard algorithm for Isotonic Regression that finds a piecewise constant solution in linear time, is the pair-adjacent violators (PAV) algorithm (Ayer et al. [12]).

6.1.2.4 Fuzzy Boundaries of Regression Based Clusters

This processing method is contributed by present research. In proposed process the train set is divided into ten parts (i. e. Δ_1, Δ_2 …., Δ_{10}). First subset of the train set (Δ_1) is used to train the classification and the regression algorithms in parallel. Models generated from these algorithms are applied in parallel, on second subset of the train set (Δ_2) to generate outputs. Classification algorithm's outputs are generated in the form of class probability estimate (CPE), whereas regression algorithm's output is always a set of floating values scattered over a range. Frequency charts are generated for the outputs of both kinds of algorithms. Frequency charts of both algorithms are plotted on single graph. Combined frequency chart is used to identify data near extremes with high confidence through fuzzy boundaries of regression based clusters. Data with higher degree of confidence is excluded from the second subset (Δ_2) of dataset that is used as the test set in this iteration. Remaining data of second subset of the train set (δ), after excluding data with higher degrees of confidence, is added into first subset of data i. e. $\Delta_2' = \Delta_1 + \delta$. This set ($\Delta_2'$) of data will have less than twenty percent of the train set. Now this data (Δ_2') is used to train the classification and the regression algorithms in parallel and third subset (Δ_3) of data is used to test the model generated from these algorithms. Data with greater confidence is excluded from the third subset of data and is added into previous data having data from first two subsets. Like this we continue to include data in the train set incrementally and ultimately we get a train set (Δ_{10}') having less than hundred percent of the actual train set. Now this train set (Δ_{10}') is applied on the actual test set (ψ) to generate output. Twenty classification algorithms and five regression algorithms were selected for the processing.

Five algorithms were selected to find more appropriate regression algorithm among many available options.

6.1.3 Algorithms used for Experimentation

Twenty classification algorithms and five regression algorithms were selected for the experimentation. These algorithms are described in this section.

6.1.3.1 Classification Algorithms

Classification algorithms were selected through preliminary tests performed over Adult dataset. There were two criteria that were used for selecting these algorithms. First criterion was to select at least one algorithm from each broader category of algorithms as explained in section 3.2 of chapter 3. Second criterion was based on preliminary tests performed on Adult dataset. All better performing algorithms were selected at first instance and then twenty algorithms were finalized for experimentation. These twenty algorithms were used for major experimentation work viz. Fixed Split, Cross Validation Platt Scaling and Five Fuzzy Cluster based processes. Out of these twenty algorithms, five classification algorithms were used for minor study to perform experiments on all processes of supervised learning i. e. Fixed Split, Cross Validation Platt Scaling, AdaBoostM1, Additive Regression, Stacking through Isotonic Regression and Five Fuzzy Cluster based processes. Different classification algorithms used for the experimentation of present study are described as follows:

(i) **ADTree:** ADTree is alternating decision tree learning algorithm (Freund et al. [62]). In ADTRee algorithm, induction of the trees has been optimized and heuristic search methods have been introduced to speed learning. This algorithm was allowed expand its nodes infinitely. Internal boosting iterations were set to 10.

(ii) **BayesNetGenerator:** This algorithm is based on Bayes Network (Friedman et al. [64]) learning using various search algorithms and quality measures. Base class for a Bayes Network classifier, provides data structures (network structure, conditional probability distributions, etc.). Number of nodes and arcs were allowed at maximum to be 10.

(iii) **Decision-Stump:** A Stump is simply a one-level decision tree. Decision Stump (Iba et al. [98]) is usually used in conjunction with a boosting algorithm. It does regression (based on mean-squared error) or classification (based on entropy). By default level one is allowed.

(iv) **Decision-Table:** This classifier is based on building and using a simple decision table majority. Decision table (Kohavi [116]) with a default rule mapping to the majority class is used to classify instances in discrete spaces with accuracy that is sometimes higher than induction algorithms. In this algorithm number of rules to be considered before terminating best first search was set to be 5.

(v) **IB-k:** This classifier is k-nearest neighbor classifier (Aha et al. [6]). It can select appropriate value of K based on cross-validation and also can do distance weighting. It is common to select *k* small and odd to break ties (typically 1, 3 or 5). Larger *k* values help reduce the effects of noisy points within the train set and the choice of *k* is often performed through cross-validation. Different parameters were set as follows: Different values for k were tried ranging 1 to 10.

(vi) **LibSVM:** LibSVM (Chang et al. [31]) is an integrated algorithm for support vector classification, regression and distribution estimation. It supports multi-class classification and runs faster than SMO. Different types of SVMs were tried like 'C-SVC', 'nu-SVC', 'one-class', 'epsilon-SVR' and 'nu-SVR'. Different types of kernels were tried like 'linear', 'polynomial', 'radial' and 'sigmoid'.

(vii) **MultiLayerPerceptron:** This classifier uses backpropagation (Haykin et al. [90]) to classify instances. This network can be built by hand, created by an algorithm or both. The network can also be monitored and modified during training time. The nodes in this network are all sigmoid (except for when the class is numeric in which case the output nodes become unthresholded linear units). Learning rate of back propagation set to be 0.3, Momentum rate 0.2 etc.

(viii) **NaiveBayesUpdateable:** This Naive Bayes classifier (John et al. [105]) uses estimator classes. This is the updateable version of NaiveBayes. This classifier uses

a default precision of 0.1 for numeric attributes when build Classifier is called with zero training instances.

(ix) **PART:** Class for generating a PART (Frank et al. [60]) decision list. Its named like this because it is based on **Par**tial Decision **T**rees. Uses separate and conquer. Builds a partial C4.5 decision tree in each iteration and makes the ”best” leaf into a rule. Minimum threshold for pruning was set to 0.25 and minimum numbers of objects per leaf were set to 2.

(x) **Random-Forest:** This classifier constructs a forest of random trees. Random Forest (Breiman et al. [26]) grows many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest). Numbers of trees to be built were set to 10.

(xi) **REPTree:** Fast decision tree learner (Witten et al. [188]). This classifier builds a decision/regression tree using information gain/variance and prunes it using reduced-error pruning (with backfitting). Only sorts values for numeric attributes once. Missing values are dealt with by splitting the corresponding instances into pieces.

(xii) **SimpleLogistic:** This classifier builds linear logistic regression models. LogitBoost with simple regression functions as base learners is used for fitting the logistic models (Sumner et al. [178]). The optimal number of LogitBoost iterations to perform is cross-validated, which leads to automatic attribute selection. Maximum number of boosting iterations were allowed to be 500.

(xiii) **SMO:** SMO (Keerthi et al. [109]) classifier Implements sequential minimal optimization algorithm for training a support vector classifier. This implementation globally replaces all missing values and transforms nominal attributes into binary ones. It also normalizes all attributes by default. Multi-class problems are solved using pair wise classification.

(xiv) **ZeroR:** This classifier builds and uses a 0-R (Zero Residual) classifier (Witten et al. [188]). It predicts the mean (for a numeric class) or the mode (for a nominal class).

(xv) **Boosting:** The idea of Boosting is to combine simple rules to form an ensemble such that the performance of the single ensemble member is improved i.e. Boosted. AdaBoostM1 algorithm was used for boosting trees (Yoav et al. [192]). Different parameters were set as follows: Number of iterations allowed was 10 and 100 percentage of weight mass being used.

(xvi) **Bagging** (**B**ootstrap A**gg**regat**ing**): It produces replications of the train set by sampling with replacement. Each replacement of the train set has the same size as the original set, but some examples can appear more than once while other don't appear at all. A classifier is generated from each replication. All classifiers are used to classify each sample from the test set using a vote scheme (Breiman [27]). Size of each bag being set to 100 and number of iterations allowed were 10.

We have applied Bagging and Boosting on ADTree, Decision Stump and Random forest algorithms. Experimental results of both Boosting and Bagging are found to be very enthusiastic.

6.1.3.2 Regression Algorithms

(i) **Isotonic Regression:** This algorithm learns through an isotonic regression model (Best et al. [19]). It picks the attribute that result in the lowest squared error. It can deal with numeric attributes only. It considers the monotonically increasing case as well as the monotonically decreasing case.

(ii) **Linear Regression:** This algorithm uses linear regression for prediction. It uses the Akaike criterion (Kropotov et al. [121]) for model selection and is able to deal with weighted instances.

(iii) **Pace Regression:** This algorithm consists of a group of estimators that are either overall optimal or optimal under certain conditions. Under regularity conditions, pace

regression (Wang et al. [186]) is provably optimal when the number of coefficients tends to infinity.

(iv) **RBFNetwork:** This algorithm implements a normalized Gaussian radial basis function network (Zhang et al. [202]). It uses the k-means clustering algorithm to provide the basis functions and learns either a logistic regression (discrete class problems) or linear regression (numeric class problems) on top of that. Symmetric multivariate Gaussians are fit to the data from each cluster. If the class is nominal it uses the given number of clusters per class. It standardizes all numeric attributes to zero mean and unit variance.

(v) **SVMreg:** SVMreg implements the support vector machine for regression (Shevade et al. [173]). The parameters can be learned using various algorithms.

Process presented in chapter 4 has been applied over twenty classification algorithms and five regression algorithms described above. Using five regression algorithms is an effort to find the most suitable algorithm to suit proposed process and default values for different regression algorithms were used for different parameters. Results generated through all regression algorithms over classification algorithms are compared with previously known processes, which were explored in chapter 3. Next section describes the metrics used for comparing the results of different processes.

6.1.4 Metrics for evaluation

Learning techniques and algorithms are used in a variety of domains. Different performance metrics are considered appropriate for different domains, e. g. Precision/Recall measures are preferred metrics for information retrieval, ROC curves/area is preferred metric for the problems related to medical domain, Lift is preferred for marketing tasks etc. Each metric is dedicated to some specific nature of algorithm evaluation. No individual metric may be used for all domains. So, there is a need to test different learning algorithms based on a large set of metrics. Any metric may belong to more than one broader category depending on its nature. Metrics used for testing algorithms are broadly categorized as follows:

(i) **Confusion Matrix Based Metrics:** Outcome of all classification tasks produces four types of output i.e. two from each instance is mapped to one element of the set { Positive, Negative} from actual positive and negative class labels, whereas other two labels {Positive, Negative} from the class predictions produced by a model (Lilian et al. [131]). Different statistics like Accuracy, Precision, Recall, Fallout, F-measure, Margin etc. are directly derived from the confusion matrix (Provost et al. [161]), whereas Lift, AUC are derived from it.

		Actual Class	
		Positive	**Negative**
Predicted Class	**Positive**	**T**rue **P**ositives	**F**alse **P**ositives
	Negative	**F**alse **N**egatives	**T**rue **N**egatives

Table 6.2: A contingency table for a binary class problem

(ii) **Threshold metrics:** The threshold metrics are accuracy, F-score and Lift (Giudici [68]). A fixed threshold 0.5 is used for Accuracy and F-Score. For lift, percent p of cases is predicted as positive and the rest as negative, for present study p is selected to be 25%. Predictions may have a significant distance from these thresholds.

(iii) **Rank Metrics:** The rank metrics used are **A**rea **U**nder the RO**C** curve (i. e. AUC) (Provost et al. [161]), Average Precision and Recall.

(iv) **Errors:** Different types of errors have been involved in this study. Absolute Error, Relative Error, Root Mean Squared Error, Squared Error and Fallout etc. have been calculated for all the algorithms and problems involved for present study. Classification error has been omitted from the tables included in Chapter 7, because it can be calculated from the accuracy measure by subtracting accuracy from one.

(v) **Probability Metrics:** Probability metrics, Root mean squared error and Mean crossed-entropy interpret the predicted value of each case as a conditional probability of that case being in the positive class.

(vi) **Other Metrics:** Other metrics like kappa and correlation are calculated**.** The kappa coefficient measures pair wise agreement among a set of coders making category judgments, correcting for expected chance agreement (Berry [17]), whereas correlation calculates the degree of relationship between attributes.

6.1.5 Bootstrap analysis for ranking of problems/metrics

The results generated from experiments depend on the choice of problems, selection of datasets and metrics applied for evaluation. There is a need to study the impact of selecting other problems or performance evaluation by other metrics, on the results. Actual result has been calculated for each algorithm's average performance (for 15 statistics) from all 10 datasets. To help evaluate the impact of the choice of problems and metrics, we performed a bootstrap analysis, where we randomly selected a problem and for this problem, we randomly selected an algorithm. For this algorithm we calculate average performance over all 15 statistics. Overall procedure for bootstrap analysis is as follows: We randomly select a bootstrap sample (sampling with replacement) from the original 10 problems (10 folds). For this sample of problems we then randomly select a bootstrap sample of 15 metrics from the original 15 metrics of a randomly selected algorithm (again sampling with replacement). For this bootstrap sample of problems and metrics we rank the twenty algorithms by mean performance across the sampled problems and metrics (and the 10 folds). This bootstrap sampling is repeated 2000 times, yielding 2000 potentially different rankings of the learning methods.

Different datasets are applied over different algorithms through different processes to generate metrics described in this chapter. Results generated through different processes are presented and evaluated in section 7.1 of next chapter. Next section describes the case studies used for evaluating effectiveness of model proposed in Chapter 5.

6.2 Case Studies for 'Genetic Information System Development and Maintenance' Model

'Genetic Information System Development and Maintenance' model was presented in Chapter 5 for knowledge based information system development. This section introduces the case studies of organizations, where proposed model was applied.

(i) Case Study 1

PEPSI FOODS Ltd. is a renowned soft drink manufacturing, multinational company operating in most of the countries worldwide. Soft drink industry is marketing based business and purely dependent upon eating and drinking habits of the people. Pepsi Foods Ltd. has diversified its business in India not only in soft drinks but also in eatables goods. Information system of the company needs to be efficient enough to meet the dynamic information requirements of the industry. As soft drink industry is marketing operations intensive, its information requirements are highly dynamic. Changing information needs has created a lot of difficulty for the software developers, as maintenance costs rise beyond the estimates and has caused unreliability even after spending sufficient amounts on the system development and maintenance. Earlier software development methodologies and models used, could not meet the requirements.

(ii) Case Study 2

Ranbaxy Laboratories Ltd. is a leading pharmaceutical multinational company of India. Rising competition from local and global players has caused the need of an efficient information system to meet the challenge of maintaining leadership and expansion of business globally. Dynamic information needs has forced the company to think over its information technology and information system strategy. Earlier information system development has failed to meet the challenge. Company is looking for any permanent solution for their information needs.

(iii) Case Study 3

Usha Power Tec. is a company committed to meet the need for power conditioning in India. Company has record tremendous growth in future and has bright future prospects as well, but rising competition and dynamic information requirements of the information systems has forced the company to think on strengthening its information system. Earlier development is unable to meet the requirement and company is lifelong solutions for its information systems.

(iv) Case Study 4

Megaleap Inc. is an online retailing organization. Company deals in a variety of e-Stores to provide a storefront to a variety of manufacturers and retailers. Company operates in a highly dynamic environment. Company has its information system operating, but dynamism of customer choices, retailers offers etc. has emerged the need for incorporating the information dynamism to be incorporated in its information system. So company is looking for long-term solution of its information system development and maintenance at optimized costs.

Proposed model is implemented in many organizations operating in a variety of business domains. 'Genetic Information System Development and Maintenance' is helpful in finding the solutions of many problems related to software development and maintenance as discussed in section 5.4 of chapter 5. However, investigations for testing the model were guided towards reliability, maintainability and costs involved. Proposed model definitely show the improvement in adaptive and perfective maintenance, so investigations need to be concentrated upon corrective maintenance/change management at low or no extra cost. Following section will hypothesize investigations in these directions.

6.3 Approach for the evaluation of 'Genetic Information System Development and Maintenance' Model

Teams of researchers and software developers were involved in the process of software development for the organizations discussed above. 'Genetic Information System Development and Maintenance' model was tried and the results were tested against data

collected from earlier developments for the same organizations and actual data of the development process and results of Case Study 1 are presented here, similar results are observed for other organizations discussed above.

Research methodology for proposed research work includes implementation of proposed model in diversified business organizations, Collection of data from actual implementation, Collection of dataset from standard organizations, Comparison of results, Hypothesis testing and concluding with results etc.

For testing parameters like reliable maintenance of information system and total system costs are of major concern. The proposed model is highly adaptive due to adoption of Internet based technologies and team involved on continuous basis for system development work for the perfection of the system. We also perform hypothesis testing against the parameters related to reliable corrective maintenance and costs involved. Datasets are collected from the organizations' offices scattered worldwide, whereas criteria for collection of data was based on System development/Maintenance Costs, Size of System, Environment of the system, Scale of operations for system etc. These factors were also scaled according to the characteristics of the system under study.

6.4 Conclusion

In this chapter, experimental setup for evaluating the performance of different previously known supervised learning, proposed process for supervised learning and proposed model for knowledge based information system development, are discussed. Section 6.1 described datasets, processes, algorithms and metrics used for experimentation for evaluating performance of supervised learning processes. Section 6.2 discussed case studies about the organizations where 'Genetic Information System Development and Maintenance' model was implemented. Hypothesis testing methodology adopted for evaluating proposed model over different metrics has also been discussed.

Chapter 7

Results, Interpretations and Evaluation

This chapter presents the results generated from massive experimentation of this research work with their evaluation. Results are again categorized according to models presented in this research work. Section 7.1 includes the results of previously known processes as well as proposed process by present research i.e. Fuzzy Boundaries of Regression Based Cluster process for supervised learning. Section 7.2 includes hypothesized results of 'Genetic Information System Development and Maintenance' model.

7.1 Experimental Results of Supervised Learning Processes

This section includes the experimental results of supervised learning processes. Experimental results are divided into two categories viz. Major study and Minor study. Major study of supervised learning processes include eight processes for experimentation, out of which three processes are previously known (i. e. Fixed Split, Cross validation and Platt Scaling) whereas five processes are created from proposed process over five regression based algorithms (i.e. Isotonic Regression, Linear Regression, Pace Regression, RBF Network and SVM Regression). Minor study consists of eight processes from major study and three other previously known processes that is two from post processing through

Additive Regression and LogitBoost, one from Stacking of Isotonic Regression. Major study includes twenty classification algorithms for experimentation whereas minor study includes five algorithms. Both studies use same datasets for experimentation.

7.1.1 Major Study

Major study includes twenty algorithms and experiments are performed over Fixed Split, Cross Validation, Platt Scaling and proposed Fuzzy clusters based model over five regression algorithm. For Fixed Split validation original dataset is divided into a train set and a test set, then experiments are performed. For Cross Validation dataset is again divided into a Train set and a Test set, This Train set is further divided into ten fold datasets, Experiments are performed over one fold with the help of others and dataset with minimum squared error is selected for testing the performance over the test set. For Platt Scaling, Cross validated model is passed through a sigmoid and probabilities based predictions are performed. For Fuzzy clustered based proposed model dataset is again divided into a Train set and a Test set, The Train set is further divided into ten fold subsets of the train data. Experiments are performed using first fold of the train set treating as a train set for over second fold of the train set treating it as a test set. Experiments were performed using classification and regression algorithms in parallel. Outputs generated from these algorithms were combined through frequency charts and a part of second fold dataset is removed from it after applying fuzzy boundaries based clusters. After extracting a part of the second fold train set, remaining data is added into first fold of the train set. Now this updated train set is used as training data over third fold of the train set treating third fold as the test set. Like this procedure continues until all ten fold of the train sets are added into the cumulative train set after extracting data using proposed process of fuzzy boundaries of regression based clusters.

7.1.1.1 Performance by Problem

Table 7.1(a, b and c) includes accuracy of twenty algorithms involved for study and is ranked in descending order based on their average performance over eight processes viz. Fixed Split, Cross Validation, Platt Scaling and Five regression algorithms based on fuzzy boundaries of regression based clusters.

7.1.1.2 Percentage of Datasets Used for Proposed Model

Table 7.2(a and b) includes the percentages of datasets extracted from original datasets after applying Theory of Fuzzy Boundaries of Regression based Clusters. These percentages of the train set were extracted from original training datasets to be used for calculating results from different algorithms.

7.1.1.3 Performance by metrics

Table 7.3(a, b, c, d and e) includes averages of fifteen metrics involved in study and these algorithms are positioned in descending order according to average accuracy. For few metrics output was generated to be NaN (**N**ot **a** **N**umber), for such metrics averages are calculated over remaining values, excluding the count of such values. These values are pointed with an asterisk (*) and if all the values (for all ten problems) are NaN, such values are represented by NaN*.

Algorithm	Validation / Calibration/Fuzzy	Adult	CovType	DS1_100	Letter	Oracle	Pen Digits	Physics	Satellite	Shuttle	Tic2000	Average
Random-Forest-Bagging	Fixed Split	0.854	0.980	0.980	0.963	0.843	0.994	0.843	0.948	1.000	0.929	0.9334
Random-Forest-Boosting	Fixed Split	0.843	0.979	0.980	0.967	0.843	0.990	0.843	0.952	1.000	0.924	0.9322
Random-Forest	Fixed Split	0.846	0.977	0.979	0.952	1.000	0.990	0.678	0.941	1.000	0.925	0.9288
RandomForest-Bagging	SVMRegFuzzy	0.854	0.982	0.978	0.963	1.000	0.981	0.708	0.822	1.000	0.927	0.9215
Random-Forest-Bagging	Cross Val	0.854	0.982	0.977	0.959	1.000	0.982	0.708	0.822	0.999	0.931	0.9215
RandomForest-Bagging	PaceFuzzy	0.852	0.983	0.980	0.963	1.000	0.982	0.708	0.817	1.000	0.930	0.9215
RandomForest-Bagging	RBFNetFuzzy	0.852	0.983	0.979	0.963	1.000	0.982	0.708	0.819	1.000	0.929	0.9215
RandomForest-Bagging	IsoFuzzy	0.852	0.982	0.979	0.963	1.000	0.983	0.708	0.819	0.999	0.929	0.9214
Random-Forest-Bagging	Platt	0.853	0.982	0.978	0.960	1.000	0.983	0.708	0.821	0.999	0.931	0.9214
RandomForest-Bagging	LinearFuzzy	0.852	0.982	0.978	0.960	1.000	0.982	0.707	0.817	1.000	0.932	0.9211
PART	Fixed Split	0.857	0.976	0.979	0.919	1.000	0.981	0.658	0.930	1.000	0.908	0.9208
REPTree	Fixed Split	0.854	0.974	0.977	0.890	1.000	0.963	0.686	0.918	0.998	0.940	0.9202
RandomForest-Boosting	PaceFuzzy	0.846	0.982	0.979	0.967	1.000	0.977	0.695	0.831	0.999	0.923	0.9198
RandomForest-Boosting	LinearFuzzy	0.845	0.982	0.977	0.966	1.000	0.979	0.690	0.830	1.000	0.925	0.9193
RandomForest-Boosting	RBFNetFuzzy	0.843	0.983	0.977	0.965	1.000	0.983	0.688	0.830	0.999	0.924	0.9193
MultiLayerPerceptron	Fixed Split	0.838	0.975	0.979	0.891	0.989	0.989	0.673	0.936	0.999	0.923	0.9191
Random-Forest-Boosting	Cross Val	0.843	0.984	0.975	0.961	1.000	0.979	0.688	0.828	0.999	0.927	0.9185
RandomForest-Boosting	SVMRegFuzzy	0.843	0.981	0.975	0.965	1.000	0.982	0.687	0.824	1.000	0.922	0.9177
RandomForest-Boosting	IsoFuzzy	0.844	0.981	0.975	0.963	1.000	0.975	0.689	0.830	1.000	0.916	0.9174
RandomForest	IsoFuzzy	0.845	0.983	0.977	0.949	1.000	0.978	0.681	0.829	0.999	0.926	0.9167
RandomForest	SVMRegFuzzy	0.843	0.980	0.981	0.948	1.000	0.980	0.684	0.822	0.999	0.926	0.9163
Random-Forest-Boosting	Platt	0.843	0.984	0.979	0.961	1.000	0.981	0.691	0.828	0.999	0.895	0.9162
RandomForest	PaceFuzzy	0.846	0.981	0.979	0.952	1.000	0.979	0.678	0.818	0.999	0.926	0.9159
RandomForest	LinearFuzzy	0.847	0.978	0.978	0.949	1.000	0.981	0.682	0.819	0.999	0.926	0.9158
RandomForest	RBFNetFuzzy	0.846	0.978	0.976	0.947	1.000	0.978	0.678	0.820	1.000	0.928	0.9150
Random-Forest	Cross Val	0.846	0.980	0.977	0.945	1.000	0.978	0.678	0.817	1.000	0.927	0.9149
ADTree-Boosting	Fixed Split	0.836	0.969	0.978	0.842	0.836	0.979	0.836	0.928	1.000	0.940	0.9143
Random-Forest	Platt	0.840	0.981	0.980	0.947	1.000	0.982	0.677	0.807	0.999	0.920	0.9132
Decision-Table	Fixed Split	0.857	0.978	0.977	0.849	1.000	0.920	0.708	0.902	0.999	0.940	0.9129
MultiLayerPerceptron	Platt	0.837	0.976	0.980	0.885	0.989	0.982	0.673	0.849	0.998	0.937	0.9106
MultiLayerPerceptron	Cross Val	0.838	0.976	0.974	0.885	0.989	0.982	0.673	0.849	0.998	0.932	0.9096
PART	Cross Val	0.857	0.974	0.977	0.911	1.000	0.968	0.658	0.838	0.999	0.912	0.9094
PART	Platt	0.857	0.974	0.977	0.911	1.000	0.968	0.658	0.838	0.999	0.912	0.9094
IB-k	Fixed Split	0.786	0.979	0.973	0.972	0.913	0.997	0.630	0.943	0.999	0.901	0.9091
ADTree-BoostingPace	PaceFuzzy	0.849	0.985	0.976	0.832	1.000	0.966	0.706	0.837	1.000	0.939	0.9089
MLP	LinearFuzzy	0.840	0.979	0.973	0.888	0.988	0.982	0.668	0.853	0.995	0.922	0.9087
REPTree	PaceFuzzy	0.850	0.980	0.974	0.899	1.000	0.954	0.682	0.809	0.999	0.939	0.9086
MLP	IsoFuzzy	0.840	0.974	0.971	0.897	0.980	0.980	0.670	0.847	0.998	0.920	0.9078
REPTree	IsoFuzzy	0.850	0.985	0.975	0.891	1.000	0.961	0.684	0.798	0.999	0.935	0.9078
REPTree	Cross Val	0.854	0.977	0.972	0.887	1.000	0.956	0.686	0.808	0.999	0.939	0.9078
REPTree	Platt	0.854	0.977	0.972	0.886	1.000	0.956	0.686	0.808	0.999	0.939	0.9077
MLP	PaceFuzzy	0.840	0.965	0.969	0.881	0.991	0.980	0.676	0.848	0.998	0.925	0.9073
REPTree	RBFNetFuzzy	0.852	0.985	0.955	0.892	1.000	0.956	0.686	0.808	0.999	0.939	0.9070
REPTree	SVMRegFuzzy	0.852	0.978	0.957	0.890	1.000	0.951	0.691	0.814	0.998	0.939	0.9069
ADTree-BoostingLinear	LinearFuzzy	0.846	0.983	0.975	0.837	1.000	0.970	0.706	0.810	1.000	0.939	0.9064
ADTree-BoostingRBFNet	RBFNetFuzzy	0.854	0.982	0.973	0.837	1.000	0.966	0.707	0.807	0.999	0.938	0.9063
ADTree-BoostingIso	IsoFuzzy	0.853	0.983	0.971	0.828	1.000	0.969	0.703	0.816	1.000	0.939	0.9062
ADTree-BoostingSVMReg	SVMRegFuzzy	0.852	0.985	0.970	0.826	1.000	0.968	0.707	0.816	0.999	0.939	0.9062
ADTree-Boosting	Cross Val	0.836	0.980	0.975	0.831	1.000	0.968	0.707	0.824	0.999	0.938	0.9059
PART	SVMRegFuzzy	0.848	0.969	0.970	0.916	1.000	0.970	0.659	0.832	0.999	0.897	0.9059
REPTree	LinearFuzzy	0.849	0.982	0.954	0.893	1.000	0.956	0.681	0.806	0.999	0.939	0.9058
PART	LinearFuzzy	0.856	0.973	0.969	0.922	1.000	0.965	0.661	0.801	0.999	0.911	0.9056
PART	PaceFuzzy	0.859	0.972	0.976	0.920	1.000	0.969	0.657	0.800	0.999	0.904	0.9056

Table 7. 1a: Accuracy of all algorithms over ten problems and their mean performance in descending order

Algorithm	Validation / Calibration/Fuzzy	Adult	CovType	DS1_100	Letter	Oracle	Pen Digits	Physics	Satellite	Shuttle	Tic2000	Average
PART	IsoFuzzy	0.857	0.978	0.969	0.917	1.000	0.961	0.659	0.811	0.999	0.899	0.9050
MLP	SVMRegFuzzy	0.830	0.969	0.969	0.892	0.990	0.980	0.669	0.834	0.998	0.916	0.9049
ADTree-Boosting	Platt	0.839	0.980	0.949	0.832	1.000	0.966	0.707	0.831	0.999	0.938	0.9042
PART	RBFNetFuzzy	0.846	0.977	0.968	0.913	1.000	0.970	0.658	0.796	0.999	0.912	0.9040
ADTree-Bagging	Fixed Split	0.852	0.968	0.979	0.757	0.836	0.931	0.836	0.932	1.000	0.940	0.9031
MLP	RBFNetFuzzy	0.834	0.981	0.969	0.891	0.989	0.977	0.673	0.839	0.998	0.870	0.9022
Decision Table	PaceFuzzy	0.854	0.974	0.978	0.831	1.000	0.906	0.708	0.827	0.999	0.939	0.9015
Decision Table	IsoFuzzy	0.857	0.985	0.979	0.839	1.000	0.906	0.700	0.811	0.998	0.939	0.9015
Decision Table	LinearFuzzy	0.857	0.942	0.975	0.828	1.000	0.905	0.708	0.827	0.999	0.939	0.8980
Decision Table	RBFNetFuzzy	0.857	0.972	0.964	0.838	1.000	0.895	0.708	0.800	0.998	0.939	0.8970
Decision-Table	Cross Val	0.857	0.938	0.980	0.832	1.000	0.909	0.708	0.804	0.998	0.939	0.8963
Decision-Table	Platt	0.857	0.938	0.980	0.828	1.000	0.906	0.708	0.807	0.998	0.939	0.8961
IB-k	Cross Val	0.786	0.976	0.966	0.971	0.913	0.989	0.630	0.826	0.999	0.906	0.8960
ADTree	Fixed Split	0.851	0.968	0.979	0.740	1.000	0.890	0.695	0.896	1.000	0.940	0.8959
Decision Table	SVMRegFuzzy	0.857	0.958	0.975	0.823	1.000	0.872	0.708	0.827	0.998	0.938	0.8955
Ibk	PaceFuzzy	0.785	0.976	0.965	0.972	0.913	0.990	0.629	0.830	0.999	0.897	0.8955
Ibk	LinearFuzzy	0.785	0.976	0.964	0.972	0.912	0.990	0.629	0.829	0.999	0.895	0.8952
IB-k	Platt	0.786	0.976	0.965	0.970	0.913	0.989	0.630	0.824	0.999	0.895	0.8947
ADTree-Bagging	Platt	0.852	0.985	0.975	0.761	1.000	0.898	0.699	0.836	0.999	0.939	0.8943
ADTree-Bagging	Cross Val	0.852	0.986	0.974	0.768	1.000	0.898	0.699	0.822	1.000	0.939	0.8937
ADTree-Bagging	SVMRegFuzzy	0.852	0.985	0.971	0.766	1.000	0.906	0.699	0.817	0.999	0.939	0.8935
Ibk	RBFNetFuzzy	0.783	0.974	0.953	0.972	0.913	0.990	0.630	0.825	0.999	0.888	0.8927
ADTree-Bagging	IsoFuzzy	0.852	0.985	0.972	0.771	1.000	0.890	0.702	0.813	0.999	0.939	0.8922
ADTree-Bagging	RBFNetFuzzy	0.852	0.985	0.974	0.761	1.000	0.898	0.699	0.812	0.999	0.939	0.8920
ADTree-Bagging	PaceFuzzy	0.852	0.985	0.969	0.755	1.000	0.904	0.700	0.813	0.999	0.939	0.8917
ADTree-Bagging	LinearFuzzy	0.853	0.985	0.970	0.754	1.000	0.889	0.701	0.817	0.999	0.939	0.8907
SimpleLogistic	Fixed Split	0.842	0.973	0.981	0.732	0.997	0.842	0.705	0.926	0.959	0.940	0.8897
SMO	Fixed Split	0.838	0.973	0.980	0.736	0.985	0.846	0.705	0.927	0.956	0.940	0.8887
Ibk	IsoFuzzy	0.775	0.976	0.961	0.972	0.875	0.990	0.625	0.824	0.999	0.881	0.8878
ADTree	Cross Val	0.851	0.985	0.970	0.757	1.000	0.877	0.695	0.803	1.000	0.939	0.8877
Ibk	SVMRegFuzzy	0.771	0.972	0.948	0.971	0.911	0.990	0.620	0.829	0.999	0.853	0.8863
ADTree	Platt	0.851	0.985	0.971	0.742	1.000	0.873	0.691	0.808	0.998	0.939	0.8859
ADTree	RBFNetFuzzy	0.852	0.985	0.969	0.715	1.000	0.877	0.695	0.821	1.000	0.939	0.8853
Decision-Stump-Boosting	Fixed Split	0.842	0.956	0.971	0.699	0.842	0.852	0.842	0.900	0.998	0.940	0.8844
ADTree	LinearFuzzy	0.851	0.985	0.971	0.715	1.000	0.878	0.695	0.808	1.000	0.939	0.8843
ADTree	IsoFuzzy	0.851	0.985	0.967	0.715	1.000	0.878	0.695	0.815	0.998	0.939	0.8842
ADTree	PaceFuzzy	0.850	0.985	0.971	0.715	1.000	0.878	0.695	0.808	1.000	0.939	0.8842
ADTree	SVMRegFuzzy	0.851	0.985	0.964	0.715	1.000	0.878	0.695	0.808	1.000	0.939	0.8836
SMO	IsoFuzzy	0.837	0.976	0.983	0.730	0.988	0.830	0.706	0.858	0.965	0.939	0.8811
SMO	RBFNetFuzzy	0.839	0.972	0.983	0.730	0.985	0.831	0.705	0.850	0.969	0.939	0.8803
SimpleLogistic	Cross Val	0.842	0.982	0.982	0.724	0.997	0.829	0.705	0.837	0.959	0.939	0.8796
SMO	PaceFuzzy	0.838	0.976	0.983	0.730	0.985	0.831	0.705	0.856	0.948	0.939	0.8790
BayesNetGen	IsoFuzzy	0.850	0.981	0.979	0.760	1.000	0.829	0.682	0.802	0.996	0.911	0.8790
SMO	LinearFuzzy	0.838	0.976	0.983	0.730	0.985	0.831	0.705	0.856	0.947	0.939	0.8790
BayesNetGenerator	Platt	0.851	0.979	0.980	0.771	0.988	0.821	0.679	0.791	0.994	0.937	0.8789
Simple Logistic	IsoFuzzy	0.841	0.975	0.983	0.723	1.000	0.829	0.709	0.831	0.959	0.939	0.8788
SMO	SVMRegFuzzy	0.834	0.976	0.982	0.730	0.984	0.831	0.706	0.856	0.951	0.939	0.8788
Simple Logistic	SVMRegFuzzy	0.842	0.975	0.983	0.724	1.000	0.827	0.709	0.828	0.959	0.939	0.8786
Simple Logistic	RBFNetFuzzy	0.841	0.974	0.982	0.724	1.000	0.828	0.707	0.831	0.959	0.939	0.8784
Simple Logistic	PaceFuzzy	0.842	0.975	0.982	0.725	1.000	0.827	0.708	0.825	0.959	0.939	0.8782
Simple Logistic	LinearFuzzy	0.842	0.975	0.983	0.722	1.000	0.828	0.708	0.825	0.959	0.939	0.8781
BayesNetGen	RBFNetFuzzy	0.833	0.982	0.981	0.760	0.991	0.826	0.680	0.790	0.995	0.916	0.8754
SMO	Platt	0.838	0.933	0.983	0.730	0.985	0.831	0.705	0.856	0.947	0.939	0.8747
BayesNetGen	SVMRegFuzzy	0.837	0.980	0.980	0.760	0.990	0.832	0.682	0.789	0.994	0.901	0.8746

Table 7.1b: Accuracy of all algorithms over ten problems and their mean performances in descending order

Algorithm	Validation / Calibration/Fuzzy	Adult	CovType	DS1_100	Letter	Oracle	Pen Digits	Physics	Satellite	Shuttle	Tic2000	Average
BayesNetGen	PaceFuzzy	0.832	0.981	0.980	0.765	0.989	0.828	0.681	0.790	0.994	0.897	0.8736
Decision-Stump-Boosting	Cross Val	0.842	0.978	0.958	0.696	1.000	0.841	0.670	0.812	0.997	0.939	0.8734
Decision-Stump-Boosting	Platt	0.842	0.978	0.958	0.696	1.000	0.836	0.670	0.815	0.998	0.939	0.8732
BayesNetGen	LinearFuzzy	0.832	0.982	0.981	0.760	0.990	0.829	0.680	0.791	0.994	0.887	0.8727
BayesNetGenerator	Fixed Split	0.830	0.939	0.977	0.770	0.989	0.830	0.680	0.872	0.992	0.835	0.8713
BayesNetGenerator	Cross Val	0.830	0.981	0.979	0.761	0.989	0.823	0.680	0.791	0.993	0.867	0.8694
SimpleLogistic	Platt	0.806	0.979	0.978	0.724	1.000	0.826	0.707	0.815	0.918	0.938	0.8692
Decision Stump-Boosting	SVMRegFuzzy	0.841	0.956	0.958	0.696	1.000	0.807	0.677	0.800	0.997	0.939	0.8672
Decision Stump-Boosting	IsoFuzzy	0.842	0.956	0.958	0.696	1.000	0.807	0.673	0.801	0.998	0.939	0.8671
Decision Stump-Boosting	RBFNetFuzzy	0.842	0.956	0.958	0.696	1.000	0.808	0.670	0.800	0.997	0.939	0.8666
Decision Stump-Boosting	LinearFuzzy	0.842	0.956	0.958	0.696	1.000	0.808	0.670	0.800	0.997	0.939	0.8666
Decision Stump-Boosting	PaceFuzzy	0.842	0.956	0.958	0.696	1.000	0.808	0.670	0.798	0.997	0.939	0.8664
SMO	Cross Val	0.838	0.933	0.983	0.730	0.985	0.831	0.705	0.856	0.856	0.939	0.8657
Decision-Stump-Bagging	Fixed Split	0.760	0.922	0.970	0.671	0.842	0.718	0.842	0.879	0.927	0.940	0.8471
Decision-Stump	Fixed Split	0.760	0.922	0.970	0.671	1.000	0.710	0.670	0.879	0.927	0.940	0.8449
NBU	IsoFuzzy	0.810	0.982	0.958	0.706	0.936	0.779	0.670	0.816	0.899	0.874	0.8431
Decision Stump-Bagging	LinearFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.827	0.928	0.939	0.8408
Decision Stump-Bagging	IsoFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.668	0.826	0.928	0.939	0.8405
Decision-Stump	Cross Val	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.823	0.928	0.939	0.8404
Decision-Stump	Platt	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.823	0.928	0.939	0.8404
Decision-Stump-Bagging	Platt	0.760	0.952	0.957	0.668	1.000	0.706	0.668	0.823	0.928	0.939	0.8401
Decision Stump-Bagging	RBFNetFuzzy	0.760	0.952	0.958	0.668	1.000	0.704	0.668	0.814	0.928	0.939	0.8390
Decision-Stump-Bagging	Cross Val	0.760	0.952	0.958	0.668	1.000	0.706	0.668	0.810	0.928	0.939	0.8389
Decision Stump-Bagging	PaceFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.671	0.796	0.928	0.939	0.8378
Decision-Stump	LinearFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.796	0.928	0.939	0.8377
Decision-Stump	PaceFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.796	0.928	0.939	0.8377
Decision-Stump	RBFNetFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.796	0.928	0.939	0.8377
Decision Stump-Bagging	SVMRegFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.668	0.797	0.928	0.939	0.8375
Decision-Stump	IsoFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.667	0.796	0.928	0.939	0.8374
Decision-Stump	SVMRegFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.667	0.796	0.928	0.939	0.8374
NBU	RBFNetFuzzy	0.808	0.982	0.980	0.705	0.807	0.755	0.664	0.818	0.902	0.890	0.8310
NBU	SVMRegFuzzy	0.819	0.983	0.971	0.705	0.759	0.764	0.667	0.813	0.899	0.873	0.8254
NBU	PaceFuzzy	0.810	0.982	0.959	0.705	0.750	0.765	0.665	0.814	0.898	0.827	0.8174
NBU	LinearFuzzy	0.810	0.983	0.963	0.705	0.750	0.763	0.666	0.813	0.898	0.823	0.8173
Naïve-Bayes-Updatable	Fixed Split	0.809	0.938	0.943	0.716	0.743	0.776	0.664	0.877	0.896	0.755	0.8117
Naïve-Bayes-Updatable	Platt	0.809	0.979	0.971	0.704	0.743	0.755	0.664	0.816	0.897	0.772	0.8110
Naïve-Bayes-Updatable	Cross Val	0.809	0.981	0.954	0.704	0.743	0.755	0.664	0.816	0.897	0.755	0.8078
LibSVM	Fixed Split	0.760	0.967	0.981	0.971	0.562	0.505	0.627	0.553	0.948	0.940	0.7814
LibSVM	IsoFuzzy	0.760	0.983	0.983	0.974	0.562	0.515	0.624	0.358	0.969	0.939	0.7667
LibSVM	RBFNetFuzzy	0.760	0.982	0.982	0.974	0.562	0.511	0.627	0.358	0.951	0.939	0.7647
LibSVM	PaceFuzzy	0.760	0.981	0.983	0.974	0.562	0.513	0.626	0.358	0.950	0.939	0.7647
LibSVM	Platt	0.760	0.981	0.983	0.972	0.562	0.511	0.627	0.358	0.943	0.939	0.7636
LibSVM	LinearFuzzy	0.760	0.982	0.983	0.974	0.562	0.497	0.626	0.358	0.951	0.939	0.7632
LibSVM	SVMRegFuzzy	0.760	0.975	0.982	0.975	0.562	0.496	0.624	0.358	0.951	0.932	0.7615
LibSVM	Cross Val	0.760	0.981	0.983	0.972	0.562	0.511	0.627	0.358	0.358	0.939	0.7052
ZeroR	Fixed Split	0.760	0.829	0.970	0.501	0.562	0.505	0.502	0.553	0.789	0.940	0.6909
ZeroR	Cross Val	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	Platt	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	IsoFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	LinearFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	PaceFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	RBFNetFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	SVMRegFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965

Table 7.1c: Accuracy of all algorithms over ten problems and their mean performance in descending order

Algorithm	Fuzzy Reg. Algo	Datasets										
		Adult	CovType	DS1_100	Letter	Oracle	PenDigits	Physics	Satellite	Shuttle	Tic2000	Avg.
ADTree	Iso	93.17	81.46	92.58	99.64	10.03	97.42	99.48	62.07	64.87	94.26	79.50
ADTree Bagging	Iso	93.17	81.46	92.58	99.64	10.03	97.42	99.48	62.07	64.87	94.26	79.50
ADTree Boosting	Iso	93.17	81.46	92.58	99.64	10.03	97.42	99.48	62.07	64.87	94.26	79.50
BayesNetGenerator	Iso	76.92	69.83	74.67	99.64	13.73	97.42	96.03	47.30	41.68	85.22	70.24
Decision-Stump	Iso	75.77	73.09	69.70	99.82	10.03	96.56	98.00	65.77	46.82	66.14	70.17
Decision-Stump Bagging	Iso	75.77	73.09	69.70	99.82	10.03	96.56	98.00	65.77	46.82	66.14	70.17
Decision-Stump Boosting	Iso	75.77	73.09	69.70	99.82	10.03	96.56	98.00	65.77	46.82	66.14	70.17
Decision-Table	Iso	89.00	66.18	84.73	99.67	13.85	92.44	97.85	59.40	52.27	80.24	73.56
IB-k	Iso	70.02	63.12	75.80	98.38	23.51	93.50	93.31	49.27	39.53	51.14	65.76
LibSVM	Iso	74.39	62.72	68.86	98.63	58.30	95.64	93.79	71.20	45.97	56.62	72.61
MultiLayerPerceptron	Iso	82.33	66.32	67.32	98.53	16.68	93.84	94.42	48.90	53.87	53.44	67.56
Naïve-Bayes-Updatable	Iso	71.12	70.17	71.28	99.20	63.24	96.16	95.07	46.50	53.98	70.82	73.75
PART	Iso	85.85	66.80	89.16	98.78	10.08	91.52	95.42	62.73	48.38	69.46	71.82
Random-Forest	Iso	72.65	60.52	76.15	98.90	28.36	94.24	97.94	48.97	39.79	62.52	68.00
Random-Forest Bagging	Iso	72.65	60.52	76.15	98.90	28.36	94.24	97.94	48.97	39.79	62.52	68.00
Random-Forest Boosting	Iso	72.65	60.52	76.15	98.90	28.36	94.24	97.94	48.97	39.79	62.52	68.00
REPTree	Iso	80.51	70.52	82.47	99.68	10.03	93.54	97.93	66.87	64.61	66.84	73.30
Simple Logistic	Iso	90.11	79.31	85.73	99.59	19.50	97.04	98.38	49.00	73.26	91.68	78.36
SMO	Iso	74.75	62.31	66.17	98.36	11.84	94.08	91.89	42.50	39.70	56.24	63.78
ZeroR	Iso	96.07	100.00	66.55	100.0	100.00	100.00	100.00	100.00	93.29	56.64	91.26
ADTree	Linear	98.32	92.99	95.81	99.05	95.59	98.88	99.96	92.60	85.10	96.92	95.52
ADTree Bagging	Linear	98.32	92.99	95.81	99.05	95.59	98.88	99.96	92.60	85.10	96.92	95.52
ADTree Boosting	Linear	98.32	92.99	95.81	99.05	95.59	98.88	99.96	92.60	85.10	96.92	95.52
BayesNetGenerator	Linear	95.35	90.52	86.89	98.30	95.68	97.82	98.72	90.60	83.68	91.76	92.93
Decision-Stump	Linear	94.77	92.39	87.46	99.72	95.59	99.54	99.68	96.03	83.33	84.06	93.26
Decision-Stump Bagging	Linear	94.77	92.39	87.46	99.72	95.59	99.54	99.68	96.03	83.33	84.06	93.26
Decision-Stump Boosting	Linear	94.77	92.39	87.46	99.72	95.59	99.54	99.68	96.03	83.33	84.06	93.26
Decision-Table	Linear	97.47	92.34	89.92	98.10	95.59	86.02	99.40	92.73	87.20	89.86	92.86
IB-k	Linear	94.63	89.19	83.49	97.39	95.69	93.86	98.10	90.60	83.32	79.16	90.54
LibSVM	Linear	94.66	89.54	84.28	97.59	97.48	97.02	98.25	96.80	84.60	79.34	91.96
MultiLayerPerceptron	Linear	96.32	89.71	83.28	97.53	95.60	94.14	98.12	91.00	83.51	79.16	90.84
Naïve-Bayes-Updatable	Linear	94.63	91.14	77.96	98.39	97.21	96.52	98.92	90.60	84.83	85.40	91.56
PART	Linear	96.24	89.76	95.02	98.07	95.59	96.42	98.97	91.83	84.41	89.48	93.58
Random-Forest	Linear	94.96	90.79	87.85	97.90	95.79	94.64	99.39	90.87	83.33	83.28	91.88
Random-Forest Bagging	Linear	94.96	90.79	87.85	97.90	95.79	94.64	99.39	90.87	83.33	83.28	91.88
Random-Forest Boosting	Linear	94.96	90.79	87.85	97.90	95.79	94.64	99.39	90.87	83.33	83.28	91.88
REPTree	Linear	95.61	90.92	87.83	98.54	95.59	96.16	99.56	95.60	86.93	86.40	93.31
Simple Logistic	Linear	99.10	93.93	92.37	99.40	95.58	97.58	99.66	91.03	90.21	99.86	95.87
SMO	Linear	94.15	88.74	83.57	97.35	95.59	93.96	97.75	90.60	83.32	78.60	90.36
ZeroR	Linear	98.58	100.00	83.57	100.0	100.00	98.78	100.00	100.00	96.75	79.98	95.77
ADTree	Pace	98.35	94.50	95.27	99.14	95.60	98.80	99.94	92.60	87.97	96.46	95.86
ADTree Bagging	Pace	98.35	94.50	95.27	99.14	95.60	98.80	99.94	92.60	87.97	96.46	95.86
ADTree Boosting	Pace	98.35	94.50	95.27	99.14	95.60	98.80	99.94	92.60	87.97	96.46	95.86
BayesNetGenerator	Pace	94.96	88.26	85.95	98.40	95.74	97.86	98.70	91.20	85.68	88.62	92.54
Decision-Stump	Pace	94.70	90.06	85.66	99.72	95.60	99.52	99.68	96.50	85.56	78.88	92.59
Decision-Stump Bagging	Pace	94.70	90.06	85.66	99.72	95.60	99.52	99.68	96.50	85.56	78.88	92.59
Decision-Stump Boosting	Pace	94.70	90.06	85.66	99.72	95.60	99.52	99.68	96.50	85.56	78.88	92.59
Decision-Table	Pace	97.13	91.80	88.16	97.96	95.60	95.82	99.25	92.97	79.06	92.14	92.99
IB-k	Pace	94.11	87.19	82.57	97.33	95.68	94.04	98.19	91.07	85.54	77.40	90.31
LibSVM	Pace	94.60	90.78	84.38	97.47	97.56	97.04	98.34	96.80	86.00	77.36	92.03
MultiLayerPerceptron	Pace	95.97	87.43	82.97	97.53	95.60	94.66	98.22	91.33	85.60	77.44	90.68
Naïve-Bayes-Updatable	Pace	93.84	88.96	85.13	98.51	97.15	97.38	98.94	91.07	86.62	83.10	92.07

Table 7.2a: Percentage of The Train set used for generating results from Fuzzy boundaries of regression based clusters process

Algorithm	Fuzzy Reg Algo	Datasets										
		Adult	CovType	DS1_100	Lette	Oracle	PenDigits	Physics	Satellite	Shuttle	Tic2000	Avg.
Random-Forest	Pace	94.40	89.38	86.98	97.81	95.86	95.02	99.47	91.33	85.56	81.76	91.76
Random-Forest Bagging	Pace	94.40	89.38	86.98	97.81	95.86	95.02	99.47	91.33	85.56	81.76	91.76
Random-Forest Boosting	Pace	94.40	89.38	86.98	97.81	95.86	95.02	99.47	91.33	85.56	81.76	91.76
REPTree	Pace	95.06	88.22	88.62	98.61	95.70	96.98	99.85	94.73	87.21	78.52	92.35
SimpleLogistic	Pace	99.10	94.97	92.67	99.52	95.60	97.66	99.65	91.77	91.00	97.44	95.94
SMO	Pace	93.66	87.03	82.49	97.31	95.60	94.04	97.85	91.07	85.53	76.72	90.13
ZeroR	Pace	98.61	100.00	82.49	100.0	100.00	100.00	100.00	100.00	97.09	77.36	95.56
ADTree	RBFNet	98.73	86.11	79.57	100.0	90.71	100.00	100.00	86.17	74.80	94.38	91.05
ADTree Bagging	RBFNet	98.73	86.11	79.57	100.0	90.71	100.00	100.00	86.17	74.80	94.38	91.05
ADTree Boosting	RBFNet	98.73	86.11	79.57	100.0	90.71	100.00	100.00	86.17	74.80	94.38	91.05
BayesNetGenerator	RBFNet	94.47	84.86	64.62	100.0	90.71	100.00	100.00	73.90	78.12	83.74	87.04
Decision-Stump	RBFNet	93.23	87.15	66.21	100.0	90.71	100.00	100.00	86.13	78.57	71.04	87.30
Decision-Stump Bagging	RBFNet	93.23	87.15	66.21	100.0	90.71	100.00	100.00	86.13	78.57	71.04	87.30
Decision-Stump Boosting	RBFNet	93.23	87.15	66.21	100.0	90.71	100.00	100.00	86.13	78.57	71.04	87.30
Decision-Table	RBFNet	96.69	90.91	76.96	100.0	90.71	100.00	100.00	85.47	78.28	73.34	89.24
IB-k	RBFNet	92.34	78.20	43.37	100.0	95.09	100.00	100.00	79.43	74.32	59.80	82.26
LibSVM	RBFNet	92.43	84.37	58.33	100.0	90.71	100.00	100.00	80.17	75.63	51.86	83.35
MultiLayerPerceptron	RBFNet	95.78	84.49	55.34	100.0	91.99	100.00	100.00	77.07	82.63	63.30	85.06
Naïve-Bayes-Updatable	RBFNet	92.60	84.69	60.05	100.0	92.98	100.00	100.00	79.83	84.69	72.40	86.72
PART	RBFNet	95.37	82.28	79.62	100.0	94.40	100.00	100.00	76.83	75.31	75.80	87.96
Random-Forest	RBFNet	93.92	83.89	63.12	100.0	90.50	100.00	100.00	79.50	76.72	68.18	85.58
Random-Forest Bagging	RBFNet	93.92	83.89	63.12	100.0	90.50	100.00	100.00	79.50	76.72	68.18	85.58
Random-Forest Boosting	RBFNet	93.92	83.89	63.12	100.0	90.50	100.00	100.00	79.50	76.72	68.18	85.58
REPTree	RBFNet	94.36	85.42	70.62	100.0	94.40	100.00	100.00	77.43	80.92	69.84	87.30
SimpleLogistic	RBFNet	96.42	89.79	86.11	100.0	92.11	100.00	100.00	81.90	88.55	97.72	93.26
SMO	RBFNet	87.40	81.91	49.42	100.0	91.85	100.00	100.00	76.57	70.89	51.22	80.93
ZeroR	RBFNet	98.88	100.00	61.95	100.0	100.00	100.00	100.00	100.00	100.00	55.08	91.59
ADTree	SVMReg	97.16	83.85	91.66	99.20	90.56	98.58	99.97	93.27	81.52	94.26	93.00
ADTree Bagging	SVMReg	97.16	83.85	91.66	99.20	90.56	98.58	99.97	93.27	81.52	94.26	93.00
ADTree Boosting	SVMReg	97.16	83.85	91.66	99.20	90.56	98.58	99.97	93.27	81.52	94.26	93.00
BayesNetGenerator	SVMReg	85.87	71.18	72.21	98.36	90.71	97.80	97.01	90.33	79.97	83.00	86.64
Decision-Stump	SVMReg	78.28	75.89	64.57	99.66	90.20	99.30	98.80	96.67	80.40	60.38	84.41
Decision-Stump Bagging	SVMReg	78.28	75.89	64.57	99.66	90.20	99.30	98.80	96.67	80.40	60.38	84.41
Decision-Stump Boosting	SVMReg	78.28	75.89	64.57	99.66	90.20	99.30	98.80	96.67	80.40	60.38	84.41
Decision-Table	SVMReg	85.30	67.49	67.03	97.19	90.38	95.46	98.53	92.00	79.75	68.38	84.15
IB-k	SVMReg	71.32	55.92	52.29	96.20	90.60	93.24	93.18	90.43	79.49	36.08	75.88
LibSVM	SVMReg	73.00	60.52	47.45	96.34	95.40	95.96	93.58	97.13	81.36	36.82	77.76
MultiLayerPerceptron	SVMReg	87.02	62.52	53.51	96.76	90.48	93.90	95.39	90.77	79.53	44.26	79.41
Naïve-Bayes-Updatable	SVMReg	79.30	72.09	59.55	98.59	93.50	97.64	97.12	90.33	82.62	70.00	84.07
PART	SVMReg	85.02	66.90	85.25	97.26	90.20	96.00	96.53	91.70	80.42	66.66	85.59
Random-Forest	SVMReg	79.55	69.39	67.64	97.05	90.89	94.28	98.94	90.77	79.49	57.88	82.59
Random-Forest Bagging	SVMReg	79.55	69.39	67.64	97.05	90.89	94.28	98.94	90.77	79.49	57.88	82.59
Random-Forest Boosting	SVMReg	79.55	69.39	67.64	97.05	90.89	94.28	98.94	90.77	79.49	57.88	82.59
REPTree	SVMReg	88.29	72.16	74.49	98.17	90.21	96.24	98.78	92.93	84.01	54.36	84.96
SimpleLogistic	SVMReg	98.91	83.53	89.61	99.72	90.65	98.84	99.93	91.33	95.65	98.74	94.69
SMO	SVMReg	69.54	51.81	45.73	96.14	90.21	93.24	91.39	90.33	79.49	28.50	73.64
ZeroR	SVMReg	95.08	100.00	55.78	100.	100.00	100.00	100.00	100.00	97.60	40.88	88.93

Table 7.2b: Percentage of The Train set used for generating results from Fuzzy boundaries of regression based clusters process

Algorithm	Validation / Calibration / Fuzzy	Abs error	Rel error	RMSE	Sqrd error	Correlation	Pred. avg	AUC	margin	kappa	Precision	recall	lift	fallout	F measure	Accuracy
Random-Forest-Bagging	Fixed Split	0.094	0.094	0.210	0.058	0.878	0.240	0.923	0.032	0.693	1.020	0.693	5.935	0.029	0.752	0.9334
Random-Forest-Boosting	Fixed Split	0.071	0.071	0.223	0.066	0.781	0.239	0.914	0.020	0.694	0.904	0.694	5.837	0.029	0.745	0.9322
Random-Forest	Fixed Split	0.101	0.101	0.189	0.051	0.800	0.297	0.909	0.040	0.711	0.932	0.729	5.595	0.040	0.773	0.9288
RandomForest-Bagging	SVMRegFuzzy	0.119	0.119	0.199	0.057	0.696	0.296	0.917	0.081	0.684	0.813	0.729	6.486	0.060	0.746	0.9215
Random-Forest-Bagging	CrossVal	0.117	0.117	0.200	0.057	0.864	0.296	0.916	0.080	0.678	1.016	0.725	6.488	0.061	0.756	0.9215
RandomForest-Bagging	PaceFuzzy	0.116	0.116	0.198	0.057	0.703	0.297	0.913	0.085	0.691	0.819	0.735	6.615	0.062	0.753	0.9215
RandomForest-Bagging	RBFNetFuzzy	0.118	0.118	0.199	0.057	0.699	0.297	0.915	0.083	0.688	0.813	0.734	6.486	0.062	0.750	0.9215
RandomForest-Bagging	IsoFuzzy	0.120	0.120	0.199	0.057	0.699	0.296	0.912	0.069	0.688	0.816	0.731	6.506	0.061	0.750	0.9214
Random-Forest-Bagging	Platt	0.117	0.117	0.199	0.057	0.865	0.298	0.916	0.081	0.679	1.015	0.728	6.493	0.063	0.758	0.9214
RandomForest-Bagging	LinearFuzzy	0.117	0.117	0.199	0.057	0.697	0.296	0.915	0.076	0.684	0.820	0.728	6.613	0.061	0.746	0.9211
PART	Fixed Split	0.089	0.089	0.209	0.065	0.710	0.297	0.880	0.100	0.705	0.818	0.734	5.081	0.046	0.767	0.9208
REPTree	Fixed Split	0.107	0.107	0.212	0.063	0.766	0.298	0.875	0.106	0.681	0.897	0.713	5.143	0.052	0.752	0.9202
RandomForest-Boosting	PaceFuzzy	0.084	0.084	0.227	0.078	0.695	0.289	0.901	0.050	0.684	0.815	0.721	6.468	0.055	0.746	0.9198
RandomForest-Boosting	LinearFuzzy	0.085	0.085	0.227	0.078	0.685	0.288	0.897	0.050	0.671	0.815	0.710	6.457	0.055	0.733	0.9193
RandomForest-Boosting	RBFNetFuzzy	0.081	0.081	0.225	0.078	0.688	0.289	0.899	0.060	0.676	0.810	0.716	6.351	0.056	0.738	0.9193
MultiLayerPerceptron	Fixed Split	0.090	0.090	0.214	0.063	0.706	0.307	0.908	0.000	0.704	0.796	0.752	4.769	0.057	0.770	0.9191
Random-Forest-Boosting	CrossVal	0.086	0.086	0.229	0.079	0.754	0.289	0.900	0.050	0.663	0.904	0.708	6.489	0.057	0.733	0.9185
RandomForest-Boosting	SVMRegFuzzy	0.084	0.084	0.230	0.080	0.675	0.291	0.896	0.060	0.662	0.805	0.708	6.207	0.059	0.725	0.9177
RandomForest-Boosting	IsoFuzzy	0.091	0.091	0.228	0.077	0.677	0.289	0.901	0.060	0.663	0.810	0.706	6.394	0.057	0.726	0.9174
RandomForest	IsoFuzzy	0.117	0.117	0.210	0.063	0.684	0.287	0.893	0.050	0.671	0.816	0.710	6.458	0.056	0.736	0.9167
RandomForest	SVMRegFuzzy	0.118	0.118	0.211	0.063	0.693	0.288	0.898	0.030	0.683	0.811	0.721	6.420	0.058	0.748	0.9163
Random-Forest-Boosting	Platt	0.092	0.092	0.231	0.080	0.775	0.296	0.900	0.051	0.691	0.885	0.739	5.963	0.062	0.763	0.9162
RandomForest	PaceFuzzy	0.114	0.114	0.208	0.062	0.686	0.289	0.893	0.050	0.674	0.810	0.716	6.458	0.059	0.739	0.9159
RandomForest	LinearFuzzy	0.114	0.114	0.208	0.062	0.682	0.290	0.895	0.040	0.671	0.807	0.712	6.363	0.059	0.736	0.9158
RandomForest	RBFNetFuzzy	0.116	0.116	0.209	0.062	0.675	0.289	0.893	0.050	0.663	0.801	0.707	6.151	0.059	0.728	0.915
Random-Forest	CrossVal	0.115	0.115	0.210	0.062	0.754	0.288	0.893	0.040	0.667	0.892	0.710	6.196	0.059	0.740	0.9149
ADTree-Boosting	Fixed Split	0.111	0.111	0.221	0.060	0.655	0.255	0.921	0.003	0.653	0.736	0.699	4.777	0.059	0.721	0.9143
Random-Forest	Platt	0.115	0.115	0.210	0.062	0.688	0.314	0.893	0.040	0.681	0.784	0.752	6.089	0.083	0.761	0.9132
Decision-Table	Fixed Split	0.113	0.113	0.222	0.068	0.965	0.302	0.863	0.106	0.667	1.141	0.709	5.042	0.063	0.767	0.9129
MultiLayerPerceptron	Platt	0.118	0.118	0.235	0.072	0.674	0.295	0.902	0.001	0.672	0.752	0.745	5.196	0.069	0.743	0.9106
MultiLayerPerceptron	CrossVal	0.100	0.100	0.232	0.072	0.668	0.296	0.902	0.000	0.667	0.737	0.748	4.670	0.070	0.739	0.9096
PART	CrossVal	0.101	0.101	0.227	0.074	0.675	0.283	0.870	0.100	0.671	0.776	0.719	5.304	0.058	0.741	0.9094

Table 7.3a: Average performance for each learning algorithm by metric (average over ten problems)

Algorithm	Validation / Calibration / Fuzzy	Abs error	Rel error	RMSE	Sqrd error	Correlation	Pred. avg	AUC	margin	kappa	Precision	recall	lift	fallout	F measure	Accuracy
PART	Platt	0.101	0.101	0.227	0.074	0.675	0.285	0.870	0.100	0.671	0.775	0.721	5.301	0.061	0.742	0.9094
IB-k	Fixed Split	0.092	0.092	0.248	0.091	0.685	0.319	0.848	0.000	0.685	0.762	0.762	4.158	0.076	0.762	0.9091
ADTree-Boosting	PaceFuzzy	0.118	0.118	0.218	0.068	0.669	0.284	0.909	0.107	0.660	0.807	0.713	5.905	0.063	0.731	0.9089
MLP	LinearFuzzy	0.102	0.102	0.231	0.071	0.676	0.293	0.897	0.000	0.674	0.755	0.743	4.909	0.067	0.746	0.9087
REPTree	PaceFuzzy	0.116	0.116	0.225	0.072	0.662	0.290	0.888	0.104	0.658	0.766	0.723	5.130	0.067	0.729	0.9086
MLP	IsoFuzzy	0.103	0.103	0.237	0.073	0.668	0.313	0.904	0.000	0.666	0.738	0.759	4.689	0.087	0.745	0.9078
REPTree	IsoFuzzy	0.120	0.120	0.225	0.072	0.662	0.295	0.871	0.100	0.658	0.761	0.722	5.132	0.073	0.731	0.9078
REPTree	CrossVal	0.117	0.117	0.229	0.074	0.728	0.295	0.865	0.106	0.653	0.813	0.726	4.529	0.073	0.733	0.9078
REPTree	Platt	0.117	0.117	0.229	0.074	0.728	0.296	0.865	0.106	0.653	0.812	0.727	4.528	0.074	0.733	0.9077
MLP	PaceFuzzy	0.103	0.103	0.235	0.073	0.666	0.294	0.900	0.000	0.663	0.743	0.741	4.606	0.069	0.736	0.9073
REPTree	RBFNetFuzzy	0.120	0.120	0.230	0.075	0.650	0.297	0.881	0.104	0.646	0.741	0.727	4.263	0.074	0.719	0.907
REPTree	SVMRegFuzzy	0.124	0.124	0.234	0.075	0.716	0.296	0.863	0.111	0.642	0.794	0.721	4.313	0.074	0.795	0.9069
ADTree-Boosting	LinearFuzzy	0.118	0.118	0.218	0.069	0.660	0.294	0.909	0.107	0.651	0.784	0.718	5.604	0.072	0.724	0.9064
ADTree-Boosting	RBFNetFuzzy	0.121	0.121	0.223	0.070	0.658	0.305	0.906	0.106	0.652	0.762	0.731	5.150	0.084	0.730	0.9063
ADTree-Boosting	IsoFuzzy	0.119	0.119	0.218	0.069	0.653	0.297	0.904	0.105	0.644	0.781	0.713	5.341	0.076	0.718	0.9062
ADTree-Boosting	SVMRegFuzzy	0.119	0.119	0.220	0.069	0.661	0.299	0.907	0.105	0.653	0.804	0.730	5.675	0.078	0.729	0.9062
ADTree-Boosting	CrossVal	0.119	0.119	0.219	0.069	0.652	0.302	0.905	0.106	0.646	0.768	0.717	5.484	0.081	0.723	0.9059
PART	SVMRegFuzzy	0.107	0.107	0.231	0.075	0.660	0.274	0.873	0.100	0.655	0.757	0.703	4.585	0.052	0.723	0.9059
REPTree	LinearFuzzy	0.119	0.119	0.232	0.075	0.715	0.295	0.861	0.107	0.641	0.795	0.721	4.290	0.074	0.795	0.9058
PART	LinearFuzzy	0.104	0.104	0.238	0.081	0.658	0.294	0.859	0.100	0.655	0.743	0.726	4.535	0.071	0.730	0.9056
PART	PaceFuzzy	0.105	0.105	0.234	0.079	0.672	0.298	0.867	0.100	0.670	0.759	0.739	5.127	0.073	0.746	0.9056
PART	IsoFuzzy	0.105	0.105	0.236	0.079	0.664	0.291	0.858	0.100	0.660	0.753	0.724	4.656	0.067	0.735	0.905
MLP	SVMRegFuzzy	0.106	0.106	0.238	0.076	0.656	0.289	0.894	0.000	0.649	0.741	0.726	4.496	0.068	0.722	0.9049
ADTree-Boosting	Platt	0.156	0.156	0.227	0.069	0.648	0.307	0.905	0.123	0.643	0.733	0.742	4.241	0.084	0.721	0.9042
PART	RBFNetFuzzy	0.104	0.104	0.234	0.080	0.665	0.304	0.858	0.100	0.662	0.737	0.746	4.577	0.079	0.738	0.904
ADTree-Bagging	Fixed Split	0.181	0.181	0.251	0.074	0.790	0.252	0.917	0.041	0.622	0.947	0.667	5.500	0.072	0.711	0.9031
MLP	RBFNetFuzzy	0.109	0.109	0.241	0.077	0.669	0.305	0.894	0.000	0.668	0.738	0.757	4.546	0.079	0.746	0.9022
Decision Table	PaceFuzzy	0.130	0.130	0.238	0.076	0.717	0.293	0.850	0.106	0.638	0.833	0.711	5.993	0.078	0.798	0.9015
Decision Table	IsoFuzzy	0.129	0.129	0.235	0.077	0.724	0.295	0.850	0.107	0.644	0.845	0.712	6.188	0.081	0.805	0.9015
Decision Table	LinearFuzzy	0.133	0.133	0.247	0.080	0.689	0.295	0.836	0.106	0.607	0.810	0.695	5.580	0.081	0.765	0.898
Decision Table	RBFNetFuzzy	0.136	0.136	0.246	0.081	0.693	0.297	0.861	0.108	0.621	0.784	0.710	4.400	0.084	0.782	0.897
Decision-Table	CrossVal	0.131	0.131	0.250	0.083	0.795	0.295	0.865	0.100	0.624	0.956	0.710	5.720	0.082	0.722	0.8963

Table 7.3b: Average performance for each learning algorithm by metric (average over ten problems)

Algorithm	Validation / Calibration / Fuzzy	Abs error	Rel error	RMSE	Sqrd error	Correlation	Pred. avg	AUC	margin	kappa	Precision	recall	lift	fallout	F measure	Accuracy
Decision-Table	Platt	0.131	0.131	0.250	0.083	0.796	0.299	0.865	0.100	0.624	0.953	0.715	5.716	0.086	0.724	0.8961
IB-k	CrossVal	0.105	0.105	0.273	0.104	0.648	0.312	0.834	0.000	0.645	0.718	0.753	4.422	0.094	0.733	0.896
ADTree	Fixed Split	0.188	0.188	0.239	0.078	0.803	0.299	0.900	0.144	0.634	0.983	0.688	5.198	0.077	0.738	0.8959
Decision Table	SVMRegFuzzy	0.136	0.136	0.245	0.080	0.632	0.295	0.874	0.100	0.625	0.740	0.715	4.966	0.085	0.710	0.8955
Ibk	PaceFuzzy	0.106	0.106	0.274	0.104	0.647	0.313	0.833	0.000	0.645	0.718	0.752	4.395	0.095	0.732	0.8955
Ibk	LinearFuzzy	0.106	0.106	0.274	0.105	0.646	0.313	0.834	0.000	0.644	0.717	0.751	4.374	0.095	0.731	0.8952
IB-k	Platt	0.127	0.127	0.271	0.101	0.648	0.314	0.834	0.019	0.646	0.716	0.757	4.393	0.096	0.734	0.8947
ADTree-Bagging	Platt	0.194	0.194	0.251	0.081	0.693	0.304	0.903	0.141	0.608	0.840	0.695	5.826	0.098	0.704	0.8943
ADTree-Bagging	CrossVal	0.192	0.192	0.247	0.082	0.694	0.306	0.903	0.137	0.616	0.812	0.705	4.970	0.100	0.713	0.8937
ADTree-Bagging	SVMRegFuzzy	0.191	0.191	0.248	0.082	0.691	0.307	0.901	0.110	0.616	0.796	0.709	4.994	0.100	0.785	0.8935
Ibk	RBFNetFuzzy	0.109	0.109	0.279	0.107	0.640	0.315	0.834	0.000	0.637	0.707	0.755	4.057	0.097	0.725	0.8927
ADTree-Bagging	IsoFuzzy	0.192	0.192	0.248	0.083	0.691	0.305	0.900	0.125	0.616	0.799	0.707	5.092	0.100	0.785	0.8922
ADTree-Bagging	RBFNetFuzzy	0.191	0.191	0.248	0.083	0.693	0.306	0.901	0.127	0.617	0.807	0.708	5.396	0.102	0.787	0.892
ADTree-Bagging	PaceFuzzy	0.191	0.191	0.248	0.082	0.683	0.311	0.900	0.128	0.608	0.786	0.708	4.768	0.106	0.778	0.8917
ADTree-Bagging	LinearFuzzy	0.191	0.191	0.248	0.083	0.679	0.303	0.901	0.127	0.605	0.791	0.697	4.856	0.099	0.775	0.8907
SimpleLogistic	Fixed Split	0.196	0.196	0.276	0.096	0.702	0.300	0.877	0.051	0.628	0.851	0.690	4.920	0.082	0.728	0.8897
SMO	Fixed Split	0.111	0.111	0.300	0.111	0.697	0.298	0.798	0.000	0.620	0.858	0.679	5.080	0.082	0.721	0.8887
Ibk	IsoFuzzy	0.114	0.114	0.285	0.112	0.636	0.317	0.831	0.000	0.633	0.707	0.756	4.245	0.103	0.727	0.8878
ADTree	CrossVal	0.194	0.194	0.253	0.085	0.616	0.307	0.888	0.129	0.607	0.775	0.707	5.514	0.106	0.704	0.8877
Ibk	SVMRegFuzzy	0.115	0.115	0.288	0.114	0.633	0.323	0.835	0.000	0.628	0.697	0.761	3.894	0.106	0.721	0.8863
ADTree	Platt	0.202	0.202	0.260	0.086	0.817	0.321	0.888	0.138	0.565	0.931	0.687	2.752	0.122	0.690	0.8859
ADTree	RBFNetFuzzy	0.190	0.190	0.250	0.085	0.613	0.311	0.888	0.130	0.601	0.780	0.707	5.570	0.113	0.702	0.8853
Decision-Stump-Boosting	Fixed Split	0.170	0.170	0.269	0.084	0.695	0.212	0.884	0.016	0.537	0.955	0.570	5.633	0.052	0.631	0.8844
ADTree	LinearFuzzy	0.192	0.192	0.253	0.085	0.614	0.315	0.887	0.129	0.602	0.780	0.711	5.685	0.117	0.704	0.8843
ADTree	IsoFuzzy	0.191	0.191	0.252	0.085	0.610	0.316	0.887	0.132	0.599	0.756	0.713	5.147	0.118	0.701	0.8842
ADTree	PaceFuzzy	0.192	0.192	0.253	0.085	0.612	0.314	0.887	0.128	0.601	0.766	0.709	5.444	0.116	0.703	0.8842
ADTree	SVMRegFuzzy	0.193	0.193	0.253	0.085	0.614	0.316	0.887	0.130	0.603	0.770	0.722	5.336	0.118	0.705	0.8836
SMO	IsoFuzzy	0.119	0.119	0.309	0.119	0.676	0.291	0.796	0.000	0.605	0.812	0.689	6.070	0.097	0.783	0.8811
SMO	RBFNetFuzzy	0.120	0.120	0.311	0.120	0.674	0.293	0.795	0.000	0.604	0.807	0.690	5.978	0.099	0.782	0.8803
SimpleLogistic	CrossVal	0.165	0.165	0.261	0.085	0.612	0.291	0.888	0.003	0.607	0.768	0.690	6.068	0.098	0.707	0.8796
SMO	PaceFuzzy	0.121	0.121	0.314	0.121	0.669	0.286	0.788	0.000	0.598	0.819	0.672	6.145	0.096	0.776	0.879
BayesNetGen	IsoFuzzy	0.138	0.138	0.257	0.092	0.634	0.307	0.893	0.096	0.632	0.739	0.737	5.367	0.107	0.736	0.879

Table 7.3c: Average performance for each learning algorithm by metric (average over ten problems)

Algorithm	Validation / Calibration / Fuzzy	Abs error	Rel error	RMSE	Sqrd error	Correlation	Pred. avg	AUC	margin	kappa	Precision	recall	lift	fallout	F measure	Accuracy
SMO	LinearFuzzy	0.121	0.121	0.314	0.121	0.669	0.286	0.788	0.000	0.598	0.818	0.673	6.106	0.096	0.776	0.879
BayesNetGenerator	Platt	0.184	0.184	0.273	0.092	0.614	0.302	0.889	0.023	0.602	0.771	0.690	6.296	0.107	0.705	0.8789
Simple Logistic	IsoFuzzy	0.209	0.209	0.288	0.105	0.672	0.293	0.866	0.102	0.602	0.810	0.686	6.046	0.101	0.781	0.8788
SMO	SVMRegFuzzy	0.121	0.121	0.314	0.121	0.672	0.294	0.796	0.000	0.604	0.802	0.692	5.863	0.101	0.783	0.8788
Simple Logistic	SVMRegFuzzy	0.207	0.207	0.286	0.105	0.671	0.293	0.865	0.149	0.601	0.812	0.684	6.109	0.100	0.780	0.8786
Simple Logistic	RBFNetFuzzy	0.211	0.211	0.287	0.105	0.670	0.293	0.865	0.116	0.600	0.810	0.683	6.024	0.101	0.779	0.8784
Simple Logistic	PaceFuzzy	0.211	0.211	0.294	0.106	0.670	0.293	0.863	0.101	0.600	0.810	0.684	6.043	0.101	0.780	0.8782
Simple Logistic	LinearFuzzy	0.173	0.173	0.265	0.086	0.671	0.293	0.877	0.057	0.600	0.812	0.684	6.125	0.101	0.780	0.8781
BayesNetGen	RBFNetFuzzy	0.140	0.140	0.272	0.097	0.632	0.314	0.892	0.000	0.628	0.741	0.742	5.606	0.116	0.736	0.8754
SMO	Platt	0.130	0.130	0.324	0.125	0.649	0.290	0.788	0.006	0.578	0.785	0.675	5.150	0.100	0.688	0.8747
BayesNetGen	SVMRegFuzzy	0.141	0.141	0.272	0.096	0.631	0.315	0.891	0.000	0.628	0.735	0.745	5.498	0.116	0.736	0.8746
BayesNetGen	PaceFuzzy	0.141	0.141	0.275	0.098	0.631	0.317	0.892	0.000	0.627	0.735	0.746	5.482	0.118	0.736	0.8736
Decision-Stump-Boosting	CrossVal	0.174	0.174	0.261	0.091	0.636	0.287	0.874	0.119	0.568	0.759	0.670	3.713	0.098	0.677	0.8734
Decision-Stump-Boosting	Platt	0.204	0.204	0.270	0.092	0.636	0.288	0.874	0.141	0.567	0.758	0.670	3.710	0.099	0.677	0.8732
BayesNetGen	LinearFuzzy	0.142	0.142	0.276	0.099	0.633	0.318	0.891	0.000	0.629	0.739	0.749	5.571	0.119	0.739	0.8727
BayesNetGenerator	Fixed Split	0.142	0.142	0.284	0.098	0.636	0.336	0.898	0.000	0.631	0.732	0.763	4.436	0.116	0.739	0.8713
BayesNetGenerator	CrossVal	0.145	0.145	0.283	0.102	0.627	0.321	0.890	0.000	0.622	0.730	0.750	5.361	0.124	0.734	0.8694
SimpleLogistic	Platt	0.209	0.209	0.271	0.091	0.575	0.277	0.888	0.102	0.559	0.760	0.639	5.407	0.096	0.661	0.8692
Decision Stump-Boosting	SVMRegFuzzy	0.182	0.182	0.261	0.092	0.605	0.280	0.874	0.128	0.539	0.749	0.629	4.048	0.097	0.716	0.8672
Decision Stump-Boosting	IsoFuzzy	0.184	0.184	0.263	0.093	0.606	0.277	0.873	0.130	0.540	0.750	0.627	4.049	0.094	0.716	0.8671
Decision Stump-Boosting	RBFNetFuzzy	0.180	0.180	0.264	0.093	0.605	0.290	0.876	0.123	0.539	0.744	0.639	4.038	0.108	0.720	0.8666
Decision Stump-Boosting	LinearFuzzy	0.179	0.179	0.262	0.092	0.604	0.290	0.874	0.122	0.538	0.745	0.638	4.039	0.107	0.720	0.8666
Decision Stump-Boosting	PaceFuzzy	0.179	0.179	0.263	0.094	0.604	0.290	0.874	0.123	0.538	0.744	0.639	4.037	0.108	0.720	0.8664
SMO	CrossVal	0.134	0.134	0.339	0.134	0.634	0.313	0.784	0.000	0.565	0.765	0.682	4.920	0.114	0.682	0.8657
Decision-Stump-Bagging	Fixed Split	0.221	0.221	0.315	0.108	0.699	0.213	0.812	0.075	0.413	1.056	0.505	2.084	0.084	0.609	0.8471
Decision-Stump	Fixed Split	0.219	0.219	0.297	0.109	0.892	0.286	0.789	0.212	0.442	1.090	0.568	1.737	0.107	0.662	0.8449
NBU	IsoFuzzy	0.177	0.177	0.329	0.121	0.535	0.290	0.864	0.000	0.525	0.688	0.656	4.217	0.124	0.654	0.8431
Decision Stump-Bagging	LinearFuzzy	0.244	0.244	0.315	0.119	0.555	0.270	0.771	0.198	0.434	0.731	0.546	4.352	0.114	0.685	0.8408
Decision Stump-Bagging	IsoFuzzy	0.253	0.253	0.314	0.118	0.555	0.260	0.780	0.204	0.434	0.735	0.536	4.361	0.104	0.681	0.8405
Decision-Stump	CrossVal	0.242	0.242	0.318	0.122	0.740	0.271	0.758	0.203	0.434	0.973	0.547	3.866	0.115	0.640	0.8404
Decision-Stump	Platt	0.251	0.251	0.319	0.122	0.740	0.271	0.758	0.228	0.434	0.973	0.547	3.866	0.115	0.640	0.8404
Decision-Stump-Bagging	Platt	0.250	0.250	0.318	0.121	0.738	0.260	0.765	0.223	0.433	0.978	0.536	3.846	0.105	0.635	0.8401

Table 7.3d: Average performance for each learning algorithm by metric (average over ten problems)

Algorithm	Validation / Calibration / Fuzzy	Abs error	Rel error	RMSE	Sqrd error	Correlation	Pred. avg	AUC	margin	kappa	Precision	recall	lift	fallout	F measure	Accuracy
Decision Stump-Bagging	RBFNetFuzzy	0.248	0.248	0.316	0.120	0.553	0.265	0.772	0.200	0.432	0.729	0.541	4.345	0.110	0.680	0.839
Decision-Stump-Bagging	CrossVal	0.244	0.244	0.316	0.120	0.737	0.263	0.765	0.205	0.431	0.974	0.539	3.864	0.108	0.634	0.8389
Decision Stump-Bagging	PaceFuzzy	0.246	0.246	0.317	0.121	0.551	0.273	0.774	0.200	0.430	0.724	0.547	4.333	0.117	0.682	0.8378
Decision-Stump	LinearFuzzy	0.246	0.246	0.320	0.124	0.551	0.277	0.757	0.208	0.430	0.722	0.552	4.329	0.122	0.683	0.8377
Decision-Stump	PaceFuzzy	0.247	0.247	0.321	0.124	0.551	0.277	0.757	0.208	0.430	0.722	0.552	4.329	0.122	0.683	0.8377
Decision-Stump	RBFNetFuzzy	0.249	0.249	0.321	0.124	0.551	0.277	0.757	0.211	0.430	0.722	0.552	4.329	0.122	0.683	0.8377
Decision Stump-Bagging	SVMRegFuzzy	0.252	0.252	0.317	0.121	0.551	0.267	0.772	0.217	0.429	0.727	0.542	4.337	0.112	0.679	0.8375
Decision-Stump	IsoFuzzy	0.254	0.254	0.320	0.124	0.551	0.264	0.757	0.219	0.429	0.728	0.539	4.340	0.109	0.677	0.8374
Decision-Stump	SVMRegFuzzy	0.252	0.252	0.322	0.125	0.551	0.264	0.753	0.216	0.429	0.728	0.539	4.340	0.109	0.677	0.8374
NBU	RBFNetFuzzy	0.184	0.184	0.340	0.130	0.525	0.282	0.857	0.000	0.517	0.703	0.640	5.227	0.132	0.661	0.831
NBU	SVMRegFuzzy	0.191	0.191	0.348	0.135	0.515	0.278	0.857	0.000	0.504	0.690	0.637	4.647	0.130	0.650	0.8254
NBU	PaceFuzzy	0.197	0.197	0.360	0.142	0.500	0.282	0.853	0.000	0.487	0.673	0.637	4.166	0.137	0.635	0.8174
NBU	LinearFuzzy	0.197	0.197	0.359	0.142	0.503	0.283	0.854	0.000	0.490	0.676	0.640	4.282	0.137	0.638	0.8173
Naïve-Bayes-Updatable	Fixed Split	0.203	0.203	0.374	0.149	0.506	0.309	0.863	0.000	0.490	0.667	0.667	3.117	0.141	0.637	0.8117
Naïve-Bayes-Updatable	Platt	0.229	0.229	0.361	0.144	0.455	0.282	0.854	0.008	0.436	0.741	0.576	6.439	0.140	0.585	0.811
Naïve-Bayes-Updatable	CrossVal	0.204	0.204	0.373	0.152	0.493	0.292	0.854	0.000	0.476	0.671	0.642	4.073	0.148	0.628	0.8078
LibSVM	Fixed Split	0.219	0.219	0.410	0.219	0.508	0.228	0.664	0.000	0.350	0.787	0.464	4.488	0.136	0.494	0.7814
LibSVM	IsoFuzzy	0.233	0.233	0.411	0.233	0.471	0.323	0.675	0.000	0.364	0.766	0.587	6.894	0.237	0.531	0.7667
LibSVM	RBFNetFuzzy	0.235	0.235	0.416	0.235	0.599	0.320	0.671	0.000	0.356	0.715	0.578	6.214	0.236	0.654	0.7647
LibSVM	PaceFuzzy	0.235	0.235	0.416	0.235	0.519	0.320	0.669	0.000	0.355	0.752	0.574	5.935	0.236	0.581	0.7647
LibSVM	Platt	0.236	0.236	0.419	0.236	0.713	0.319	0.667	0.000	0.352	0.961	0.570	5.136	0.236	0.551	0.7636
LibSVM	LinearFuzzy	0.237	0.237	0.417	0.237	0.527	0.419	0.670	0.000	0.357	0.696	0.674	5.824	0.334	0.654	0.7632
LibSVM	SVMRegFuzzy	0.238	0.238	0.421	0.238	0.465	0.422	0.674	0.000	0.360	0.627	0.685	5.340	0.337	0.594	0.7615
LibSVM	CrossVal	0.295	0.295	0.475	0.295	0.685	0.403	0.630	0.000	0.271	0.854	0.597	4.764	0.336	0.518	0.7052
ZeroR	Fixed Split	0.364	0.364	0.411	0.182	NaN	0.200	0.500	0.303	0.000	1.063	0.200	0.250	0.200	0.308	0.6909
ZeroR	CrossVal	0.392	0.392	0.430	0.200	NaN	0.400	0.500	0.332	0.000	0.498	0.400	0.500	0.400	0.304	0.5965
ZeroR	Platt	0.392	0.392	0.430	0.200	NaN	0.400	0.500	0.332	0.000	0.498	0.400	0.500	0.400	0.304	0.5965
ZeroR	IsoFuzzy	0.398	0.398	0.431	0.200	NaN	0.400	0.500	0.341	0.000	0.373	0.400	1.000	0.400	0.513	0.5965
ZeroR	LinearFuzzy	0.394	0.394	0.430	0.200	NaN	0.400	0.500	0.336	0.000	0.373	0.400	1.000	0.400	0.513	0.5965
ZeroR	PaceFuzzy	0.394	0.394	0.430	0.200	NaN	0.400	0.500	0.336	0.000	0.373	0.400	1.000	0.400	0.513	0.5965
ZeroR	RBFNetFuzzy	0.398	0.398	0.431	0.200	NaN	0.400	0.500	0.339	0.000	0.373	0.400	1.000	0.400	0.513	0.5965
ZeroR	SVMRegFuzzy	0.402	0.402	0.432	0.200	NaN	0.400	0.500	0.345	0.000	0.373	0.400	1.000	0.400	0.513	0.5965

Table 7.3e: Average performance for each learning algorithm by metric (average over ten problems)

7.1.1.4 Critical analysis of Results generated from supervised learning processes of Major study

(a) Performance of Processes

Table 7.1 and Table 7.3 include the performance of eight processes over twenty algorithms used for present research in descending order of their accuracy. Table 7.2 includes percentage of data used for training after removing a part of the original train set through Fuzzy boundaries of regression based clusters. Overall average of data used for experimentation of proposed process is around 86%. This figure can be reduced to 76.56% if algorithms and processes having a train set lesser or equal to average size of the train set, are considered. Data varies from 10.03% to 95.93% for experimentation through proposed process. Cross validation includes flat 90% data for whole experimentation. Table 7.1 represents accuracy of algorithms in descending order. Random Forest algorithm has clearly topped in Table 7.1. Fixed Split process has got top positions. As Fixed Split does not fulfill reliability criteria, so pre processed and post processed results may be more reliable. Fourth position is acquired by SVM Regression based pre processing through proposed process. Fifth position is acquired by Cross Validation based pre processing. Sixth, Seventh and Eighth positions are acquired by Pace Regression, RBF Network and Isotonic Regression algorithms based on proposed processes. Ninth position is acquired by post processing through Platt Scaling. Fuzzy boundaries of Regression based clusters based process has registered impressive performance with a reduction in the size of the train set. Few algorithms like ZeroR and LibSVM indicated very less reduction in their train dataset through their slow learning behavior. For rest of the classification algorithms there was significant reduction in the size of the train set. Table 7.4 indicates the relationship between sizes of the train set as well as size of model. Most of algorithms' model increase in size with increasing number of records. Figure 7.1 displays this relationship very well. Algorithms ADTree, Decision Stump, Naïve Bayes Updatable and ZeroR are exceptions in this regard. ADTree and Decision Stump algorithms keep size of models restricted to level 1 so their sizes continue to be of same size. Naïve Bayes Updatable algorithm calculates same type of parameters for all models like Mean and Standard Deviation etc. ZeroR algorithm optimizes the model same way, but learning capability of ZeroR algorithm is found to be negligible. Experimentation experience of present

research with ZeroR algorithm indicates this algorithm to be non learning algorithm. After investigating ZeroR algorithm more closely, it was found that this algorithm checks the majority of cases in the train set. Most of the datasets used for study had more negative cases, so output of these problems from ZeroR algorithm resulted in negative cases only. Such results may not be appropriate for any classification task. Sizes of models from rest of the algorithms increase with increasing number records in the train set. Reduction in the size of the train set without compromising the performance of outcome is impressive contribution from proposed process. Table 7.5 and Figure 7.2 include the sizes of models for different processes. Cross Validation may be used to create complete or partial model. Sizes of complete model are equal to the sizes of Fixed Split models. Sizes of models from Fuzzy Boundaries of Regression based Cluster process have shown consistent reduction in their sizes as compared to other types of models. Within these five regression algorithms sizes of models for Isotonic regression and SVM regression algorithms are better than other three types.

	Size of The Train Set (Number of Records)									
Algorithm	**1000**	**2000**	**3000**	**4000**	**5000**	**6000**	**7000**	**8000**	**9000**	**10000**
ADTree	5	5	5	5	5	5	5	5	5	5
ADTree-Bagging	18	18	18	18	18	18	18	18	18	18
ADTree-Boosting	18	18	18	18	18	18	18	18	18	18
BayesNetGenerator	304	596	887	1179	1470	1761	2052	2343	2634	2925
Decision-Stump	3	3	3	3	3	3	3	3	3	3
Decision-Stump-Bagging	5	5	5	5	5	5	5	5	5	5
Decision-Stump-Boosting	5	5	5	5	5	5	5	5	5	5
Decision-Table	303	600	893	1187	1489	1773	2066	2378	2665	2967
IB-k	150	295	441	586	732	877	1023	1168	1314	1459
LibSVM	233	463	693	924	1153	1383	1613	1844	2075	2305
MultiLayerPerceptron	309	600	892	1182	1476	1766	2056	2347	2645	2933
Naïve-Bayes-Simple	5	5	5	5	5	5	5	5	5	5
PART	54	77	139	135	149	207	179	191	249	294
Random-Forest	449	861	1281	1712	2123	2485	2935	3313	3745	4178
Random-Forest-Bagging	3721	7124	10567	14102	17253	20639	24284	27662	31133	34640
Random-Forest-Boosting	3403	6597	9854	12975	15837	19296	22441	25577	28828	32152
REPTree	14	11	14	22	24	18	20	32	23	25
SimpleLogistic	594	1175	1759	2341	2926	3507	4088	4669	5257	5839
SMO	446	883	1320	1756	2195	2631	3067	3502	3946	4381
ZeroR	2	2	2	2	2	2	2	2	2	2

Table 7.4: Sizes of models in Kilobytes for different algorithms with increasing sizes of the train set

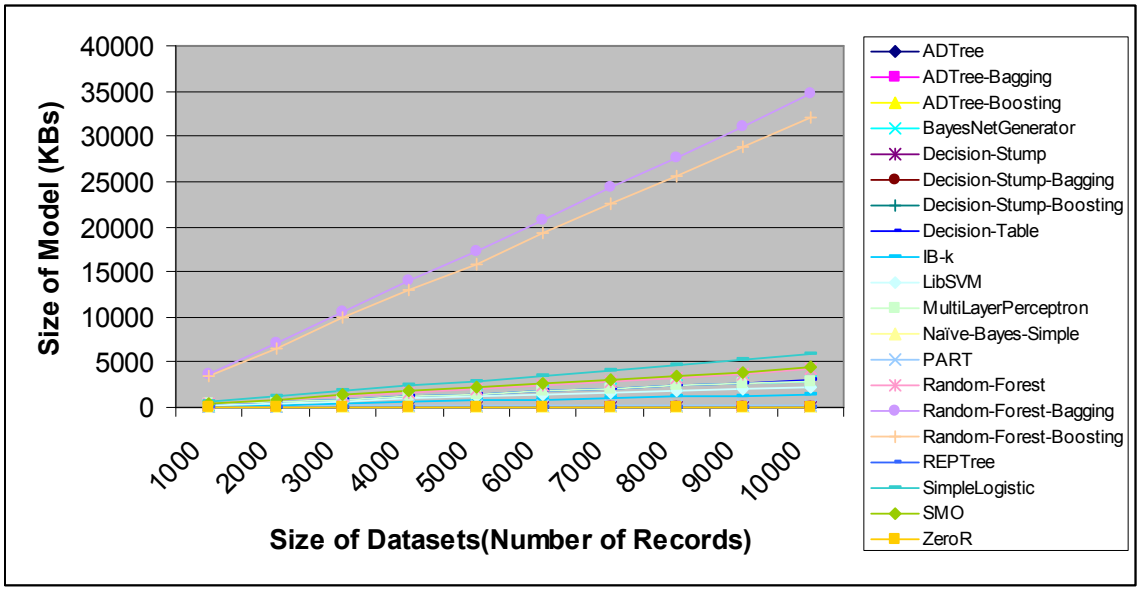

Figure 7.1: Relationship of size of models with increasing sizes of datasets

Algorithm	Fixed Split	Platt	Cross Val		Iso Fuzzy	Linear Fuzzy	Pace Fuzzy	RBF Net Fuzzy	SVM Reg Fuzzy
			Complete	Partial					
ADTree	5	6	5	5	5	5	5	5	5
ADTree-Bagging	18	18	18	18	18	18	18	18	18
ADTree-Boosting	18	18	18	18	18	18	18	18	18
BayesNetGenerator	2925	2926	2925	2634	2254	2790	2779	2764	2514
Decision-Stump	3	4	3	3	3	3	3	3	3
Decision-Stump-Bagging	5	6	5	5	5	5	5	5	5
Decision-Stump-Boosting	5	6	5	5	5	5	5	5	5
Decision-Table	2967	2967	2967	2674	2643	2893	2921	2870	2534
IB-k	1459	1460	1459	1314	1023	1381	1374	1348	1042
LibSVM	2305	2305	2305	2074	1728	2185	2184	2141	1690
MultiLayerPerceptron	2933	1474	2935	2645	2422	2828	2818	2812	2558
Naïve-Bayes-Simple	5	6	6	5	5	5	5	5	5
PART	294	294	294	262	303	322	280	231	258
Random-Forest	4178	4178	4178	3763	4096	4147	4159	4163	4090
Random-Forest-Bagging	34640	34641	34640	31231	34004	34375	34518	34485	33824
Random-Forest-Boosting	32152	32153	32152	28782	30175	31754	31760	31705	30547
REPTree	25	25	25	31	21	34	37	37	27
SimpleLogistic	5839	5839	5839	5258	5265	5786	5786	5631	5775
SMO	4381	4381	4381	3946	3273	4126	4105	3832	3046
ZeroR	2	2	2	2	2	2	2	2	2

Table 7.5: Sizes of models in Kilobytes for different algorithms over different processes

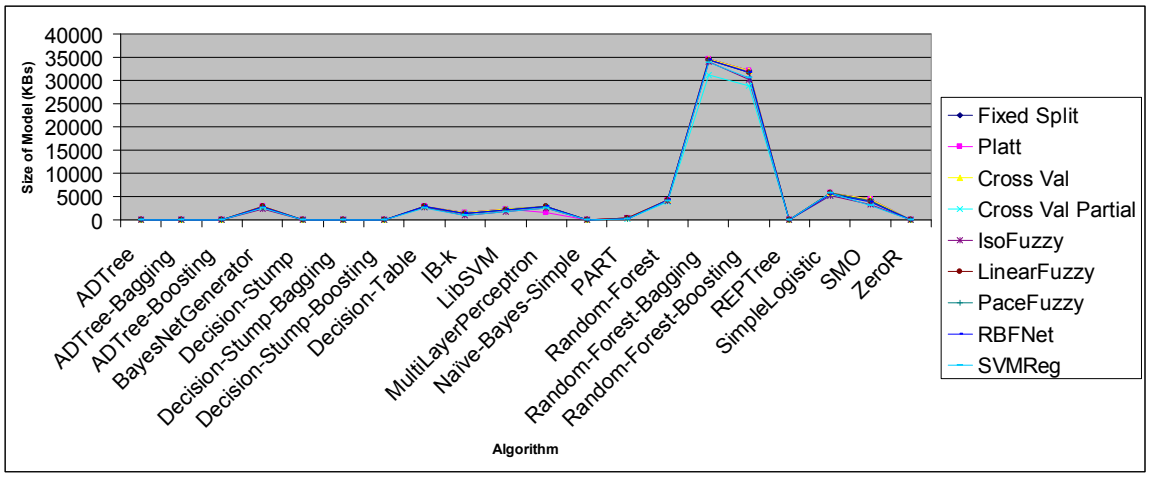

Figure 7.2: Sizes of models from all algorithms for different processes

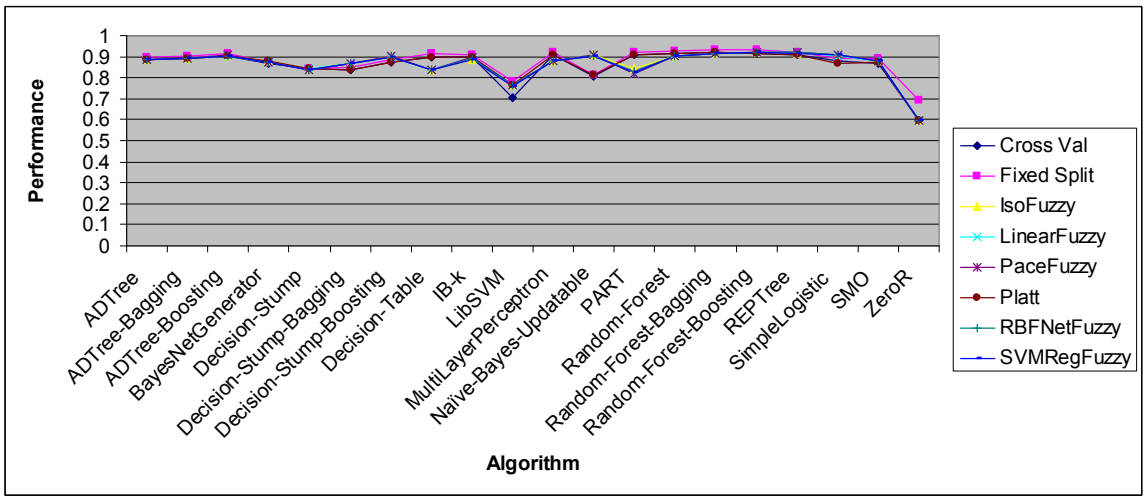

Figure 7.3: Performance of Different algorithms for different processes

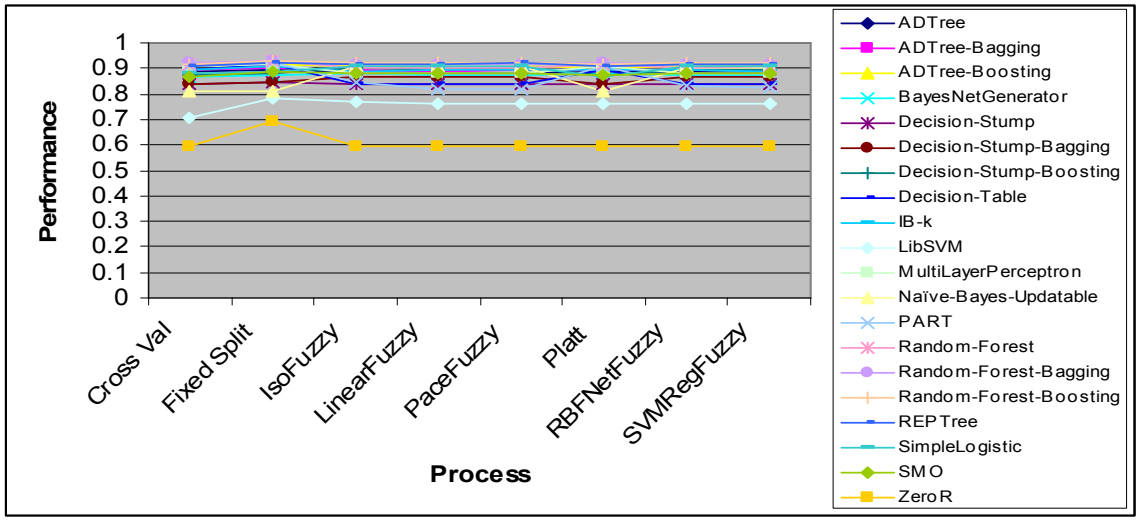

Figure 7.4: Performance of different Processes for different algorithms

(b) Performance of algorithms

Most of the top positions are occupied by Random Forest, Random Forest Bagging and Random Forest Boosting algorithms. Other classification algorithms that registered their presence in top twenty five positions are PART, REPTree and Multi Layer Perceptron, all of these performed through Fixed Split. Other algorithms that registered their existence in top fifty positions are ADTree Boosting, Decision Table and IB-k. Most of the algorithms register their presence in top hundred positions except LibSVM and ZeroR. Figures 7.3 and 7.4 represent the performance of different algorithms for different processes and performance of different processes for different algorithms respectively.

(c) Performance of Regression based algorithms for Fuzzy Boundaries of Regression Based Clusters process

All five regression algorithms perform reasonably well for supervised learning. But, experiences with experimentation indicate that RBF Network algorithm does not support proposed theory so well due to its small range of output being unsupportive to the formation of clusters for present theory. Rest of the four algorithms support proposed theory very well. Among these regression algorithms SVM Regression performs consistently well, whereas Isotonic Regression algorithm deviates many a time.

(d) Performance of processes for different algorithms over different problems/datasets included for present research

Figure 7.5 includes the performance of all ten datasets over eight processes for all twenty algorithms used for major study. Boosted algorithms received very varied performance over different datasets. Fuzzy Boundaries of Regression based cluster process works reasonably well over included all five regression algorithms. Experimental experience with RBF Network indicates very less pruning due to small range of output from this algorithm. Output used to be scattered very less towards extremes and very high concentration was towards center and creates difficulty in forming clusters through classification output. As very less pruning is involved, its output can not be trusted the way Fixed Split lack in reliability criteria. Cross Validation seems to dip a lot in performance for LibSVM, Naïve Bayes Updatable and ZeroR,

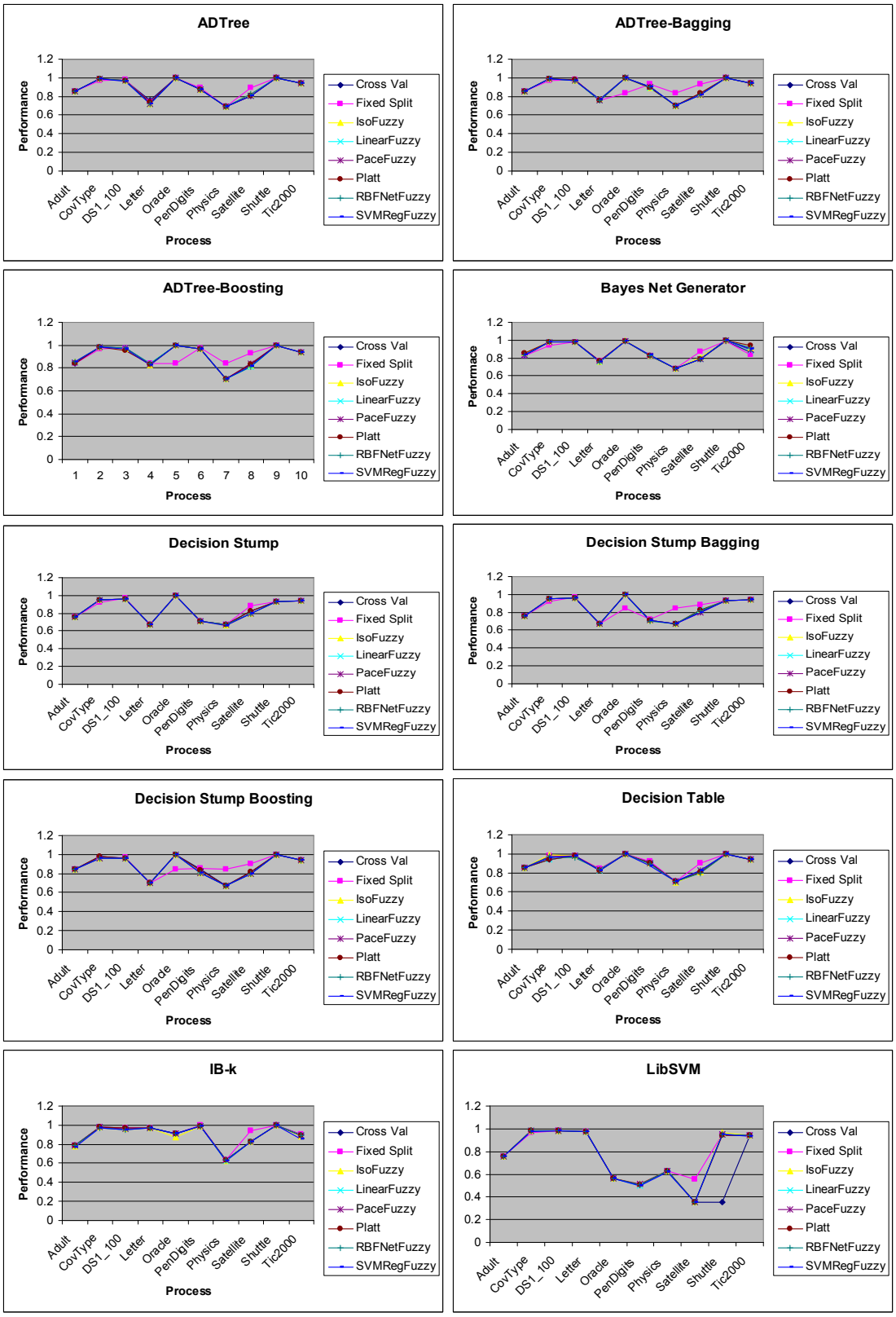

Figure 7.5a: Performance of Individual algorithms for different processes over different datasets/problems

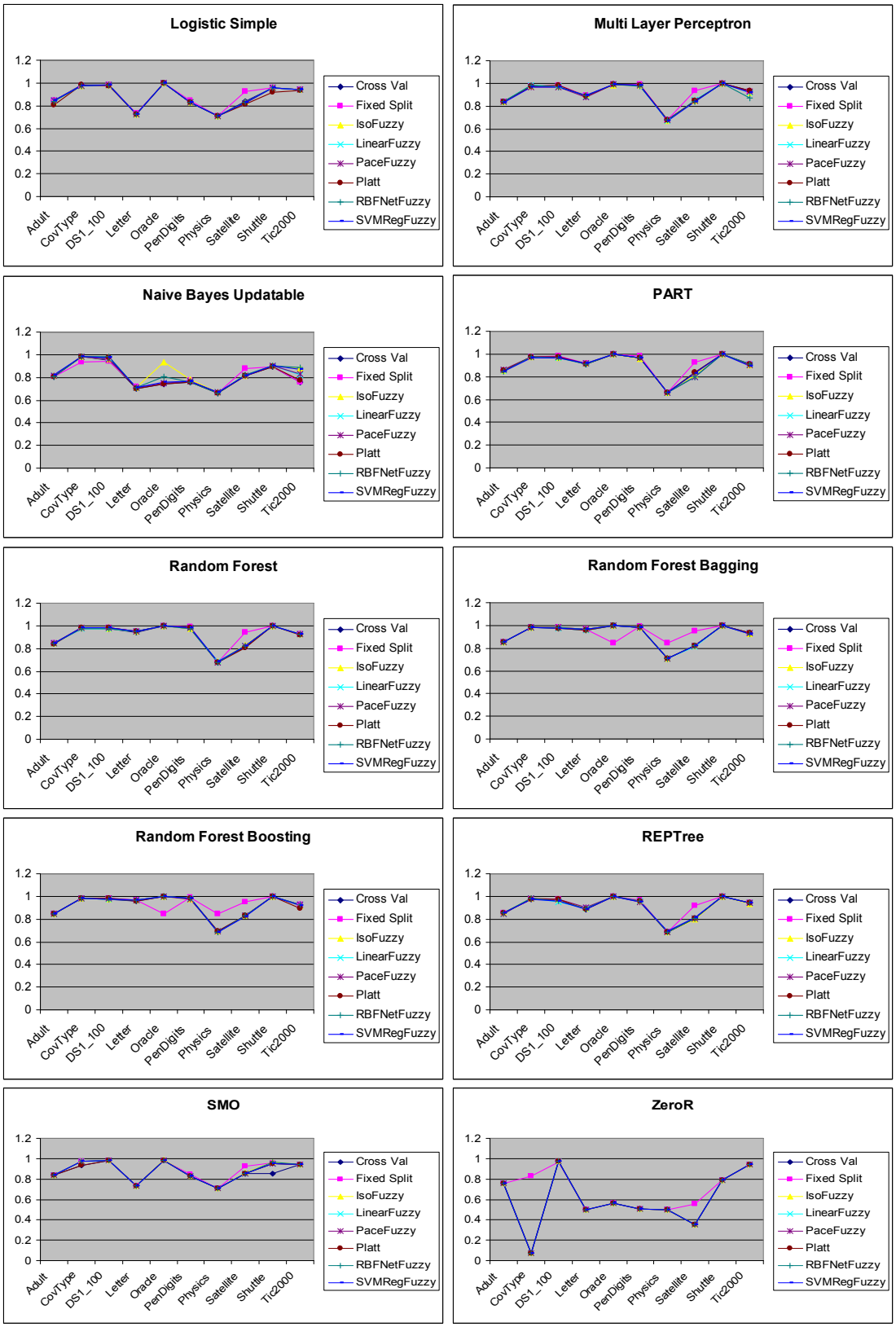

Figure 7.5b: Performance of Individual algorithms for different processes over different datasets/problems.

Algorithm	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th	13th	14th	15th	16th	17th	18th	19th	20th
ADTree	0.12	0.04	0.07	0	0.02	0.04	0.04	0.06	0	0.08	0.01	0.09	0.08	0.06	0.11	0	0.02	0	0.06	0.03
ADTree-Bagging	0.08	0.11	0.05	0	0.05	0.03	0.09	0	0.04	0.02	0.07	0.14	0.04	0.04	0.04	0	0	0	0.11	0.05
ADTree-Boosting	0.06	0.04	0.04	0.08	0.09	0.11	0	0.05	0.14	0	0.13	0	0.1	0.03	0.12	0	0	0.1	0.03	0
BayesNetGenerator	0	0	0.07	0.09	0.04	0.03	0	0.08	0.05	0.16	0.02	0.11	0.07	0	0	0.2	0.09	0.03	0	0
Decision-Stump	0.09	0	0	0	0	0	0.07	0.07	0	0.08	0	0.18	0	0.08	0.12	0.16	0.08	0	0.11	0
Decision-Stump-Bagging	0.01	0	0	0	0	0	0.18	0.03	0	0.01	0.07	0.05	0	0.1	0.13	0.08	0.12	0	0.14	0
Decision-Stump-Boosting	0.02	0	0.1	0	0	0	0.09	0.15	0	0.06	0.12	0	0.13	0.08	0.12	0.09	0	0	0.09	0
Decision-Table	0.11	0.04	0.09	0	0.05	0.14	0	0	0.19	0.01	0.14	0.03	0	0	0.02	0	0.11	0	0	0.01
IB-k	0.07	0.16	0.03	0.04	0.01	0.09	0.12	0	0.09	0	0.03	0	0.05	0	0	0.06	0.1	0.07	0.04	0
LibSVM	0	0	0.13	0	0.1	0.02	0.07	0.04	0.05	0.01	0	0	0	0	0	0	0	0.14	0.04	0.4
MultiLayerPerceptron	0	0.07	0.09	0.08	0.05	0.07	0.04	0.07	0	0.11	0.03	0	0.05	0.01	0	0.09	0	0	0.05	0
Naïve-Bayes-Updatable	0	0	0	0	0.03	0.08	0.08	0.14	0.03	0.05	0.01	0.01	0.1	0.26	0.1	0.12	0.01	0.11	0	0
PART	0.13	0.12	0.02	0.1	0.09	0.03	0.12	0.07	0	0	0.05	0.13	0	0	0	0	0.1	0.07	0.01	0
Random-Forest	0	0.16	0.11	0.05	0.12	0.05	0	0	0.01	0.07	0	0.03	0.09	0	0	0.08	0	0.11	0	0
Random-Forest-Bagging	0.1	0.1	0.03	0.14	0.14	0	0	0	0.05	0.04	0.16	0.13	0	0	0.03	0	0	0.09	0	0
Random-Forest-Boosting	0.1	0.06	0.14	0.1	0.07	0.01	0	0	0.1	0.05	0.14	0.01	0.15	0	0.07	0.02	0	0.09	0	0
REPTree	0.1	0.1	0.03	0.09	0.03	0.14	0.1	0	0	0.09	0.02	0.09	0.06	0	0	0.1	0	0	0	0.18
SimpleLogistic	0.01	0	0	0.08	0.08	0.09	0	0.06	0.14	0.05	0	0	0.08	0.16	0.08	0	0.03	0.05	0	0
SMO	0	0	0	0.15	0.03	0.07	0	0.18	0.11	0.11	0	0	0	0.18	0.06	0	0	0	0.09	0.05
ZeroR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.34	0.14	0.23	0.28

Table 7.6: Bootstrap Analysis of Overall Rank by Mean Performance across Problems and Metrics

(e) Analysis of results from Bootstrap analysis

The bootstrap analysis complements the t-test and is considered to be better among statisticians. The results depend on the choice of problems and metrics. What impact might have on the results, if other problems were selected or performance were evaluated on other metrics? For example, IBk and PART perform better in many aspects, but are lesser accurate than Random Forest, so get inferior position in performance metric tables. Certain transformations have been done over results of different metrics for bootstrap analysis. This type of conversion is justified, because certain metrics like errors and loss metrics represent the complement of performance, whereas metric like Normalized absolute Error and Root Relative Squared error has very wide range over different algorithms and may affect the results badly. Error metrics and loss metrics were converted into their complement values i.e. if x is the error then value included for bootstrap analysis is (1-x). All metrics like Lift having range beyond 0 to 1 were converted into a range from 0 to 1 by using following model: if x is the value then (x-min)/(max-min) is calculated to convert x into a range [0, 1], where min and max are the minimum and maximum values of that metric over all algorithms. Table 7.6 shows the frequency that each method ranks 1st, 2nd, 3rd, etc. The 0.13 entry for PART algorithm in the column for 1st place, tells us that there are only 13% chances that PART would have placed 1st in the table of results, if we had selected other problems and/or metrics. Random Forest has consistently performed well by getting 10% first position for Random Forest, Random Forest Bagging and Random Forest Boosting. ADTree, Decision Stump, Decision Table and IBk have got first position with numeric figures of 12%, 9%, 11% and 7% respectively. ADTree, ADTree Bagging and ADTree Boosting has registered slight decline in performance, whereas Decision Stump, Decision Stump Bagging and Decision Stump Boosting has registered substantial decline in their performance. ADTree Bagging and IB-k algorithms performed well at second position. On the other hand, LibSVM and ZeroR continued to perform worst here as well.

(f) Comparison of Results with benchmark studies

There are two comprehensive studies that have been performed yet. First one is Statlog (King et al. [111]) and other is recent one (Caruana et al. [30]). One of the major differences between earlier two studies and current study, is about the selection of

datasets i.e. earlier studies were mostly based on small datasets, whereas present study includes most of the datasets that are bigger in size and simple rule of probability states that increasing number of instances produces more accurate as well as reliable results and minimizes the chances of deviation. When Statlog study was conducted, algorithms like Decision Stump and Random Forest etc. were not being developed and data mining was in its initial phase of development. During two decades data mining field has become mature enough. Statlog study presented the results for individual datasets. We compiled and processed the data for comparison and found that piecewise linear classifier DIPOL92 to be performing best for their tests, whereas Decision Tree was ranked second followed by the Back Propagation and kNN (k Nearest Neighbor) etc. Clearly, the absence of better algorithms like Random Forest and Decision Stump etc. at that time kept the high quality performance far away from current standards.

Other recently conducted study (Caruana et al. [30]) states that Boosted decision tree with Platt scaling algorithm is the best performer, whereas Random Forest with platt scaling is the second best performer. Experiments were performed through cross validation, Platt Scaling and Isotonic Regression. Top performers were Boosted Decision Tree, Random Forest, Bagged decision tree, SVM (Support Vector Machine), ANN (Artificial Neural Network) etc. Results of present study have marked Random Forest to be best performer followed by PART, Decision Table and Multi Layer Perceptron algorithms. Results of these studies are not comparable directly because of different domain of problems and sizes of datasets used. However, we still provide an overview of results from that study with present study in Table 7.7. Present study includes 10 problems, 20 algorithms over eight processes, whereas previous study included 11 problems, 10 algorithms over three processes viz. Fixed Split, Isotonic Regression and Platt Scaling. There is a significant difference between the average accuracy of both studies.

7.1.2 Minor Study

Minor study includes five algorithms and experiments are performed over three other calibration methods than major study like Additive Regression, Logistic Regression and Isotonic Regression. Limitation for regression based methods is that these require fully numeric values even for class attribute, all the datasets used for present study were

numeric values except. Most of the classification algorithms do not support the processing of numeric class, so in depth investigations found five algorithms out of twenty algorithms fully supportive to this minor study. Multi Layer Perceptron was suitable in most of the aspects, but it created problems, while experimentation with Additive Regression. So, present minor study included five algorithms viz. Decision Table, Decision Stump, IB-k, LibSVM and ZeroR for experimentation. On Additive regression ten fold cross validation has been applied and study is performed through meta classifiers. Logit Boost involves internal cross validation, so fixed split experimentation is performed. For Isotonic regression, stacking is done in conjunction with other algorithms and ten fold cross validation is performed.

7.1.2.1 Performance by Problem

Table 7.8 includes accuracy of five algorithms for all ten datasets that are involved for study and are ranked in descending order, based on their average accuracy. Five algorithms used for study are IBk, Decision Stump, Decision Table, LibSVM and ZeroR across six dimensions i. e. Fixed Split, Cross Validation, Platt Scaling, Additive Regression, Logit Boost and Isotonic Regression. Results indicate the better performance through Logit Boost calibration, followed by Additive Regression. Isotonic Regression has degraded the performance of the algorithms. IBk and Decision Table has topped the chart with calibration and individually. Additive Regression has enhanced the performance of ZeroR algorithm and has uplifted its performance significantly.

7.1.2.2 Performance by metrics

Table 7.9 includes averages of fifteen metrics involved in study and different metrics are positioned in descending order according to average accuracy. For few metrics output was generated to be NaN (**N**ot **a N**umber) and Infinity, for such metrics, averages were calculated over remaining values, excluding the count of such values. These values are pointed with an ‘NaN’ or with ‘Inf’, if all the values (for all ten problems) are NaN or Infinity.

Previous Study			Present Research		
Algorithm	**Validation / Calibration**	**Average Accuracy**	**Algorithm**	**Validation / Calibration / Fuzzy**	**Average Accuracy**
Boosted Decision Tree	Platt	0.843	Random-Forest-Bagging	Fixed Split	0.9334
Random Forest	Platt	0.872	Random-Forest-Boosting	Fixed Split	0.9322
Bagged Decision Tree	Fixed Split	0.846	Random-Forest	Fixed Split	0.9288
Boosted Decision Tree	Iso	0.826	RandomForest-Bagging	SVMRegFuzzy	0.9215
Random Forest	Fixed Split	0.872	Random-Forest-Bagging	CrossVal	0.9215
Bagged Decision Tree	Platt	0.841	RandomForest-Bagging	PaceFuzzy	0.9215
Random Forest	Iso	0.861	RandomForest-Bagging	RBFNetFuzzy	0.9215
Bagged Decision Tree	Iso	0.826	RandomForest-Bagging	IsoFuzzy	0.9214
SVM	Platt	0.824	Random-Forest-Bagging	Platt	0.9214
ANN	Fixed Split	0.803	RandomForest-Bagging	LinearFuzzy	0.9211
SVM	Iso	0.813	PART	Fixed Split	0.9208
ANN	Platt	0.815	REPTree	Fixed Split	0.9202
ANN	Iso	0.803	RandomForest-Boosting	PaceFuzzy	0.9198
Boosted Decision Tree	Fixed Split	0.834	RandomForest-Boosting	LinearFuzzy	0.9193
k-NN	Platt	0.757	RandomForest-Boosting	RBFNetFuzzy	0.9193
k-NN	Fixed Split	0.756	MultiLayerPerceptron	Fixed Split	0.9191
k-NN	Iso	0.755	Random-Forest-Boosting	CrossVal	0.9185
Boosted Decision Stump	Platt	0.724	RandomForest-Boosting	SVMRegFuzzy	0.9177
SVM	Fixed Split	0.817	RandomForest-Boosting	IsoFuzzy	0.9174
Boosted Decision Stump	Iso	0.709	RandomForest	IsoFuzzy	0.9167
Boosted Decision Stump	Fixed Split	0.741	RandomForest	SVMRegFuzzy	0.9163
Decision Tree	Iso	0.648	Random-Forest-Boosting	Platt	0.9162
Decision Tree	Fixed Split	0.647	RandomForest	PaceFuzzy	0.9159
Decision Tree	Platt	0.651	RandomForest	LinearFuzzy	0.9158
Logistic Regression	Fixed Split	0.636	RandomForest	RBFNetFuzzy	0.915
Logistic Regression	Iso	0.627	Random-Forest	CrossVal	0.9149
Logistic Regression	Platt	0.63	ADTree-Boosting	Fixed Split	0.9143
Naïve Bayes	Iso	0.579	Random-Forest	Platt	0.9132
Naïve Bayes	Platt	0.576	Decision-Table	Fixed Split	0.9129
Naïve Bayes	Fixed Split	0.496	MultiLayerPerceptron	Platt	0.9106

Table 7.7: Comparison of previous and current study through Average Accuracy

Algorithm	Validation / Calibration / Fuzzy	Adult	CovType	DS1_100	Letter	Oracle	PenDigits	Physics	Satellite	Shuttle	Tic2000	Average
Decision-Table	Fixed Split	0.857	0.978	0.977	0.849	1.000	0.920	0.708	0.902	0.999	0.940	0.9129
IB-k	Fixed Split	0.786	0.979	0.973	0.972	0.913	0.997	0.630	0.943	0.999	0.901	0.9091
Decision-Table	PaceFuzzy	0.854	0.974	0.978	0.831	1.000	0.906	0.708	0.827	0.999	0.939	0.9015
Decision-Table	IsoFuzzy	0.857	0.985	0.979	0.839	1.000	0.906	0.700	0.811	0.998	0.939	0.9015
Decision-Table	LinearFuzzy	0.857	0.942	0.975	0.828	1.000	0.905	0.708	0.827	0.999	0.939	0.8980
Decision-Table	RBFNetFuzzy	0.857	0.972	0.964	0.838	1.000	0.895	0.708	0.800	0.998	0.939	0.8970
Decision-Table	Cross Val	0.857	0.938	0.980	0.832	1.000	0.909	0.708	0.804	0.998	0.939	0.8963
Decision-Table	Platt	0.857	0.938	0.980	0.828	1.000	0.906	0.708	0.807	0.998	0.939	0.8961
IB-k	Cross Val	0.786	0.976	0.966	0.971	0.913	0.989	0.630	0.826	0.999	0.906	0.8960
Decision-Table	SVMRegFuzzy	0.857	0.958	0.975	0.823	1.000	0.872	0.708	0.827	0.998	0.938	0.8955
Ibk	PaceFuzzy	0.785	0.976	0.965	0.972	0.913	0.990	0.629	0.830	0.999	0.897	0.8955
Ibk	LinearFuzzy	0.785	0.976	0.964	0.972	0.912	0.990	0.629	0.829	0.999	0.895	0.8952
IB-k	Platt	0.786	0.976	0.965	0.970	0.913	0.989	0.630	0.824	0.999	0.895	0.8947
Ibk	RBFNetFuzzy	0.783	0.974	0.953	0.972	0.913	0.990	0.630	0.825	0.999	0.888	0.8927
Ibk	IsoFuzzy	0.775	0.976	0.961	0.972	0.875	0.990	0.625	0.824	0.999	0.881	0.8878
Ibk	SVMRegFuzzy	0.771	0.972	0.948	0.971	0.911	0.990	0.620	0.829	0.999	0.853	0.8863
Decision-Stump	Fixed Split	0.760	0.922	0.970	0.671	1.000	0.710	0.670	0.879	0.927	0.940	0.8449
Decision-Stump	Cross Val	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.823	0.928	0.939	0.8404
Decision-Stump	Platt	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.823	0.928	0.939	0.8404
Decision-Stump	LinearFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.796	0.928	0.939	0.8377
Decision-Stump	PaceFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.796	0.928	0.939	0.8377
Decision-Stump	RBFNetFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.670	0.796	0.928	0.939	0.8377
Decision-Stump	IsoFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.667	0.796	0.928	0.939	0.8374
Decision-Stump	SVMRegFuzzy	0.760	0.952	0.958	0.668	1.000	0.706	0.667	0.796	0.928	0.939	0.8374
LibSVM	Fixed Split	0.760	0.967	0.981	0.971	0.562	0.505	0.627	0.553	0.948	0.940	0.7814
LibSVM	IsoFuzzy	0.760	0.983	0.983	0.974	0.562	0.515	0.624	0.358	0.969	0.939	0.7667
LibSVM	RBFNetFuzzy	0.760	0.982	0.982	0.974	0.562	0.511	0.627	0.358	0.951	0.939	0.7647
LibSVM	PaceFuzzy	0.760	0.981	0.983	0.974	0.562	0.513	0.626	0.358	0.950	0.939	0.7647
LibSVM	Platt	0.760	0.981	0.983	0.972	0.562	0.511	0.627	0.358	0.943	0.939	0.7636
LibSVM	LinearFuzzy	0.760	0.982	0.983	0.974	0.562	0.497	0.626	0.358	0.951	0.939	0.7632
LibSVM	SVMRegFuzzy	0.760	0.975	0.982	0.975	0.562	0.496	0.624	0.358	0.951	0.932	0.7615
LibSVM	Cross Val	0.760	0.981	0.983	0.972	0.562	0.511	0.627	0.358	0.358	0.939	0.7052
ZeroR	Fixed Split	0.760	0.829	0.970	0.501	0.562	0.505	0.502	0.553	0.789	0.940	0.6909
ZeroR	Cross Val	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	IsoFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	LinearFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	PaceFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	Platt	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	RBFNetFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965
ZeroR	SVMRegFuzzy	0.760	0.077	0.971	0.497	0.562	0.511	0.502	0.358	0.789	0.939	0.5965

Table 7.8: Accuracy of algorithms from minor study over ten problems and their mean performance in descending order

Algorithm	Validation/ Calibration/Fuzzy	Abs. error	Rel. error	RMSE	Sqrd. error	Corr-elation	Pred. average	AUC	margin	Kappa	precision	recall	Lift	Fallout	F-measure	Accuracy
Decision-Table	Fixed Split	0.113	0.113	0.222	0.068	0.965	0.302	0.863	0.106	0.667	1.141	0.709	5.042	0.063	0.767	0.9129
IB-k	Fixed Split	0.092	0.092	0.248	0.091	0.685	0.319	0.848	0.000	0.685	0.762	0.762	4.158	0.076	0.762	0.9091
Decision Table	LogitBoost	0.126	0.126	0.236	0.068	0.662	0.306	0.909	0.004	0.651	0.788	0.711	5.219	0.076	0.726	0.9061
Decision-Table	PaceFuzzy	0.130	0.130	0.238	0.076	0.717	0.293	0.850	0.106	0.638	0.833	0.711	5.993	0.078	0.798	0.9015
Decision-Table	IsoFuzzy	0.129	0.129	0.235	0.077	0.724	0.295	0.850	0.107	0.644	0.845	0.712	6.188	0.081	0.805	0.9015
Decision-Table	LinearFuzzy	0.133	0.133	0.247	0.080	0.689	0.295	0.836	0.106	0.607	0.810	0.695	5.580	0.081	0.765	0.8980
Decision-Table	RBFNetFuzzy	0.136	0.136	0.246	0.081	0.693	0.297	0.861	0.108	0.621	0.784	0.710	4.400	0.084	0.782	0.8970
Ibk	LogitBoost	0.104	0.104	0.271	0.103	0.651	0.312	0.834	0.000	0.648	0.721	0.754	4.495	0.093	0.735	0.8968
Decision-Table	CrossVal	0.131	0.131	0.250	0.083	0.795	0.295	0.865	0.100	0.624	0.956	0.710	5.720	0.082	0.722	0.8963
Decision-Table	Platt	0.131	0.131	0.250	0.083	0.796	0.299	0.865	0.100	0.624	0.953	0.715	5.716	0.086	0.724	0.8961
IB-k	CrossVal	0.105	0.105	0.273	0.104	0.648	0.312	0.834	0.000	0.645	0.718	0.753	4.422	0.094	0.733	0.8960
Decision-Table	SVMRegFuzzy	0.136	0.136	0.245	0.080	0.632	0.295	0.874	0.100	0.625	0.740	0.715	4.966	0.085	0.710	0.8955
Ibk	PaceFuzzy	0.106	0.106	0.274	0.104	0.647	0.313	0.833	0.000	0.645	0.718	0.752	4.395	0.095	0.732	0.8955
Ibk	AddReg	0.105	Inf	0.272	0.104	0.652	0.314	0.000	1.000	0.639	0.665	0.762	3.822	0.114	0.695	0.8954
Ibk	LinearFuzzy	0.106	0.106	0.274	0.105	0.646	0.313	0.834	0.000	0.644	0.717	0.751	4.374	0.095	0.731	0.8952
IB-k	Platt	0.127	0.127	0.271	0.101	0.648	0.314	0.834	0.019	0.646	0.716	0.757	4.393	0.096	0.734	0.8947
Ibk	RBFNetFuzzy	0.109	0.109	0.279	0.107	0.640	0.315	0.834	0.000	0.637	0.707	0.755	4.057	0.097	0.725	0.8927
Ibk	IsoFuzzy	0.114	0.114	0.285	0.112	0.636	0.317	0.831	0.000	0.633	0.707	0.756	4.245	0.103	0.727	0.8878
Ibk	SVMRegFuzzy	0.115	0.115	0.288	0.114	0.633	0.323	0.835	0.000	0.628	0.697	0.761	3.894	0.106	0.721	0.8863
Decision Stump	LogitBoost	0.168	0.168	0.253	0.086	0.606	0.292	0.885	0.107	0.599	0.758	0.691	5.166	0.094	0.695	0.8836
Decision-Stump	Fixed Split	0.219	0.219	0.297	0.109	0.892	0.286	0.789	0.212	0.442	1.090	0.568	1.737	0.107	0.662	0.8449
Decision-Stump	CrossVal	0.242	0.242	0.318	0.122	0.740	0.271	0.758	0.203	0.434	0.973	0.547	3.866	0.115	0.640	0.8404
Decision-Stump	Platt	0.251	0.251	0.319	0.122	0.740	0.271	0.758	0.228	0.434	0.973	0.547	3.866	0.115	0.640	0.8404
Decision-Stump	LinearFuzzy	0.246	0.246	0.320	0.124	0.551	0.277	0.757	0.208	0.430	0.722	0.552	4.329	0.122	0.683	0.8377
Decision-Stump	PaceFuzzy	0.247	0.247	0.321	0.124	0.551	0.277	0.757	0.208	0.430	0.722	0.552	4.329	0.122	0.683	0.8377
Decision-Stump	RBFNetFuzzy	0.249	0.249	0.321	0.124	0.551	0.277	0.757	0.211	0.430	0.722	0.552	4.329	0.122	0.683	0.8377
Decision-Stump	IsoFuzzy	0.254	0.254	0.320	0.124	0.551	0.264	0.757	0.219	0.429	0.728	0.539	4.340	0.109	0.677	0.8374
Decision-Stump	SVMRegFuzzy	0.252	0.252	0.322	0.125	0.551	0.264	0.753	0.216	0.429	0.728	0.539	4.340	0.109	0.677	0.8374
Decision Table	AddReg	0.153	Inf	0.271	0.085	0.638	0.313	0.000	1.000	0.296	0.302	1.000	1.000	1.000	0.428	0.7917
Decision Stump	AddReg	0.189	Inf	0.273	0.093	0.613	0.320	0.000	1.000	0.208	0.346	1.000	1.078	0.900	0.456	0.7891

Table 7.9a: Average performance for each learning algorithm of minor study by metric (average over ten problems)

Algorithm	Validation/ Calibration/Fuzzy	Abs. error	Rel. error	RMSE	Sqrd. error	Corr-elation	Pred. average	AUC	margin	Kappa	precision	recall	Lift	Fallout	F-measure	Accuracy
LibSVM	Fixed Split	0.219	0.219	0.410	0.219	0.508	0.228	0.664	0.000	0.350	0.787	0.464	4.488	0.136	0.494	0.7814
LibSVM	IsoFuzzy	0.233	0.233	0.411	0.233	0.471	0.323	0.675	0.000	0.364	0.766	0.587	6.894	0.237	0.531	0.7667
LibSVM	RBFNetFuzzy	0.235	0.235	0.416	0.235	0.599	0.320	0.671	0.000	0.356	0.715	0.578	6.214	0.236	0.654	0.7647
LibSVM	PaceFuzzy	0.235	0.235	0.416	0.235	0.519	0.320	0.669	0.000	0.355	0.752	0.574	5.935	0.236	0.581	0.7647
LibSVM	Platt	0.236	0.236	0.419	0.236	0.713	0.319	0.667	0.000	0.352	0.961	0.570	5.136	0.236	0.551	0.7636
LibSVM	LinearFuzzy	0.237	0.237	0.417	0.237	0.527	0.419	0.670	0.000	0.357	0.696	0.674	5.824	0.334	0.654	0.7632
LibSVM	SVMRegFuzzy	0.238	0.238	0.421	0.238	0.465	0.422	0.674	0.000	0.360	0.627	0.685	5.340	0.337	0.594	0.7615
LibSVM	LogitBoost	0.245	0.245	0.396	0.209	0.390	0.323	0.702	0.025	0.345	0.662	0.574	5.266	0.245	0.522	0.7562
LibSVM	CrossVal	0.295	0.295	0.475	0.295	0.685	0.403	0.630	0.000	0.271	0.854	0.597	4.764	0.336	0.518	0.7052
LibSVM	AddReg	0.392	Inf	0.430	0.200	0.000	0.371	0.000	1.000	0.000	0.302	1.000	1.000	1.000	0.428	0.6978
ZeroR	AddReg	0.392	Inf	0.430	0.200	0.000	0.371	0.000	1.000	0.000	0.302	1.000	1.000	1.000	0.428	0.6978
Decision Stump	IsoReg	0.348	Inf	0.430	0.200	0.000	0.371	0.000	1.000	0.000	0.302	1.000	1.000	1.000	0.428	0.6978
Decision Table	IsoReg	0.348	Inf	0.430	0.200	0.000	0.371	0.000	1.000	0.000	0.302	1.000	1.000	1.000	0.428	0.6978
Ibk	IsoReg	0.348	Inf	0.430	0.200	0.000	0.371	0.000	1.000	0.000	0.302	1.000	1.000	1.000	0.428	0.6978
LibSVM	IsoReg	0.348	Inf	0.430	0.200	0.000	0.371	0.000	1.000	0.000	0.302	1.000	1.000	1.000	0.428	0.6978
ZeroR	IsoReg	0.348	Inf	0.430	0.200	0.000	0.371	0.000	1.000	0.000	0.302	1.000	1.000	1.000	0.428	0.6978
ZeroR	Fixed Split	0.364	0.364	0.411	0.182	NaN	0.200	0.500	0.303	0.000	1.063	0.200	0.250	0.200	0.308	0.6909
ZeroR	LogitBoost	0.392	0.392	0.430	0.200	0.000	0.400	0.500	0.332	0.000	0.187	0.400	0.500	0.400	0.257	0.5965
ZeroR	CrossVal	0.392	0.392	0.430	0.200	NaN	0.400	0.500	0.332	0.000	0.498	0.400	0.500	0.400	0.304	0.5965
ZeroR	IsoFuzzy	0.398	0.398	0.431	0.200	NaN	0.400	0.500	0.341	0.000	0.373	0.400	1.000	0.400	0.513	0.5965
ZeroR	LinearFuzzy	0.394	0.394	0.430	0.200	NaN	0.400	0.500	0.336	0.000	0.373	0.400	1.000	0.400	0.513	0.5965
ZeroR	PaceFuzzy	0.394	0.394	0.430	0.200	NaN	0.400	0.500	0.336	0.000	0.373	0.400	1.000	0.400	0.513	0.5965
ZeroR	Platt	0.392	0.392	0.430	0.200	NaN	0.400	0.500	0.332	0.000	0.498	0.400	0.500	0.400	0.304	0.5965
ZeroR	RBFNetFuzzy	0.398	0.398	0.431	0.200	NaN	0.400	0.500	0.339	0.000	0.373	0.400	1.000	0.400	0.513	0.5965
ZeroR	SVMRegFuzzy	0.402	0.402	0.432	0.200	NaN	0.400	0.500	0.345	0.000	0.373	0.400	1.000	0.400	0.513	0.5965

Table 7.9b: Average performance for each learning algorithm of minor study by metric (average over ten problems)

7.1.2.3 Critical analysis of Results generated from supervised learning processes of Minor study

Five algorithms were used for minor study to be evaluated over eleven process viz. eight processes from major study and three new processes.

(a) Performance of Processes

Analysis of all eleven processes indicate that newly added process LogitBoost performs slightly better than other pre processing and post processing based processes. All five regression based algorithm based fuzzy boundaries of regression based clusters processes perform reasonably well. Isotonic regression is also applied through stacking with classification algorithm, but it performs very poorly. Newly added process Additive regression for post processing of data also performs poorly. Table 7.8 and Figure 7.6 display the performance of different algorithms over different processes averaged over all problems/datasets.

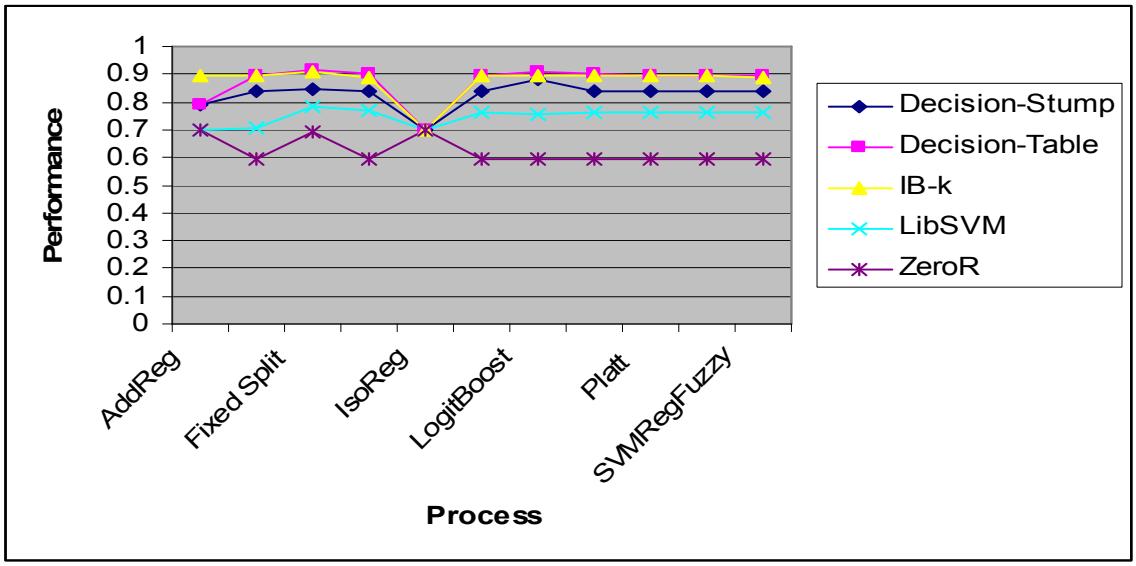

Figure 7.6: Performance of Different algorithms for different processes

(b) Performance of algorithms

Among all five algorithms used for minor study Decision Tree algorithm has performed better than all other algorithms. IB-k algorithm has also performed quite close to Decision-Table, followed by decision Stump and LibSVM. ZeroR has performed poorly in

non learning way. Again ZeroR algorithms has behaved in non learning way and present study's outcome discards the use of ZeroR algorithm for classification purposes. Table 7.9 and Figure 7.7 display the performance of different algorithms for different processes averaged over all ten problems/datasets.

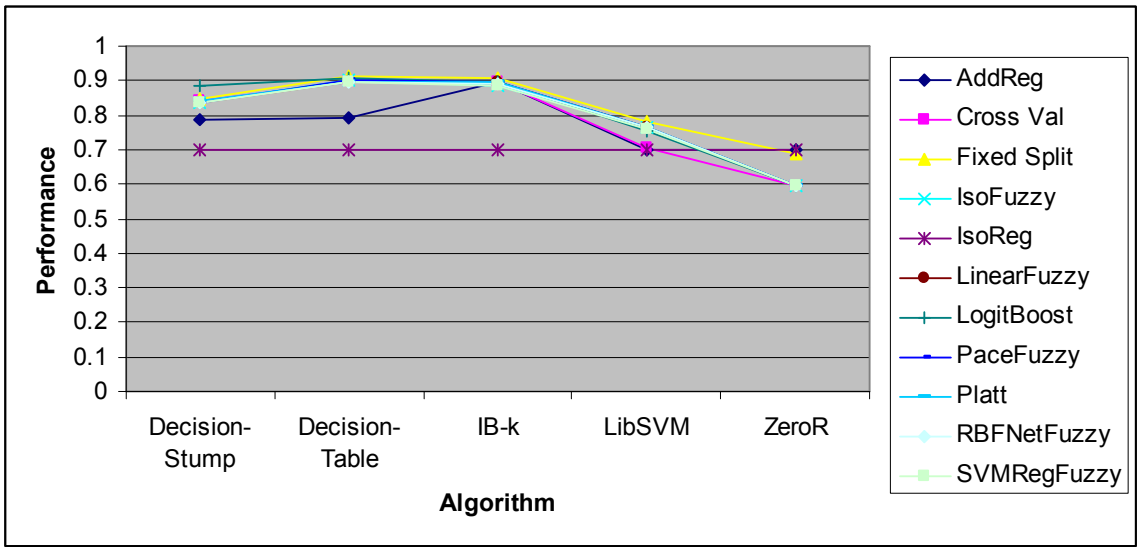

Figure 7.7: Performance of different Processes for different algorithms

(c) Performance of processes for different algorithms over different problems/datasets included for present research

Figure 7.8 includes the performance of all ten datasets over eleven processes for individual algorithm used for minor study. Decision Stump and Decision Table algorithms received very varied performance over different datasets. IB-k algorithm receives very worse performance for Additive Regression. Additive Regression and Stacking through Isotonic Regression perform poorly for all algorithms.

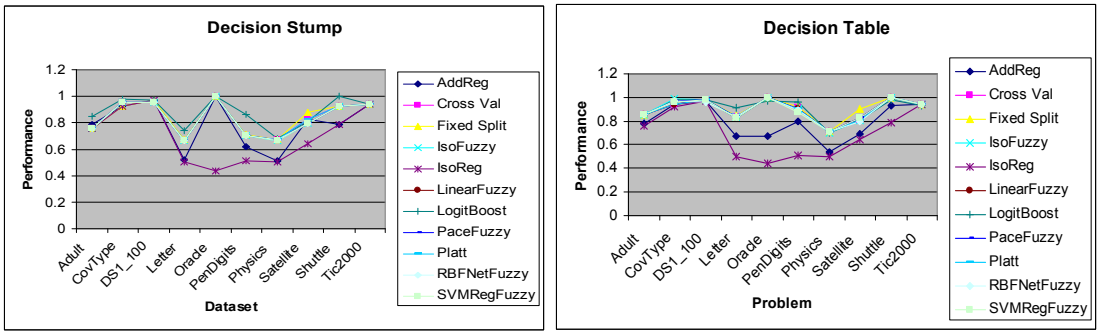

Figure 7.8a: Performance of Individual algorithms for different processes over different datasets/problems

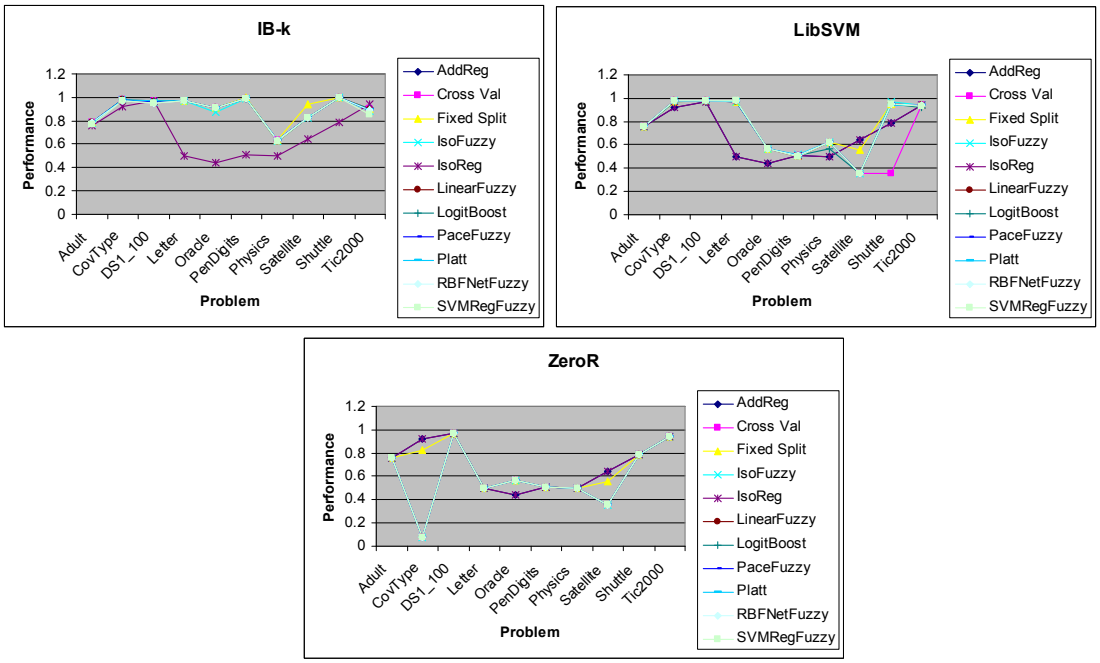

Figure 7.8b: Performance of Individual algorithms for different processes over different datasets/problems

7.1.2.4 Comparison of Performance from Regression Algorithms of Fuzzy Boundaries of Regression Based Clusters process with Stacked Algorithms

In this section performance of different regression algorithms used for proposed process are compared with same regression algorithms stacked with same classification algorithms of minor study presented in this chapter. Clearly, proposed process performs better than stacking process in four cases out of five as shown in Table 7.10 and Figure 7.9. ZeroR algorithm is evaluated to be non learner. Here good outcome is only due to large number of negative classes used for training, otherwise it doesn't learn or classify anything. So, the proposed process behave better than stacking as well and is also not limited by nominal or numeric class constraints imposed by processes like stacking etc.

Algorithm	Fuzzy		Stacking	
	Fuzzy Regression	Accuracy	Regression Stacking	Accuracy
Decision-Stump	IsoFuzzy	0.837393	Iso	0.697791
Decision-Table	IsoFuzzy	0.901458	Iso	0.697791
Ibk	IsoFuzzy	0.887834	Iso	0.697791
LibSVM	IsoFuzzy	0.76666	Iso	0.697791
ZeroR	IsoFuzzy	0.596525	Iso	0.697791
Decision-Stump	LinearFuzzy	0.837678	Linear	0.697791
Decision-Table	LinearFuzzy	0.897955	Linear	0.697791
Ibk	LinearFuzzy	0.895173	Linear	0.697791
LibSVM	LinearFuzzy	0.76318	Linear	0.697791
ZeroR	LinearFuzzy	0.596525	Linear	0.697791
Decision-Stump	PaceFuzzy	0.837678	Pace	0.697791
Decision-Table	PaceFuzzy	0.901505	Pace	0.697791
Ibk	PaceFuzzy	0.895461	Pace	0.697791
LibSVM	PaceFuzzy	0.764671	Pace	0.697791
ZeroR	PaceFuzzy	0.596525	Pace	0.697791
Decision-Stump	RBFNetFuzzy	0.837678	RBFNet	0.697791
Decision-Table	RBFNetFuzzy	0.896997	RBFNet	0.697791
Ibk	RBFNetFuzzy	0.892729	RBFNet	0.697791
LibSVM	RBFNetFuzzy	0.76469	RBFNet	0.697791
ZeroR	RBFNetFuzzy	0.596525	RBFNet	0.697791
Decision-Stump	SVMRegFuzzy	0.837393	SVMReg	0.697791
Decision-Table	SVMRegFuzzy	0.89549	SVMReg	0.697791
Ibk	SVMRegFuzzy	0.886314	SVMReg	0.697791
LibSVM	SVMRegFuzzy	0.761506	SVMReg	0.697791
ZeroR	SVMRegFuzzy	0.596525	SVMReg	0.697791

Table 7.10: Comparison of results from Fuzzy process and stacking processes

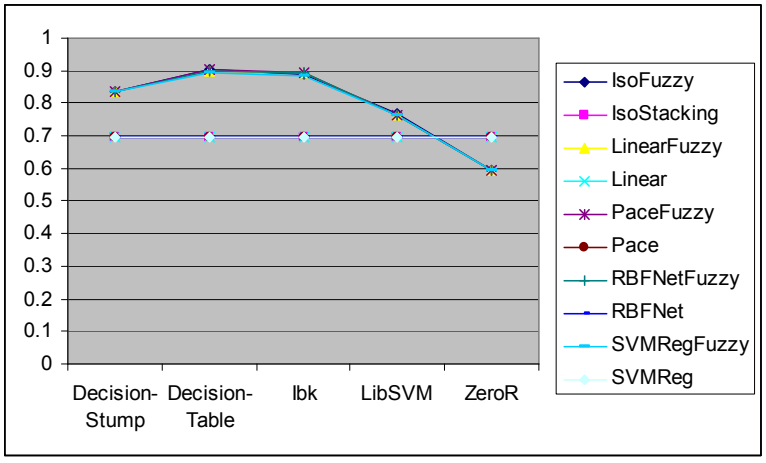

Figure 7.9: Comparison of performance from different algorithms from different Fuzzy theory based processes and from stacking of same algorithms

7.1.3 Graphical Comparison

A graphical comparison involving ROC and Precision/Recall graphs of algorithms is prepared for Cross Validated experiments on Adult dataset.

7.1.3.1 ROC Curves

An ROC graph is a technique for visualizing, organizing and selecting classifiers based on their performance. ROC graphs are two-dimensional graphs in which True Positive rate is plotted on the Y axis and False Positive rate is plotted on the X axis. An ROC graph is compared on the basis of behavior of the curve in graph. A curve sharply rising towards Y axis is considered to be better than the diagonal or a curve sharply bending towards X axis. Clearly, in Figures 7.10 and 7.12, better performing algorithms like Random forests, boosted decision stumps etc. have their curves rising towards Y-axis marking better performance for them. SMO, Decision Stump etc. are rising diagonally indicate average performance. ZeroR algorithm with only one point in the middle indicates static output for the algorithm.

7.1.3.2 Precision/Recall Curves

Precision is the ratio of True Positives to the Sum of True Positives and False Positives, Recall is the ratio of True Positives to the Sum of True Positives and False Positives. A Precision/Recall curve bending towards origin is considered to be worst performance, whereas a curve rising away from origin towards 1 for X and Y axis collectively, is considered to be better performance. Clearly, in Figures 7.11 and 7.13, algorithms like Random Forests, ADTrees etc. are rising away from origin indicate better performance, whereas diagonal curves for algorithms like SMO etc. indicate average performance. These graphs are indicating the scenario of Adult problem. Curves can dramatically change for other problems depending upon their results.

Figures 7.10 and 7.11 display the output from Fixed Split experimentations, whereas Figures 7.12 a nd 7.13 compares output from five regression algorithms over classification algorithms through proposed process ‘Fuzzy Boundaries of Regression based clusters’.

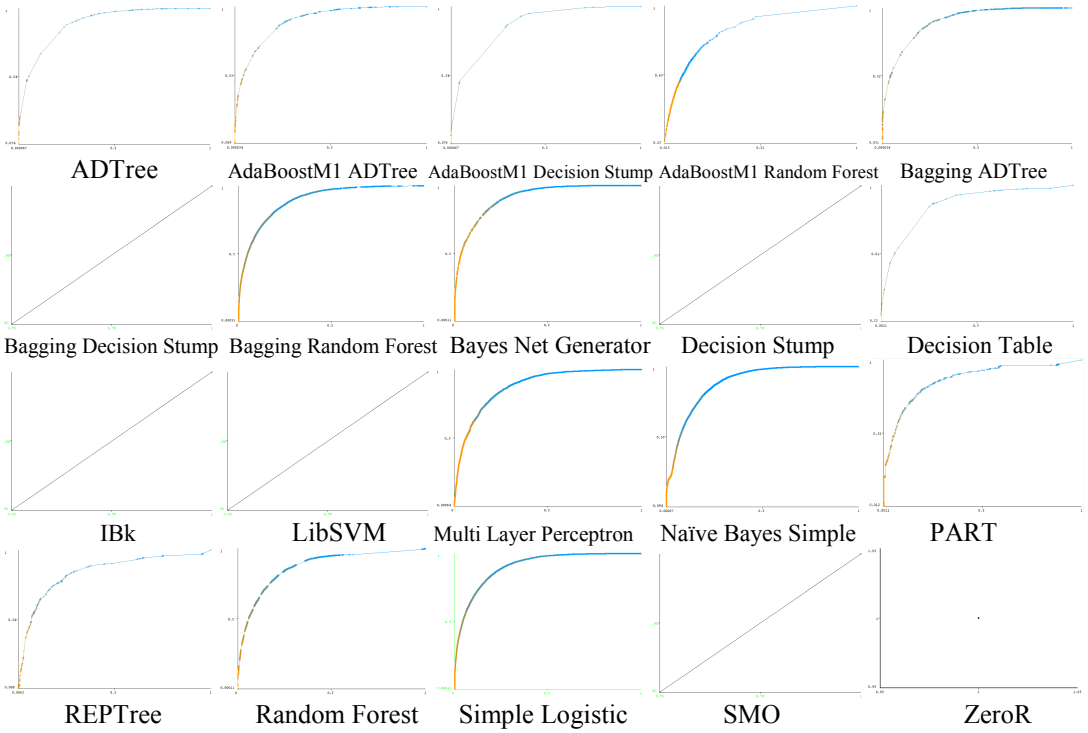

Figure 7.10: ROC graphs of twenty algorithms for Adult problem (X Axis-False Positive Rate, Y-Axis True Positive Rate).

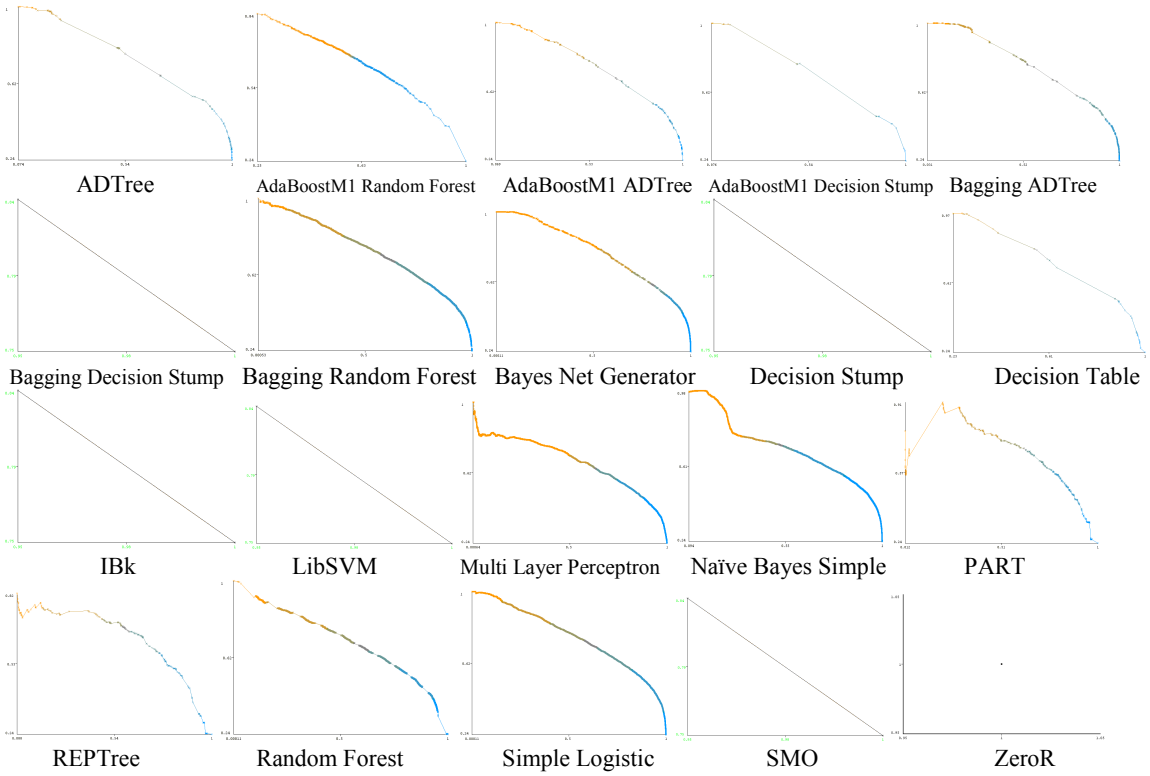

Figure 7.11: Precision/Recall graphs of twenty algorithms for Adult problem (X Axis-Recall, Y Axis-Precision).

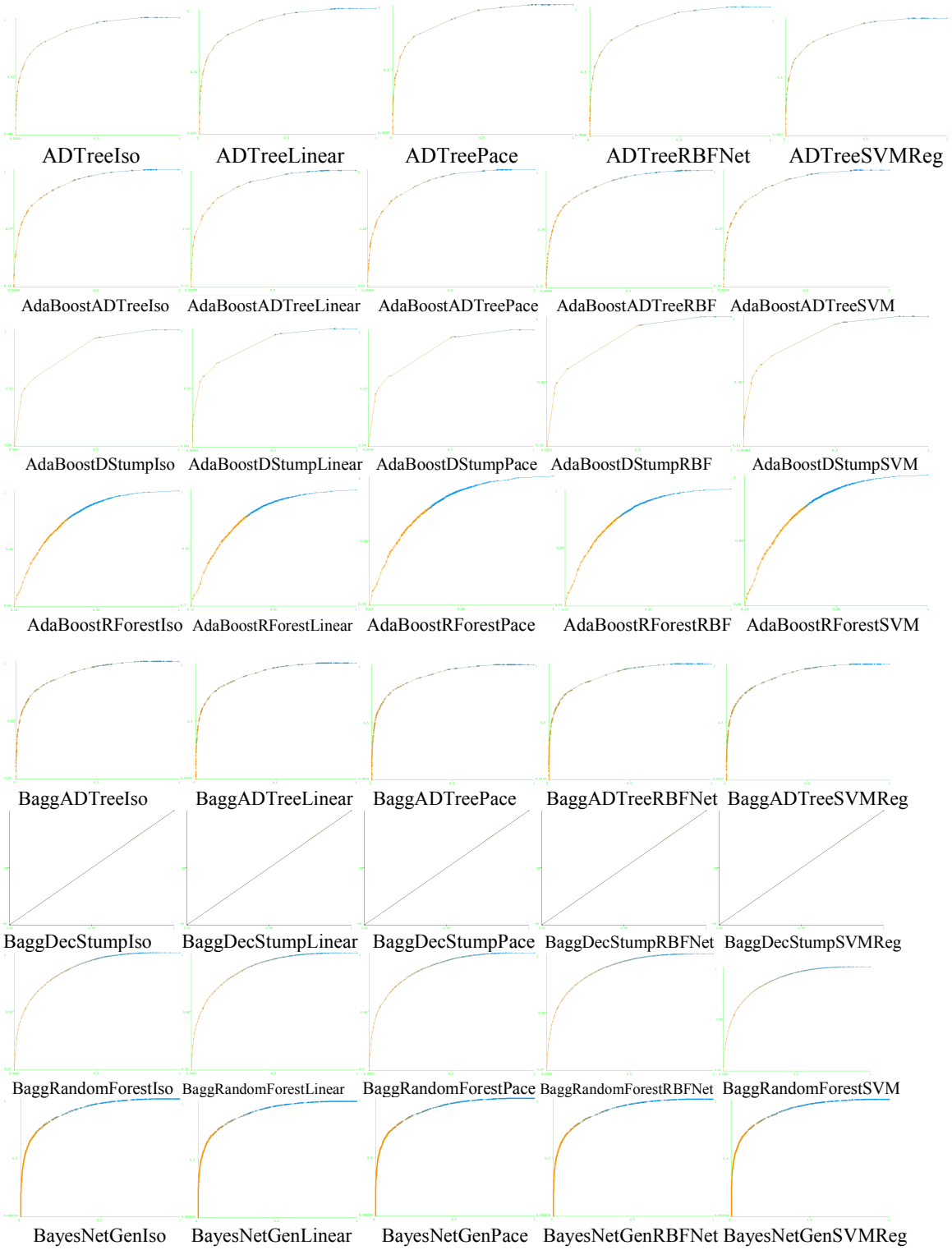

Figure 7.12a: ROC graphs of twenty algorithms from all five fuzzy processes for Adult problem (X Axis-False Positive Rate, Y-Axis True Positive Rate).

Figure 7.12b: ROC graphs of twenty algorithms from all five fuzzy processes for Adult problem (X Axis-False Positive Rate, Y-Axis True Positive Rate).

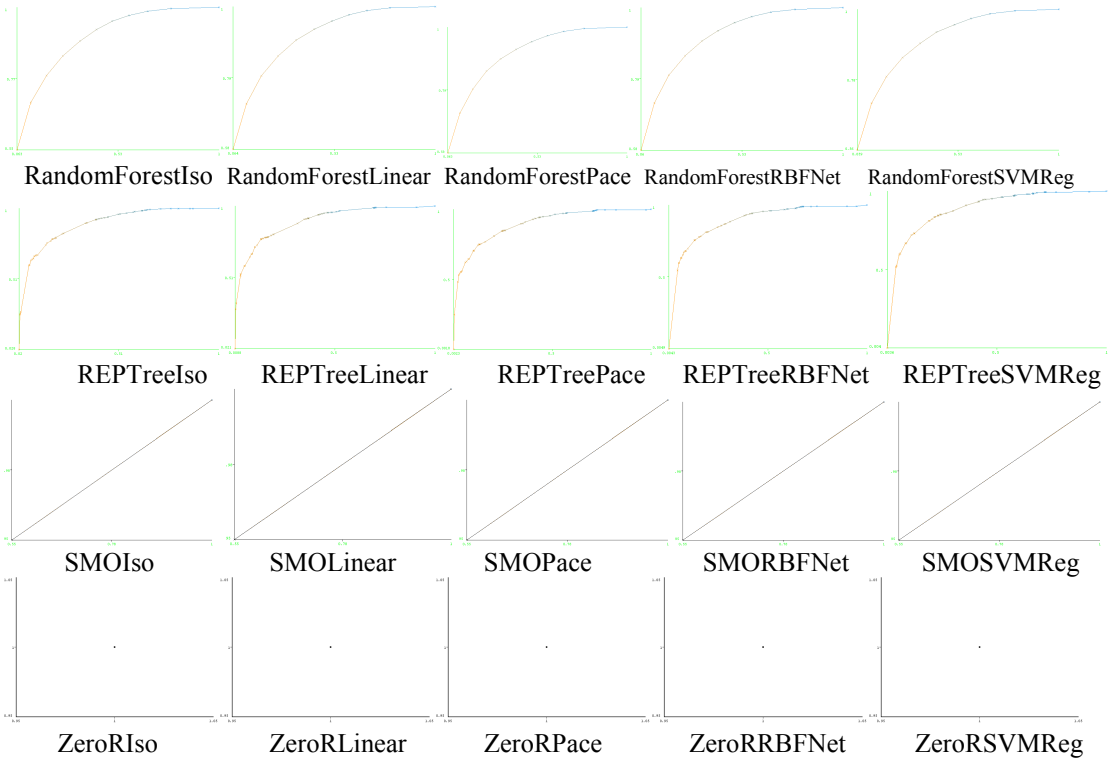

Figure 7.12c: ROC graphs of twenty algorithms from all five fuzzy processes for Adult problem (X Axis-False Positive Rate, Y-Axis True Positive Rate).

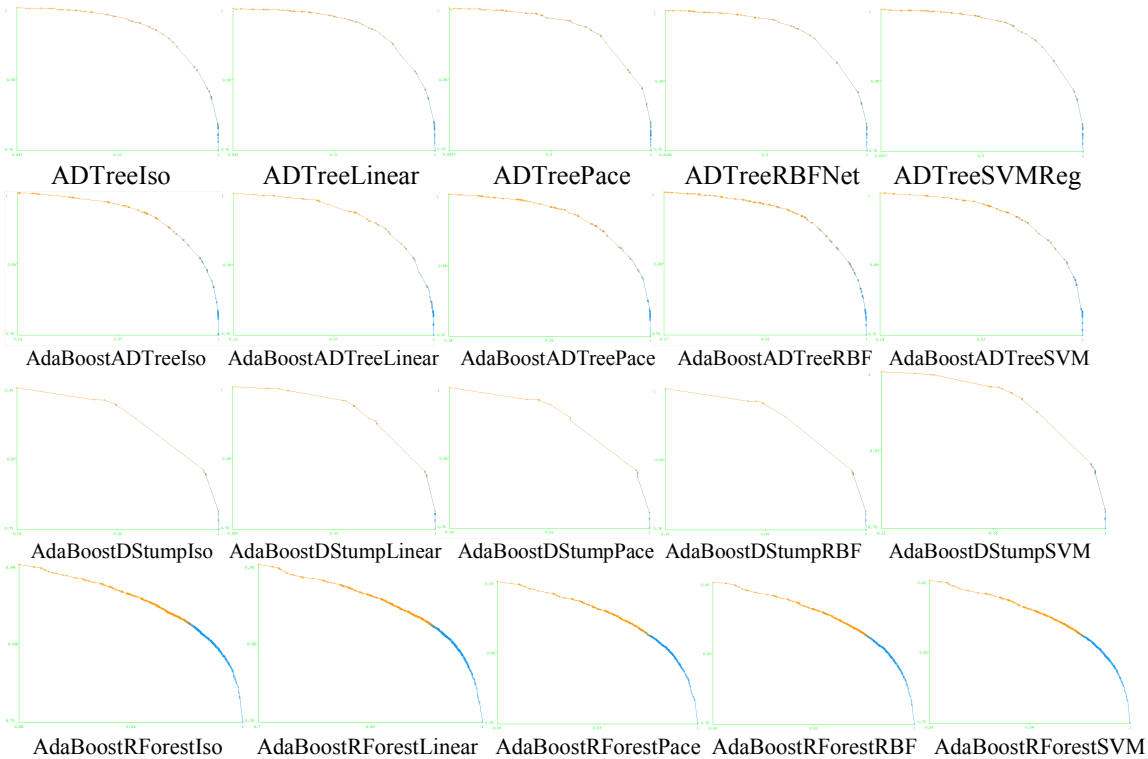

Figure 7.13a: Precision/Recall graphs of twenty algorithms from all five fuzzy processes for Adult problem (X Axis-Recall, Y Axis-Precision).

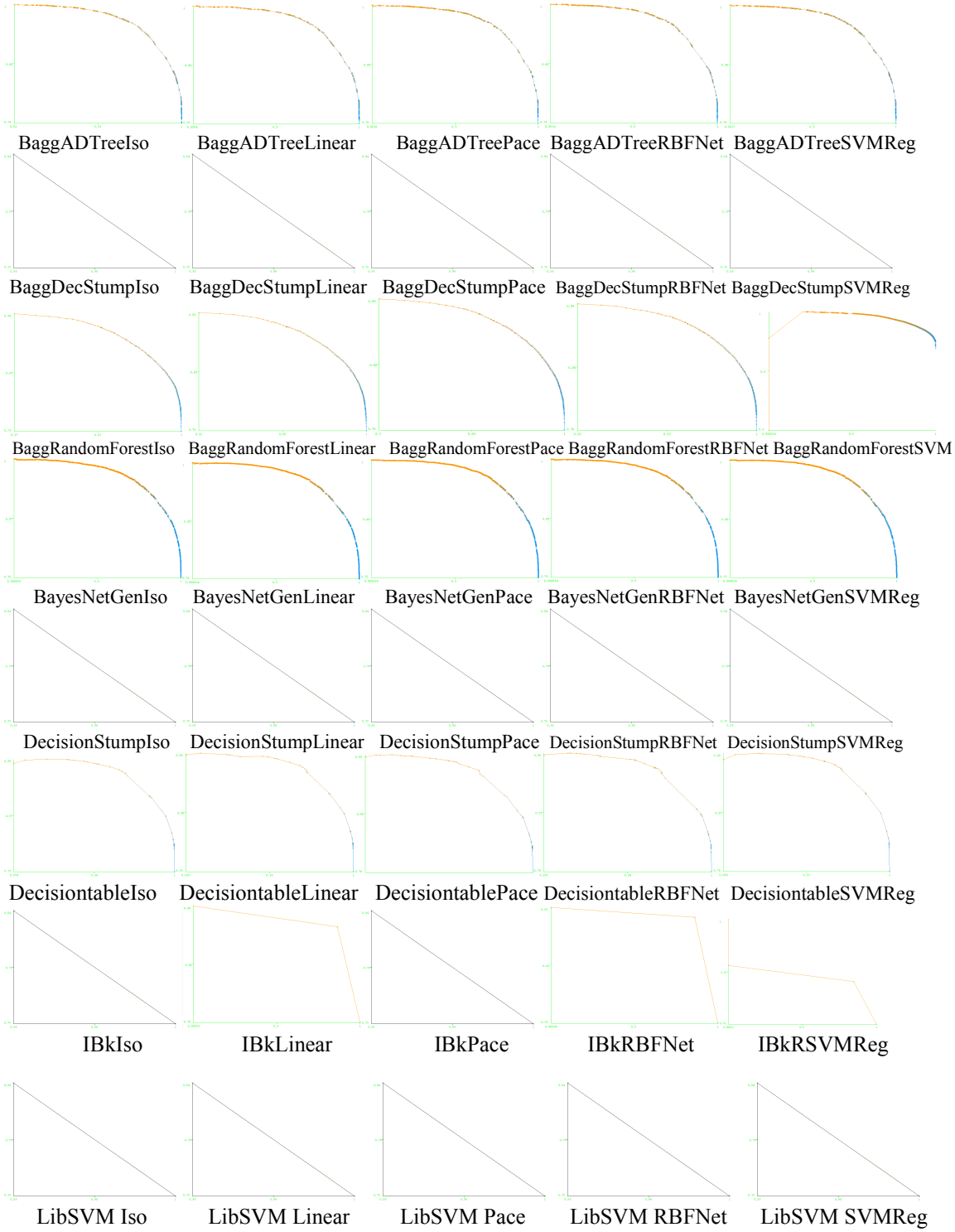

Figure 7.13b: Precision/Recall graphs of twenty algorithms from all five fuzzy processes for Adult problem (X Axis-Recall, Y Axis-Precision).

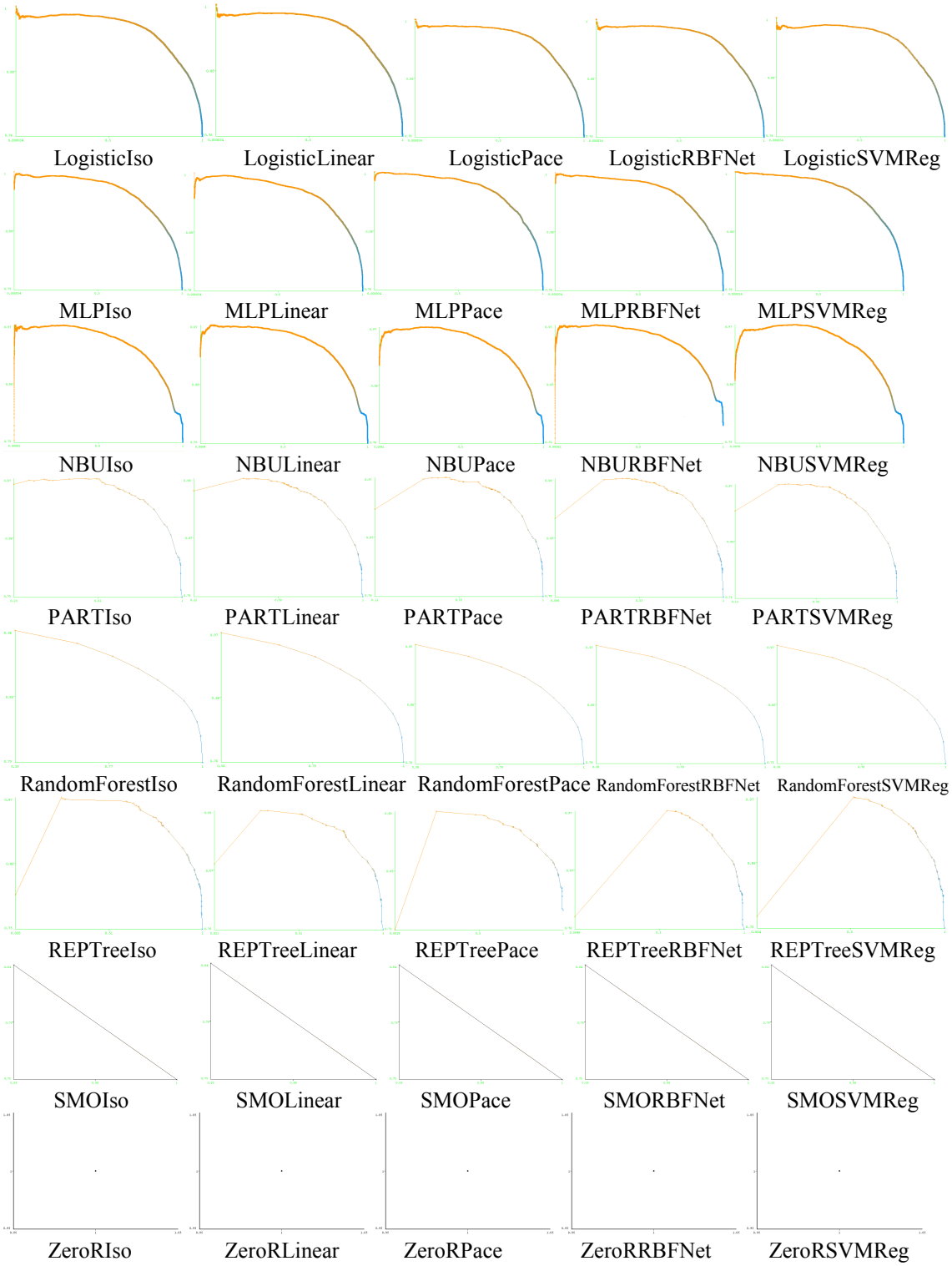

Figure 7.13c: Precision/Recall graphs of twenty algorithms from all five fuzzy processes for Adult problem (X Axis-Recall, Y Axis-Precision).

7.2 Hypothesis Testing for 'Genetic Information System Development and Maintenance' Model

Teams of researchers and software developers were involved in the process of software development for the organizations discussed in Chapter 5. 'Genetic Information System Development and Maintenance' model was tried and the results were tested against data collected from earlier developments for the same organizations and actual data of the development process and results of Case Study 1 discussed in section 6.2 of section 6 are presented here, similar results are observed for other case studies discussed in same section.

Research methodology for proposed research work includes implementation of proposed model in diversified business organizations, Collection of data from actual implementation, Collection of dataset from standard organizations, Comparison of results, Hypothesis testing and concluding with results etc.

For testing parameters like reliable maintenance of information system and total system costs are of major concern. Proposed model is highly adaptive due to adoption of Internet based technologies and team involved on continuous basis for system development work for the perfection of the system. Hypothesis testing is thus performed against the parameters related to reliable corrective maintenance and costs involved. Data is collected from the organizations' offices scattered worldwide, whereas criteria for collection of data was based on Type of Business organization, System development/Maintenance Costs, Size of System, Environment of the system, Scale of operations for system etc. and these factors were also normalized according to the characteristics of the system under study.

7.2.1 Reliability testing through numbers of faults received after initial implementation of information systems

Reliability of a system depends on the number of faults received and action taken within same week as faults received. First of all, hypothesis related to number of faults received after initial implementation of system is tested. Null and alternate hypothesis for reliability

testing in terms of number of faults received after initial implementation of system, are as follow:

H_0 = There is no significant difference between reliability trends of proposed model and other earlier models in terms of number of faults received and corrected after new system is implemented.

H_1 = There is a significant difference between reliability trends of proposed and other earlier models.

Significant level is kept 95% for the acceptance of Null hypothesis. On the basis on data collected from earlier implementations and data collected from proposed model's implementation, following graphical comparisons are produced. Figure 7.14 include the difference between the faults received for benchmark system and system under study.

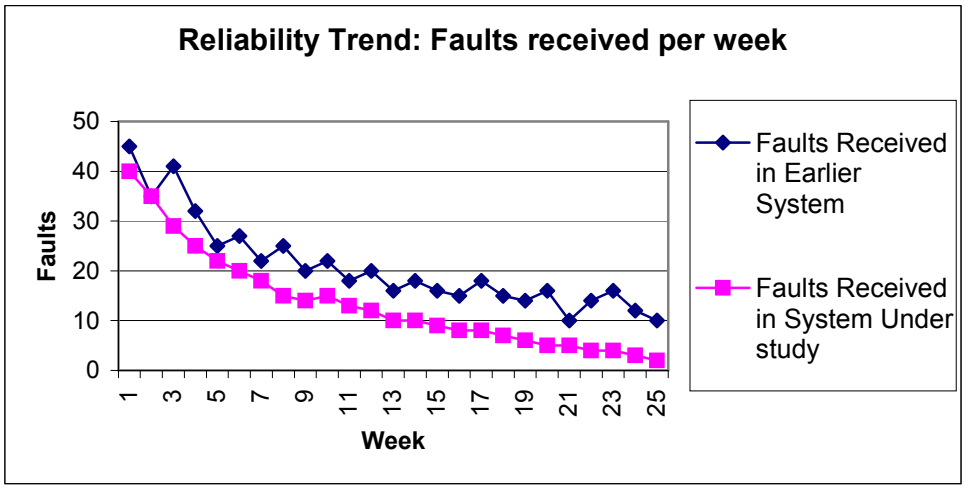

Figure 7.14: Reliability Trend: Faults received per week after initial implementation

Continuous decline in number of faults is recorded from proposed methodology as compared to the earlier system. Above data is applied to test the hypothesis, t-tests are performed and results show the rejection of null hypothesis implying the acceptance of alternative hypothesis i.e. there is a significant difference between faults received and corrected from current system and earlier implementations.

7.2.2 Reliable corrective maintenance testing through action taken within same week against the faults received after initial implementation of information system.

Reported faults need to be corrected as soon as possible. Faults received and action taken to correct them within same weeks as they received is the criteria used for testing reliable corrective maintenance. Null and alternate hypothesis for reliability testing are as follow:

H_0 = There is no significant difference between reliability trends of proposed model and other earlier models in terms of action taken against faults received after new system implemented.

H_1 = There is a significant difference between reliability trends of proposed and other earlier models.

Significant level is kept 95% for the acceptance of Null hypothesis. Figure 7.15 include the comparison of action taken against the reported faults within same week for proposed model.

Figure 7.16 illustrates the comparison of percentage of faults corrected within the same week as fault(s) reported for earlier system development and for proposed model implementation.

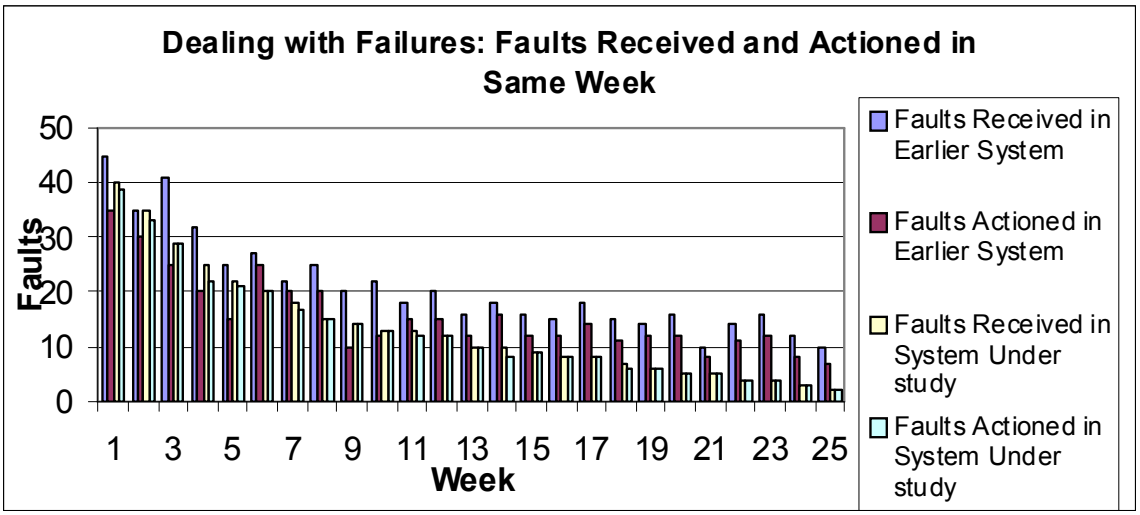

Figure 7.15: Dealing with failures: Faults Received and Action taken in Same Week.

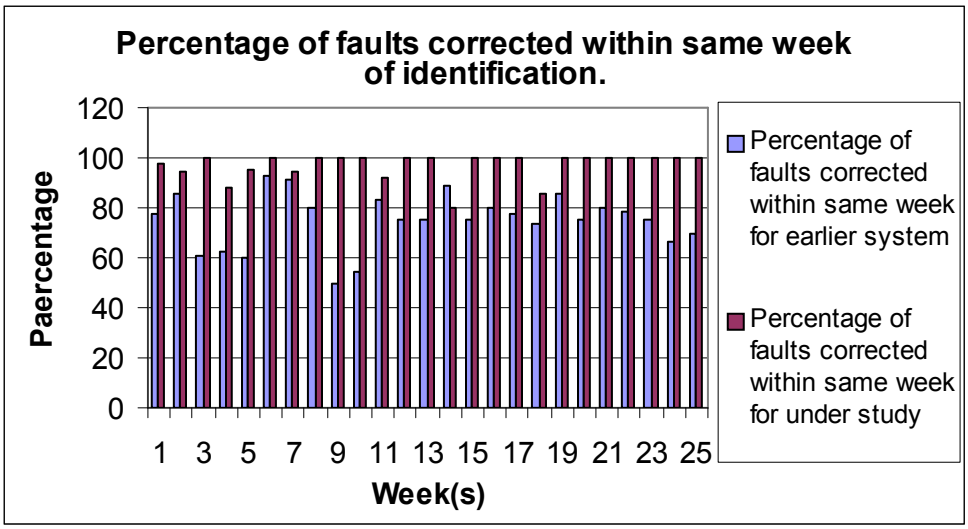

Figure 7.16: Percentage of faults corrected within same week of identification.

Above data is applied to test the hypothesis, t-test for testing hypothesis and results indicate the rejection of null hypothesis implying the acceptance of alternative hypothesis and indicating a significant improvement in taking action against faults received.

7.2.3 Testing the difference in cumulative costs for the systems based on other models and proposed model.

'Genetic Information System Development and Maintenance' model involves a team of company employees to look after the system continuously contributing to higher system development costs, but maintenance costs for such system development will be minimized or will hide behind development cost. So, cumulative costs (including system development and maintenance costs including effort involved) for the system will be very similar to the system developed and maintained through other models. Null and alternate hypothesis for testing the differences between cumulative development and maintenance costs for earlier models and proposed model are defined as follow:

H_0 = There is no significant difference between cumulative development and maintenance costs of proposed model and other earlier models.

H_1 = There is a significant difference between cumulative development and maintenance costs of proposed and other earlier models.

Significant level is kept 95% for the acceptance of Null hypothesis. On the basis collection of standard and actual datasets following graphical comparisons are produced.

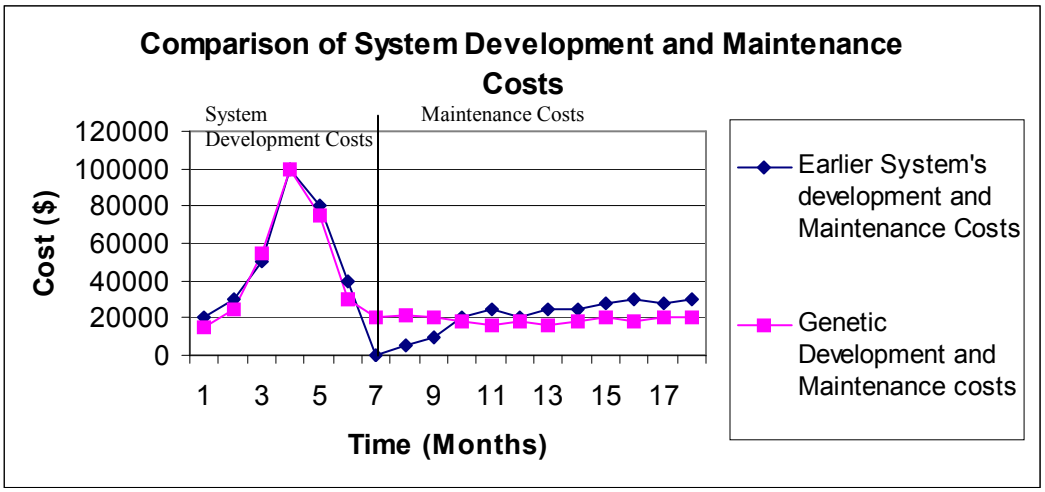

Figure 7.17: Comparison of System Development and Maintenance Costs

.

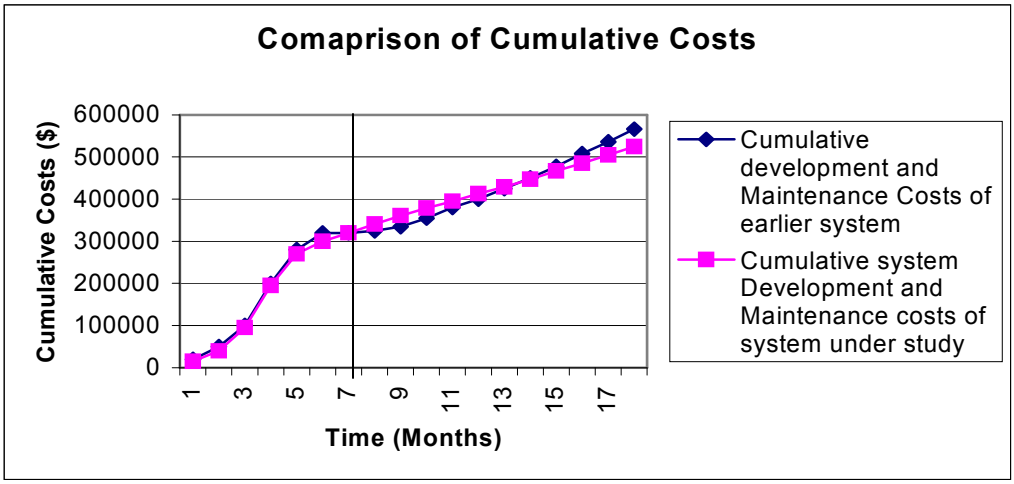

Figure 7.18: Comparison of Cumulative Costs

Figures 7.17 and 7.18 illustrate the comparison of the systems based on costs involved. From comparison, we find that with time, maintenance costs for other benchmark system increases very rapidly, whereas it remains under control for proposed model. Cumulative costs for system development and maintenance are almost same for a newly developed system, followed by decline in costs for other models and then followed by tremendous

increase in total costs of the system development and maintenance for benchmark system, whereas for proposed model cumulative costs are lesser in long run. Above data is applied to test the hypothesis through t-test and results indicate the acceptance of null hypothesis implying the rejection of alternative hypothesis, which implies that there is no significant difference between the total costs of the earlier models and proposed model.

7.2.4 Comparative analysis to check the effect of proposed model over software reuse

Effect of proposed model on software's reuse is studied and compared, based on the Lines of Code (LOC) reused during different revisions of software. The de-facto standard for measuring the reuse level (Poulin [159]), is the percentage of software (LOC) reused as compared to the Total size of software (LOC), i.e.

$$\frac{\text{Reused Software}}{\text{Total Software}} X 100 \qquad \ldots\ldots (7.1)$$

Software reuse is compared for system under study with earlier used system in terms of percentage of lines of codes reused during each revision. Figure 7.19 displays the percentage of software reused during each revision.

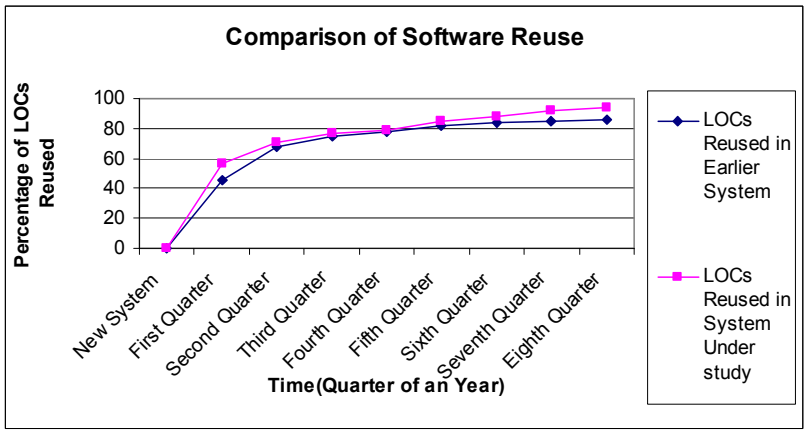

Figure 7.19: Comparison of Software Reused in Percentage of Lines of Code reused.

Analysis of this comparison shows improvement in software reuse for the system developed through proposed model.

7.2.5 Generalized comparison of proposed model with other models

Comparative study of different models based on general observations and expert opinion about different models, is summarized in Table 7.11, it contains the Comparison of models based on costs, reliability, maintainability, adaptability, portability and reusability.

Model(s) / System Factors	Water Fall Model	Spiral Model	Iterative Model	Proto-typing model	Component Based Development	Genetic Development and Maintenance
Development cost	Moderate	Moderate	High	High	High	High
Maintenance cost	High	High	Moderate	Moderate	Moderate	Low
Reliability	Low	Moderate	High	Moderate	Moderate	High
Maintainability	Low	Moderate	Moderate	Moderate	Moderate	High
Adaptability	Low	Moderate	Moderate	High	High	High
Portability	Low	Moderate	Moderate	High	High	High
Reusability	Low	Moderate	Moderate	Moderate	High	High

Table 7.11: Comparison of different models on the basis with cost, reliability, maintainability and adaptability of information systems

There are many other aspects like user satisfaction, goal orientation, human and organizational factors where the proposed model is observed to be better than any other implementation.

7.3 Conclusion

In this chapter, experimentation results of different supervised learning processes and hypothesized results of proposed 'Genetic Information System Development and Maintenance' model are presented. Massive experimentation work has been performed for evaluating and comparing previously known supervised learning processes as well as for the proposed process. Experimentation work is divided into two sub categories viz. Major Study and Minor Study. Major study concentrated its evaluation over three previously known processes (i.e. Fixed Split, Cross Validation and Platt Scaling) and proposed process through five regression algorithms (i.e. Isotonic Regression, Linear Regression, Pace Regression, RBF Network and SVM Regression). Major study included twenty classification algorithms, five regression algorithms and ten datasets for evaluating eight processes. Minor study needed to be separated from minor study as three post processing processes Additive Regression, LogitBoost and Isotonic Regression (through Stacking) require class attribute of datasets to be numeric, whereas most of the classification algorithms demand class attribute to be nominal. After analyzing different factors, five classification algorithms are included for experimentation through eleven processes.

Three new processes are added for experimentation in minor study. Output of this massive experimentation is presented through tables and graphs. Results of major study indicate that proposed process consistently performs well over all five regression algorithms, but RBF Network algorithm pruned the train set very marginally, whereas other algorithms pruned trained data very well. SVM Regression algorithm performs better than rest of the regression algorithms. Through experimentation it was shown that size of model generated from classification is directly related to the size of the train set and complexity of model increases with increasing size of the model. Preprocessing method Cross Validation reduces the size of the train set by ten percent (in case of ten fold cross validation) to 90%, whereas proposed process reduces this size to a range from 10.03% to 95.93%. Overall average of the train set used for proposed study was around 86%. Here 95.93% size of the train set was for ZeroR algorithm, which is identified by proposed process to be non learning algorithm and is advised not be used for any classification purposes. If we avoid algorithms and processes having the train set more than overall average, then this figure can be reduced to 76.56%. Results of major study have identified Random Forest algorithm to be best algorithm among all twenty algorithms used for study, whereas ZeroR algorithm is

identified to be poorest performer, when all outcomes were compared in terms of accuracy. Results were also analyzed on the basis of other metrics. Results of other metrics identified PART algorithm to be performing better than other algorithms individually, followed by ADTree. Random Forest algorithm performed consistently well in terms of other metrics over individual, bagged and boosted way. Results of minor study have also indicated reasonably good behavior of proposed process. Out of other added processes in minor study, LogitBoost has shown better performance in many respects, but its limitation is due to its demand for numeric class and most of classification algorithms demand nominal classes, it restricts the use of LogitBoost method. Proposed process enjoys the advantage to be used with all types of algorithms, only data coding may be required at most i.e. conversion of nominal attributes into numeric attributes through coding, without affecting the processing of classification algorithm. Results of proposed process from minor study are also being compared with results from stacking of same regression algorithms with same classification algorithms. Results of proposed process are found to be far better than results from Stacking. Results of proposed process over all regression algorithms are also presented through ROC curve and Precision/Recall graphs.

Hypothesized results from case study used for investigations of 'Genetic Information System Development and Maintenance' model are also presented in this chapter. Hypothesis were proposed for (i) significant difference between reliability trends of proposed model and other earlier models in terms of number of faults received and corrected after new system is implemented (ii) significant difference between reliability trends of proposed model and other earlier models in terms of action taken against faults received after new system implemented (iii) significant difference between cumulative development and maintenance costs of proposed model and other earlier models. Results from first two hypotheses indicate that proposed model significantly reduces the number of faults received and significantly contributes improves in the correction of faults. Result of third hypothesis indicates that there is no significant difference between cumulative development and maintenance costs of proposed model and other earlier models. Hypothesized results show that proposed model can significantly improve corrective and perfective system maintenance without extra expenditures.

Chapter 8

Conclusions and Future Directions

This chapter summarizes the contents of this thesis and discusses about the limitations as well as contributions of present research. This chapter also highlights the vision for future research work.

8.1 Summary of Present Research

In this thesis, problems related to supervised learning for data mining and system development for knowledge based systems have been analyzed and an effort have been made to find solutions for the problems related to earlier processes and models for supervised learning and knowledge based system development respectively.

Initially this research work concentrated upon investigating past developments in the areas of data mining, supervised learning as well as software and knowledge engineering to find the past developments through literature survey. Then this research work concentrated upon enumerating different processing methods used for supervised learning. These different processes are devised through a general Knowledge Discovery in Databases (KDD) process. From this general process, six specific processes were identified to be

widely practiced worldwide. After investigating different processes for supervised learning, limitations of these methods were identified. After identifying problems related to different supervised learning processes, investigations were directed towards finding the solutions for these problems. Search for finding solutions of these problems motivated to develop a new process for supervised learning. This motivation resulted in the form of Fuzzy Boundaries of Regression Based Clusters process.

Proposed process is based on parallel processing of classification as well as regression algorithms. Proposed process allows pruning the training data without compromising the performance of outcome. It reduces the size of the train set significantly, while analyzing each record of data qualitatively as well as quantitatively through classification and regression algorithms respectively. Proposed process as well as previously known processes were applied on twenty classification algorithms and five regression algorithms over ten datasets gathered from internationally renowned organizations. After investigating six previously used processes and a new proposed process for supervised learning, question arose about suitability of previously known system development models for knowledge based system that involves the processing of two or more of these processes simultaneously. All previously known models were investigated for this purpose, but none of the system development model was recognized to be satisfactory for such a dynamic knowledge based system development.

This necessity to find satisfactory model led to development of model for developing and maintaining dynamic knowledge based systems. This led to the construction of ‘Genetic Information System Development and Maintenance’ model. This model postulated the need of creating and maintaining a team of software developers within the organization for whom system is to be developed. Further this model advocated the need of system development and maintenance activities to run simultaneously throughout the life of an organization. Different advantages of proposed model were enumerated. Proposed process for supervised learning and model for knowledge based system was needed to be investigated through quantitative evaluation. So, a case study was included to hypothesize and prove different aspects of this model.

Massive experimentation work was performed for evaluating and comparing previously known supervised learning processes as well as proposed process. Experimentation setup

was divided into two sub categories viz. Major Study and Minor Study. Major study concentrated its evaluation over three previously known processes (i.e. Fixed Split, Cross Validation and Platt Scaling) and proposed process through five regression algorithms (i.e. Isotonic Regression, Linear Regression, Pace Regression, RBF Network and SVM Regression).

Major study included twenty classification algorithms, five regression algorithms and ten datasets for evaluating eight processes. Minor study needed to be separated from minor study as three processes Additive Regression, LogitBoost and Isotonic Regression through Stacking require class attribute of datasets to be numeric, whereas most of the classification algorithms demand class attribute to be nominal. After analyzing different factors, five classification algorithms were included for experimentation through eleven processes. Three new processes were added for experimentation in minor study. Output of this massive experimentation was presented through tables and graphs. Results of major study indicate that proposed process consistently performs well over all five regression algorithms. Out of which RBF Network algorithm pruned the train set very marginally, whereas other algorithms pruned trained data very well. SVM Regression algorithm performs better than rest of the regression algorithms.

Through experimentation it was shown that size of model generated from classification is directly related to the size of the train set and the complexity of model increases with increasing size of the model. Cross Validation method reduces the size of the train set by 10% (in case of ten fold cross validation), whereas proposed process reduces this size up to 90% having the train set size ranging from 10.03% to 95.93% of the original size. The overall average of the train set size for proposed model was reduced to 86%. Here 95.93% size of the train set was for ZeroR algorithm, which is identified by proposed process to be non learning algorithm and is advised not be used for any classification purposes. If algorithms and processes having the train set more than overall average avoided, then this figure reduced to 76.56%.

Results of major study have identified Random Forest algorithm to be the best algorithm among all twenty algorithms used for study, whereas ZeroR algorithm is identified to be poorest performer, when all outcomes were compared in terms of accuracy. Results were also analyzed on the basis of other metrics. Results of other metrics identified PART

algorithm to be performing better than other algorithms individually, followed by ADTree. Random Forest algorithm performed consistently well in terms of other metrics over individual, bagged and boosted way. Results of minor study have also indicated reasonably good behavior of proposed process. Out of other added processes in minor study, LogitBoost had shown better performance in many respects, but its limitation is due to its applicability for numeric class problems, whereas most of the classification algorithms demand nominal classes, which restricts the use of LogitBoost method. The proposed process enjoys the advantage that it may be used with all types of algorithms. Preprocessing of data required for proposed process may be data coding at most i.e. conversion of nominal attributes into numeric attributes through coding for regression algorithms without affecting the processing of classification algorithms. Results of proposed process from minor study were also being compared with results from stacking of same regression algorithms with classification algorithms. Results of proposed process were found to be far better than results from Stacking. Results of proposed process over all regression algorithms were also presented through ROC curve and Precision/Recall graphs.

'Genetic Information System Development and Maintenance' model was evaluated through a case study of Pepsi Foods Ltd., India. Proposed model was implemented in this organization and data was gathered through this implementation. Gathered data was compared with data collected from earlier system development of same organization. Hypothesized results indicated that significant corrective and perfective maintenance can be achieved without any extra expenditure through the use of proposed model.

8.2 Contributions of Present Research

This section is a short summary of the contributions of present research. The major contributions of this thesis are summarized as follows:

(i) Present research work investigated data mining, supervised learning as well as software and knowledge engineering thoroughly and presented these developments in chronological order. This literature review will work as a reference guide for future researches.

(ii) Present research investigated for finding different supervised learning processes practiced worldwide. Enumeration of these processes will help future researchers to find a collection of these processes at one place that is in this thesis and to speed up their future research developments.

(iii) Present research identified potential problems related to previously known supervised processes and presented a new process for supervised learning named as Fuzzy Boundaries of Regression Based Clusters. Proposed process involved classification algorithms, regression algorithms, fuzzy sets theory and clustering to perform supervised learning for classification purposes.

(iv) Present research contributed a model for knowledge based system development named 'Genetic Information System Development and Maintenance' model. This model advocated for organization's own team building for system development and maintenance purposes. This model defined the way to tackle the dynamism involved in development and maintenance of knowledge based systems.

(v) For present research massive experimentation was performed for supervised learning and this massive experimentation turned present research work to be one of the biggest researches ever being conducted worldwide in data mining area till date. Experimental setup and results of present research will guide future researchers for finding right way of experimentation and for comparing their results with results of present research.

(vi) Present research evaluated different supervised learning processes over a wide range of metrics. Different metrics are considered preferable for different industrial applications. Experimental results of present research will help industrial application for finding right kind of supervised learning process as well as comparing performance of their organizational outputs with outcomes of present research.

(vii) Present research work introduced a case study for evaluating 'Genetic Information System Development and Maintenance' model and presented hypothesized results. These hypothesized results will inspire future researches to evaluate future researches in similar ways.

Above listed major contributions of present research are also supported by many small but easily identifiable contributions like dataset searching, identifying different metrics for evaluation etc. In overall, present research will support many future researches in many ways mentioned above and in many other unmentioned ways.

8.3 Limitations of Present Research

There are certain issues related to new proposed process for supervised learning as well as for new proposed model for knowledge based system development and maintenance. The following issues have been identified as problems that should be addressed in future researches:

(i) Fuzzy boundaries of regression based clusters process was applied over problems related to binary classes. Further investigations are needed to check its applicability for multi class problems (i.e. problems having three or more classes).

(ii) Fuzzy boundaries of regression based clusters process involves preprocessing of data. Effect of post processing of data is needed to be analyzed for finding hybrid approaches for the pruning of supervised learning.

(iii) Fuzzy boundaries of regression based clusters process involves clustering of the output from classification as well as regression algorithms. Boundaries of such clusters are supposed to be found at extreme regions of binary classes. There is a need for finding possibly improved clustering algorithms.

(iv) Fuzzy boundaries of regression based clusters process involves parallel processing through regression based algorithms. Five regression algorithms were being investigated, more regression based algorithms need to be investigated for possible improvement of the results generated from the proposed process.

(v) A large number of classification algorithms have been investigated for Fuzzy boundaries of regression based clusters process, more algorithms may be investigated through this proposed process.

(vi) Ten problems/datasets were investigated in this thesis, this Fuzzy boundaries of regression based clusters process can be applied on other problems from different domains for pruned knowledge extraction.

(vii) 'Genetic Information System Development and Maintenance' model was investigated for cost, corrective and perfective maintenance in present thesis, further investigations need to be performed for investigating other metrics more closely.

(viii) 'Genetic Information System Development and Maintenance' model advocated for developing team of system developers from organizational employees. There may be certain behavioral aspects like organizational politics, team management etc. that need to be investigated and analyzed more closely.

8.4 Future Research Work

In this thesis, a new supervised learning process and a new system development model were presented. Future research work will necessarily extend proposed process and model for further improvements in similar directions as explored in this thesis as well as in new unexplored directions. Future research work with current vision may be enlisted as follows:

(i) Fuzzy boundaries of regression based clusters process in this thesis was explored for binary problems, future researches need to explore this process for multi class problems having number of classes more than two.

(ii) Fuzzy boundaries of regression based clusters process explained a way for finding clusters for outputs of classification as well as regression algorithms. Future researches need to find new algorithms for finding possibly improved clusters.

(iii) Ten problems were used to examine different processes for supervised learning algorithms. These datasets were collected from different internationally renowned organizations and from a variety of areas. As data mining is now-a-days used in many more areas, so future researches will explore many new areas.

(iv) As present research concentrate towards finding pre processing based process for supervised learning. This pre processing based process has left cushion for creating hybrid approaches through merging of current approach with post processing approaches. Future research work will concentrate upon finding such hybrid approaches through existing and new methods.

(v) Future researches need to investigate more classification as well as regression algorithm for investigating different processes of supervised learning algorithms.

(vi) Most of the processing for Fuzzy boundaries of regression based clusters process was being done in phases separately in required sequence. There is a need to develop such software that may integrate all the phases of proposed process and can process all phases in coordinated way.

(vii) 'Genetic Information System Development and Maintenance' model need to be investigated for many more metrics than used for investigations in present research. Future researches will target to enhance the popularity of proposed model by applying it in many more organizations as well as by analyzing the results for many more metrics.

(viii) There is a need to find the impact of behavioral aspects by forming a team of system developers within the organization for which system is to be developed as advocated by 'Genetic Information System Development and Maintenance' model. Management researches need to be conducted for analyzing such behavioral aspects and to analyze the effect over outcome of proposed model. Future researches will be inspired to include behavioral parameters for finding their impact on performance metrics.

(ix) 'Genetic Information System Development and Maintenance' model will be also be popularized for enterprise wide global application development of the organizations and its suitability and effect will be analyzed in future researches.

8.5 Conclusion

There is general process for knowledge discovery in databases, but many specific alterations are used globally. This research work identified six such processes and critical analyzed them. These investigations generated many questions about previously practiced processes for supervised learning. Investigations of this research work were guided towards finding the solutions to the problems generated by these questions. Present research proposed a process for supervised learning called Fuzzy boundaries of regression based clusters process for the solutions of these problems. Proposed process was experimented for twenty classification and five regression based algorithms over ten datasets/problems.

Proposed process involves parallel processing of classification and regression algorithms for pruning of training data. Experimental results confirmed the better performance of algorithms with very large reduction in the size of the train set. Proposed process consistently performed well for all classification and regression algorithms used for study. Random Forest algorithm was confirmed by the results to be best classification algorithm among all classification algorithms used for study. ZeroR algorithm was identified to be poorest performer and experimentation experiences strengthened the view about discarding ZeroR algorithm due to its non learning behavior. Detailed comparison of results over different previously known as well as proposed process was included in this thesis.

Selection of any process for knowledge extraction depends on kind of problem at hand. It may require trying many processes simultaneously. But, use of such a large number of processes again raised questions about system development model that must be supportive for managing the complexity of such a dynamic knowledge based system development. Present research work proposed a model for developing and maintaining such dynamic systems. ‘Genetic Information System Development and Maintenance’ model was presented in this thesis for developing and maintaining highly dynamic systems. Proposed model advocated system developer’s team building from own organization’s employees for which information system is to be developed. Proposed model also advocated for merging system development and maintenance efforts, so that suitable system development and maintenance can be achieved throughout the life of an organization. Proposed model was investigated through a case study. Hypothesized results showed that proposed model can

significantly improve corrective and perfective system maintenance without extra expenditures.

In summary, this thesis has made significant contributions to the fields of data mining as well as to software and knowledge engineering. Proposed process for supervised learning and model for information system development model will become milestones in coming times. Massive results generated and presented in this thesis have not only placed this research into one of the biggest researches, but also its results will be used as reference guide for future researches in many forthcoming years. Researches will be continued in finding solutions to the problems raised in current thesis and for the problems that exist in different computing domains or are arising with time. It is hoped that contributions of present research and future researches will contribute to society in long run by improved knowledge learning and system development.

References

1. Aggarwal Rakesh, Ghosh Sakti, Imielinski Tomasz, Iyer Bala and Swami Arun “An interval classifier for database mining applications”, VLDB, 1992.

2. Agrawal R. and Psaila Giuseppe, “Active Data Mining”, KDD, pp. 3-8, 1995.

3. Agrawal Rakesh, Imielinski Tomasz and Swami Arun N., “Mining Association Rules between Sets of Items in Large Databases”, SIGMOD Conference, pp. 207-216, 1993.

4. Agrawal Rakesh, Imielinski Tomasz and Swami Arun, “Database Mining: A Performance Perspective”, IEEE Transactions on Knowledge and Data Engineering, Special issue on Learning and Discovery in Knowledge-Based Databases, Vol. 5, No. 6, pp. 914-925, Dec 1993.

5. Agrawal Rakesh, Srikant Ramakrishnan, “Fast Algorithms for Mining Association Rules in Large Databases”, VLDB, pp. 487-499, 1994.

6. Aha D. and Kibler D., “Instance-based learning algorithms”, Machine Learning, Vol. 6, pp. 37-66, 1991.

7. Aik Choon Tan, David Gilbert and Yves Deville, “Multi-Class Protein Fold Classification Using a New Ensemble Machine Learning Approach”, Genome Informatics, Vol. 14, pp. 206–217, 2003.

8. Alavi M., “An Assessment of the Prototyping Approach to Information Systems”, Communications of the ACM, 27, Vol. 6, pp. 556-564, 1984.

9. Aoyama M., “Component-Based Software Engineering: Can it Change the Way of Software Development?”, Proceedings Volume II of the 1998 International Conference on Software Engineering, Apr 1998.

10. Armenise, P., Bandinelli, S., Ghezzi, C. and Morzenti, A., “A survey and assessment of software process representation formalisms”, International Journal of Software Engineering and Knowledge Engineering, Vol. 3, pp. 410-26, 1993.

11. Atlas L., Connor J. and Park D., “A performance comparison of trained multi-layer perceptrons and trained classification trees”, Systems, man and cybernetics: proceedings of the 1989 IEEE international conference, Cambridge, pp. 915–920, 1991.

12. Ayer M., Brunk H., Ewing G., Reid W. & Silverman E., “An empirical distribution function for sampling with incomplete information”. Annals of Mathematical Statistics, Vol. 5, pp. 641-647, 1955.

13. Bandinelli S., Fugetta A. and Grigoli S., “Process modelling in the large with SLANG”, Proceedings of the 2nd International Conference on Software Process, Berlin, pp. 75-93, 1993.

14. Basili V. R. and Turner A. J., “Iterative Enhancement: A Practical Technique for Software Development”, IEEE Transactions on Software Engineering 1, Vol. 4, pp. 390-396, 1975.

15. Bauer E. and Kohavi R., “An empirical comparison of voting classification algorithms: Bagging, boosting, and variants”, Machine Learning, Vol. 36, 1999.

16. Bayardo Roberto J. Jr. and Daniel Miranker P., “A Complexity Analysis of Space-Bounded Learning Algorithms for the Constraint Satisfaction Problem”, Proc. of the 13th Nat’l Conf. on Artificial Intelligence, pp. 298-304, 1996.

17. Berry C. C. “The kappa statistic”, Journal of the American Medical Association, Linguistics, COLING-90, Vol. 2, pp. 251-256, 1992.

18. Bersoff E., “Elements of software configuration management”, IEEE Trans. Software Engg., 1984.

19. Best M. J. and Chakravarti N., "Active set algorithms for isotonic regression; a unifying framework", Mathematical Programming, Vol. 47, pp. 425–439, 1990.

20. Bider I., Johannesson P. and Perjons E., “Goal-Oriented Patterns for Business Processes”, position paper for Workshop on Goal-Oriented Business Process Modeling GBPM’02, 2002.

21. Blake C. and Merz C., UCI repository of machine learning databases, 1998.

22. Bocca Jorge B., “Logic Programming Environments for Large Knowledge Bases”, A Practical Perspective Abstract. VLDB, pp. 563, 1991.

23. Boehm B. W., Gray T. E. and Seewaldt T., “Prototyping Versus Specifying: A Multiproject Experiment”, IEEE Transactions on Software Engineering, SE-10, Vol. 4, pp. 290 -402, 1984.

24. Boehm Barry W., “A spiral model of software development and enhancement”, IEEE Computer, pp. 61-72, 1988.

25. Brandtweiner R. and Scharl A., “An institutional approach to modeling the structure and functionality of brokered electronic markets”, International Journal of Electronic Commerce, Vol. 33, pp. 71-88, 1999.

26. Breiman Leo, “Random Forests”, Machine Learning, pp. 5-32, 2001.

27. Breiman L., Friedman J. H., Olshen R. A. and Stone C. J., “Classification and Regression Trees”, Wadsworth and Brooks, Monterey, Ca, 1984.

28. Bubenko J., Rolland C., Loucopoulos P. and De Antonellis V., “Facilitating ‘fuzzy to formal’ requirements modelling”, Proceedings of the IEEE 1st Conference on Requirements Engineering, ICRE’94, Colorado Springs, CO and Taipei, pp. 154-158, 1994.

29. Burrows C., George G. and Dart S., “Configuration Management”, Ovum Ltd., 1996.

30. Caruana Rich and Niculescu-Mizil Alexandru, “An Empirical Comparison of Supervised Learning Algorithms”, Proceedings of the 23 rd International Conference on Machine Learning, Pittsburgh, PA, 2006.

31. Chang Chih-Chung and Lin Chih-Jen, “LIBSVM - A Library for Support Vector Machines”, URL http://www.csie.ntu.edu.tw/cjlin/libsvm/, 2001.

32. Chen M. S., Han J. and Yu P. S., “Data Mining: An Overview from a Database Perspective”, IEEE Transactions on Knowledge and Data Engineering, pp. 866-883, 1996.

33. Cheung D., Han J., Ng V. and Wong C.Y., “Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique”, Proc. of 1996 Int'l Conf. on Data Engineering ICDE'96, New Orleans, Louisiana, USA, Feb 1996.

34. Cheung D., Han J., Ng V. T., Fu A. W. and Fu Y., “A Fast Distributed Algorithm for Mining Association Rules”, Proc. of 1996 Int'l Conf. on Parallel and Distributed Information Systems PDIS'96, Miami Beach, Florida, USA, Dec 1996.

35. Cheung D.W., Fu A. W.-C. and Han J., “Knowledge Discovery in Databases: A Rule-Based Attribute-Oriented Approach”, Int'l Symp. on Methodologies for Intelligent Systems ISMIS'94, Charlotte, North Carolina, pp. 164-173, Oct 1994.

36. Cheung David W., Ng Vincent T., Fu Ada W. and Fu Yongjian, “Efficient Mining of Association Rules in Distributed Databases”, IEEE Transactions on Knowledge and Data Engineering, Dec 1996.

37. Cheung Yin-Ling and Fu Ada Wai-Chee, “Mining Frequent Itemsets without Support Threshold: With and without Item Constraints”, IEEE Transactions On Knowledge And Data Engineering, Vol. 16, No. 6, Jun 2004.

38. Cho N. J., “Internet business in Korea: the state-of-art”, Management and Computer, pp. 188-220, 1999.

39. Conradi R. and Westfechtel B., “Towards a Uniform Version Model for Software Configuration Management”, Proceedings of the Seventh International Workshop on SoRware Configuration Management, New York, 1997.

40. Cooper G. F., Aliferis C. F., Ambrosino R., Aronis J., Buchanan B. G., Caruana R., Fine M. J., Glymour C., Gordon G., Hanusa B. H., Janosky J. E., Meek C., Mitchell T., Richardson T., and Spirtes P., “An evaluation of machine learning methods for predicting pneumonia mortality”, Artificial Intelligence in Medicine, 1997.

41. Dart S., “Concepts in Configuration Management Systems”, Proceedings of the Third International Workshop on Software Configuration Management, ACM SIGSOFT, pp. 1-18, 1991.

42. Darwin Charles, “The Origin of Species”, 6th Edition, John Murray, London, pp. 278, 1902.

43. Decker S., Daniel M., Erdmann M. and Studer R., “An enterprise reference scheme for integrating model based knowledge engineering and enterprise modeling”, Proceedings of the 10th European Workshop on Knowledge Acquisition, Modeling and Management, EKAW’97, Springer-Verlag, Heidelberg, Lecture Notes in Artificial Intelligence, LNAI, pp. 1319, 1997.

44. Dhar Vasant, Chou Dashin and Provost Foster, “Discovering Interesting Patterns for Investment Decision Making with GLOWER - A Genetic Learner Overlaid With Entropy Reduction”, Data Mining and Knowledge Discovery, 2000.

45. Dimitrios Gunopulos, Roni Khardont, Heikki Mannila and Hannu Toivonen, “Data mining, Hypergraph Transversals, and Machine Learning”, ACM, 1997.

46. Di-Rong Chen, QiangWu, Yiming Ying and Ding-Xuan Zhou, “Support Vector Machine Soft Margin Classifiers: Error Analysis”, Journal of Machine Learning Research 5, pp. 1143–1175, 2004.

47. Drucker Peter F., “The Practice of Management”, Harper & Row., New York, 1954.

48. Estublier J., Dami S. and Amiour M., “High Level Process Modeling for SCM Systems”, Proceedings of the Seventh International Workshop on Software Configuration Management, New York, 1997.

49. Eugene Ie, Jason Weston, William Stafford Noble and Christina Leslie, “Multi-class protein fold recognition using adaptive codes”, Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany, 2005.

50. Evfimievski Alexandre, Srikant Ramakrishnan, Agrawal Rakesh and Gehrke Johannes, “Privacy Preserving Mining of Association Rules”, KDD, 2002.

51. Fabio Aiolli and Alessandro Sperduti, “Multiclass Classification with Multi-Prototype Support Vector Machines”, Journal of Machine Learning Research 6, pp. 817–850, 2005.

52. Fahrmeir L., Haussler W. and Tutz G., “Diskriminanz analyse”, Multivariate statistische Verfahren, Verlag de Gruyter, Berlin, 1984.

53. Fayyad U., Piatetsky-Shapiro G. and Smyth P., “The KDD process for extracting useful knowledge from volumes of data”, CACM 39 11, pp. 27-34, 1996.

54. Feiler P., “Configuration Management Models in Commercial Environments”, Technical Report CMU/SEI-91-TR-7, Software Engineering Institute, Pittsburgh, Pennsylvania, Mar 1991.

55. Ferri C., J. Hernández-Orallo and Salido M.A., “Volume Under the ROC Surface for Multi-class Problems”, ECML, 2003.

56. Floyd C., “A Systematic Look at Prototyping”, Budde, R., Kuhlenkamp, K., Mathiassen, L. and Zullighoven, H. Eds. Approaches to Prototyping, Springer-Verlag: Heidelberg, pp. 1-17, 1984.

57. Forrest Stephanie, “Genetic Algorithms”, ACM Computing Surveys, Vol. 28, No. 1, 1996.

58. Frakes William, Terry Carol, “Software Reuse: Metrics And Models”, ACM Computing Surveys, Vol. 28, No. 2, 1996.

59. Franc Vojtˇech and V´aclav Hlav´aˇc, “Multi-class Support Vector Machine”, ICPR, Quebec, 2002.

60. Frank Eibe and Witten Ian H., “Generating Accurate Rule Sets Without Global Optimization”, Fifteenth International Conference on Machine Learning, pp. 144-151, 1998.

61. Frawley William J., Piatetsky-Shapiro Gregory and Matheus Christopher J., “Knowledge Discovery in Databases: An Overview”, AAAI, 1992.

62. Freund Y. and Mason L., “The alternating decision tree learning algorithm”, Proceeding of the Sixteenth International Conference on Machine Learning, Bled, Slovenia, pp. 124-133, 1999.

63. Friedman J., Hastie T. and Tibshirani R., “Additive Logistic Regression: a Statistical View of Boosting”, Stanford University, 1998.

64. Friedman N., Geiger D. and Goldszmidt M., “Bayesian Network Classifiers”, Machine Learning, Vol. 29, pp. 131–163, 1997.

65. Fu Yongjian and Han Jiawei, “Mining Multiple-Level Association Rules in Large Databases”, IEEE Transactions On Knowledge And Data Engineering, Sep 1999.

66. Garofalakis Minos N., Rastogi Rajeev and Shim Kyuseok, “SPIRIT: Sequential Pattern Mining with Regular Expression Constraints”, VLDB, 1999.

67. Garofolakis Minos, Hyun Dongjoon, Rastogi Rajeev and Shim Kyuseok, “Efficient algorithms for constructing Decision trees with constraints”, SIGKDD, 2000.

68. Giudici P., “Applied data mining”, John Wiley and Sons, New York, 2003.

69. Gomaa H., “The Impact of Rapid Prototyping on Specifying User Requirements”, ACM SIGSOFT, Software Engineering Notes, Vol. 8, No. 2, pp. 17-28, 1983.

70. Gonsalves G. C., Lederer A. L., Mahaney R. C. and Newkirk H. E., “A customer resource life cycle interpretation of the impact of the World Wide Web on competitiveness: expectations and achievements”, International Journal of Electronic Commerce Vol. 41, pp. 103-120, 1999.

71. Gordon V. S. and Bieman J. M., “Rapid Prototyping: Lessons Learned”, IEEE Software, Vol. 12, No. 1, pp. 85-95, 1994.

72. Gorman R. P. and Sejnowski T. J., “Analysis of hidden units in a layered network trained to classify sonar targets”, Neural networks, Vol. 1, Part 1, pp. 75–89, 1988.

73. Gupta S. K., Bhatnagar Vasudha and Wasan S. K., “Architecture for Knowledge Discovery and Knowledge Management”, Knowledge Information Systems, pp. 310-336, 2005.

74. Gupta S. K., Bhatnagar Vasudha and Wasan S. K., “On Mining of Data”, IETE Journal of Research, Special issue on Data and Knowledge Engineering, Vol. 47, No. 1, pp. 5-18, Jan/Feb 2001.

75. Hamill J., “The Internet and international marketing”, International Marketing Review 145, pp. 300-323, 1997.

76. Han J. and Pei J., “Mining Frequent Patterns by Pattern-Growth: Methodology and Implications", ACM SIGKDD Explorations Special Issue on Scaleble Data Mining Algorithms, 22, Dec 2000.

77. Han J., Cai Y., Cercone N. and Huang Y., “Discovery of Data Evolution Regularities in Large Databases”, Journal of Computer and Software Engineering, Vol. 31, pp. 41-69, 1995.

78. Han J., Fu Y. and Tang S., “Advances of the DBLearn System for Knowledge Discovery in Large Databases”, Proc. 1995 Int'l Joint Conf. on Artificial Intelligence IJCAI'95, Montreal, Canada, pp. 2049-2050, Aug. 1995.

79. Han J., Fu Y. and Ng R., “Cooperative Query Answering Using Multiple Layered Databases”, Proc. 2nd Int'l Conf. on Cooperative Information Systems CoopIS'94, Toronto, Canada, pp. 47-58, May 1994.

80. Han J., Fu Y., Huang Y., Cai Y. and Cercone N., “DBLearn: A system prototype for knowledge discovery in relational databases”, Proc. ACM-SIGMOD Int'l Conf. on Management of Data SIGMOD'94, Minneapolis, MN, May 1994 System prototype demonstration, pp. 516, 1994.

81. Han J., Fu Y., Koperski K., Melli G., Wang W. and Zaïane O. R., “Knowledge Mining in Databases: An Integration of Machine Learning Methodologies with Database Technologies”, Canadian Artificial Intelligence, Oct 1995.

82. Han J., Fu Y., Koperski K., Wang W. and Zaiane O., “DMQL: A Data Mining Query Language for Relational Databases”, SIGMOD'96 Workshop on Research Issues on Data Mining and Knowledge Discovery DMKD'96, Montreal, Canada, Jun 1996.

83. Han J., Gong W. and Yin Y., “Mining Segment-Wise Periodic Patterns in Time-Related Databases”, Proc. of Int'l Conf. on Knowledge Discovery and Data Mining KDD'98 , New York City, NY, pp. 214-218, Aug. 1998.

84. Han J., Lakshmanan L. V. S. and Ng R. T., “Constraint-Based, Multidimensional Data Mining”, COMPUTER special issues on Data Mining, 328, pp. 46-50, 1999.

85. Han J., Lu W. and Ooi B. C., “Discovery of General Knowledge in Large Spatial Databases”, Proc. of Far East Workshop on Geographic Information Systems FEGIS'93, Singapore, June 1993, pp. 275-289, 1993.

86. Han J., Nishio S., Kawano H. and Wang W., “Generalization-Based Data Mining in Object-Oriented Databases Using an Object-Cube Model”, Data and Knowledge Engineering , 251-2, pp. 55-97, 1998.

87. Han J., “Mining Knowledge at Multiple Concept Levels”, Proc. 4th Int'l Conf. on Information and Knowledge Management CIKM'95, Baltimore, Maryland, pp. 19-24, Nov 1995.

88. Han Jiawei, Cai Yandong and Cercone Nick, “Concept-Based Data Classification in Relational Databases”, AAAI Workshop on Knowledge Discovery in Databases, 1991.

89. Han Jiawei, Cai Yandong, and Cercone Nick, “Knowledge Discovery in Databases: An Attribute-Oriented Approach”, Proceedings of the 18th VLDB Conference, Vancouver, British Columbia, Canada, 1992.

90. Haykin Simon, “Neural Networks - A Comprehensive Foundation”, 2nd ed., Prentice-Hall, Englewood Cliffs, 1998.

91. Hearst Marti A., “Direction-Based Text Interpretation as an Information Access Refinement”, Text Based Intelligent Systems, 1992.

92. Holsheimer Marcel and Siebes Arno, “Data Mining: The Search for Knowledge in Databases”, Technical Report CS-R9406, CWI, Amsterdam, 1994.

93. Holsheimer Marcel, Kersten Martin L., Mannila Heikki and Toivonen Hannu, ”A Perspective on Databases and Data Mining”, Technical Report CS-R9531, CWI, Amsterdam, 1995.

94. Hong Chai and Carlotta Domeniconi, “An Evaluation of Gene Selection Methods for Multi-class Microarray Data Classification”, Proceedings of the Second European Workshop on Data Mining and Text Mining in Bioinformatics, 2004.

95. Hsieh Pi-Fuei, Wang Deng-Shiang and Hsu Chia-Wei, “A Linear Feature Extraction for Multiclass Classification Problems Based on Class Mean and Covariance Discriminant Information”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, No. 2, Feb 2006.

96. Hunt J., Lamers F., Renter J., and Tichy W.F., “Distributed Configuration Management via Java and the World Wide Web”, Proceedings of the Seventh International Workshop on Software Configuration Management, New York, 1997.

97. Hyde A., “The Proverbs of Total Quality Management: Recharting the Path to Quality Improvement in the Public Sector”, Public Productivity and Management Review, 161, pp. 25-37, 1992.

98. Iba W. and Langley P., “Induction of one-level decision trees”, Proc. of the Ninth International Machine Learning Conference, Aberdeen, Scotland: Morgan Kaufmann, 1992.

99. Ishikawa K., “What is Total Quality Control? The Japanese way”, Englewood Cliffs, New Jersey, Prentice- Hall, 1985.

100. Jacobson I., Booch G. and Rumbaugh J., “Unified Software Development Process, Object Technology Series”, Addison-Wesley, New York, NY, 1999.

101. Jain A. K., Murty M. N. and Flynn P. J., “Data Clustering: A Review”, ACM Computing Surveys, Vol. 31, No. 3, September 1999.

102. Jarzabek S. and Ling T.W., “Model-based support for business reengineering”, Information and Software Technology, Vol. 38, No. 5, pp. 355-74, 1996.

103. Jason D., Rennie M. and Rifkin Ryan, “Improving Multiclass Text Classification with the Support Vector Machine”, Massachusetts Institute of Technology as AI Memo 2001-026 and CBCL Memo 210, October 2001.

104. Johannes F¨urnkranz and Flach Peter A., “An Analysis of Rule Evaluation Metrics”, Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

105. John George H. and Langley Pat, “Estimating Continuous Distributions in Bayesian Classifiers”, Eleventh Conference on Uncertainty in Artificial Intelligence, San Mateo, pp. 338-345, 1995.

106. Joshi R. C., Zahid M. A. H. and Mittal Ankush, “Least common ancestor based efficient method for constructing rooted super trees”, Journal of Bioinformatics and Biomedical Engineering, Vol. 1, No. 5, pp. 1-6, 2005.

107. Kaushaar J. M. and Shirland L. E., “A Prototyping Method for Applications Development End Users and Information Systems Specialists”, MIS Quarterly, Vol. 9, No. 4, pp. 189 -197, 1985.

108. Kawano H., Nishio S., Han J., and Hasegawa T., “How Does Knowledge Discovery Cooperate with Active Database Techniques in Controlling Dynamic Environment?”, Proc. 5th Int'l Conf. on Database and Expert Systems Applications DEXA'94, Athens, Greece, pp. 370-379, Sep 1994.

109. Keerthi S. S., Shevade S. K., Bhattacharyya C. and Murthy K. R. K., “Improvements to Platt’s SMO Algorithm for SVM Classifier Design”, Neural Computation,133, pp. 637-649, 2001.

110. Khomyakov M. and Bider I., “Achieving workflow flexibility through taming the chaos”, paper presented at OOIS 2000 – 6th International Conference on Object Oriented Information Systems, Springer-Verlag, Berlin, pp. 85-92, 2000.

111. King R., Feng C. and Shutherland A., “Statlog: comparison of classification algorithms on large real world problems”, Applied Artificial Intelligence, 1995.

112. Kirkwood, C., Andrews B. and Mowforth P., “Automatic detection of gait events: a case study using inductive learning techniques”, Journal of biomedical engineering, 1123, pp. 511–516, 1989.

113. Kish Leslie, “Statistical Design for Research”, John Wiley and Sons, 1987.

114. Klemettinen Mika, Mannila Heikki and Toivonen Hannu, “A data-mining methodology and its application to semi-automatic knowledge acquisition”, DEXA, Toulouse, France, pp. 670-677, 1997.

115. Kohavi Ron and Sommerfield Dan, ”Data Mining using MLC++, A Machine Learning Library in C++”, Tools with AI, pp. 234–245, 1996.

116. Kohavi Ron, “The Power of Decision Tables”, 8th European Conference on Machine Learning, pp. 174-189, 1995.

117. Komarek P., Gray A., Liu T. and Moore A., “High Dimensional Probabilistic Classification for Drug Discovery”, Biostatics, COMPSTAT, 2004.

118. Kovalerchuk B., “Relational Text Mining and Visualization”, Knowledge-Based Intelligent Information Engineering Systems and Allied Technologies, KES 2002. IOS Press, Amsterdam, part 2, pp. 1549-1554, 2002.

119. Kovalerchuk B. and Andonie R., “Neural Networks for Data Mining: Constrains and Open Problems”, Proceedings of the 12th European Symposium on Artificial Neural Networks ESANN’2004, M. Verleysen ed., Bruges, Belgium, pp. 449-458, April 2004.

120. Kramer Stefan and Kaindl Hermann, “Coupling And Cohesion Metrics For Knowledge-Based Systems Using Frames And Rules”, ACM Transactions On Software Engineering And Methodology, Vol. 13, No. 3, pp. 332–358, 2004.

121. Kropotov D. A. and Vetrov D. P., “An Automatic Relevance Determination Procedure Based on Akaike Information Criterion for Linear Regression Problems”, ICML Workshop on Sparse Optimization and Variable Selection, 2008.

122. Kueng P. and Kawalek P., “Goal-based business process models: creation and evaluation”, Business Process Management Journal, Vol. 3, No. 1, pp. 17-38, 1997.

123. Kumar Vipin, Joshi Mahesh V. and Agrawal Ramesh C., “Mining Needles in a Haystack: Classifying Rare Classes via Two-Phase Rule Induction”, SIGMOD'01 conference on Management of Data, 2001.

124. Kumar Vipin, Joshi Mahesh V. and Agrawal Ramesh C., “Evaluating Boosting Algorithms to Classify Rare Classes: Comparison and Improvements”, First IEEE International Conference on Data Mining, 2001.

125. Kun Deng, Chris Bourke, Scott Stephen and Vinodchandran N. V., “New Algorithms for Optimizing Multi-Class Classifiers via ROC Surfaces”, Proceedings of the ICML 2006 workshop on ROC Analysis in Machine Learning, Pittsburgh, USA, 2006.

126. Leblang D. B., “Managing the Software Development Process with ClearGuide”, Proceedings of the Seventh International Workshop on Software Configuration Management, New York, 1997.

127. LeCun Y., Jackel L. D., Bottou L., Brunot A., Cortes C., Denker J. S., Drucker H., Guyon I.,Muller U. A., Sackinger E., Simard P. and Vapnik V., “Comparison of learning algorithms for handwritten digit recognition”, International Conference on Artificial Neural Networks, Paris, pp. 53-60, 1995.

128. Lee S., “Business use of internet-based information systems: the case of Korea”, European Journal of Information Systems, Vol. 12, Issue 3, Sep 2003.

129. Li Tao, Zhu Shenghuo and Ogihara Mitsunori, “A New Distributed Data Mining Model based on similarity”, ACM, 2003.

130. Li W., Han J. and Pei J., “CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules”, Proc. 2001 Int. Conf. on Data Mining ICDM'01, San Jose, CA, Nov 2001.

131. Lilian. H. Tang and Chen L., “Improved computation of beliefs based on confusion matrix for combining multiple classifiers”, Electronics Letters, Vol. 40, No. 4, February 2004.

132. Lim T.-S., Loh W.-Y. and Shih Y.-S., “A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms”, Machine Learning, Vol. 40, pp. 203-228, 2000.

133. Liu Bing and Hu Minqing, “Mining and Summarizing Customer Reviews”, ACM, 2004.

134. Liu Guimi, Lu Hongjun, Lu Wenwu, Xu Yabo and Yu Jeffery Xu, “Efficient mining of frequent patterns Using ascending frequency ordered Prefix Tree”, Data Mining and Knowledge Discovery, Kluwer Academic Publishers, 2004.

135. Maniatty William A. and Zaki Mohammed J., “Systems Support for Scalable Data Mining”, ACM-SIGKDD, 2000.

136. Mannila Heikki and Hannu Toivonen, “Levelwise Search and Borders of Theoriesin Knowledge Discovery”, Data Mining and Knowledge Discovery, Vol. 1, pp. 241–258, 1997.

137. Mannila Heikki, Toivonen Hannu and Verkamo A. Inkeri, “Efficient Algorithms for Discovering Association Rules”, KDD Workshop, pp. 181-192, 1994.

138. Mannila Heikkie, “Local and global Methods in data mining: Basic Techniques and open problems”, ICALP, 2002.

139. Marca D. A. and McGowan C. L., “IDEF0/SADT: Business Process and Enterprise Modeling”, Eclectic Solutions Inc., San Diego, CA, 1993.

140. Matthew E. Taylor, Cynthia Matuszek, Bryan Klimt and Michael Witbrock. “Autonomous Classification of Knowledge into an Ontology”, 20th International FLAIRS Conference FLAIRS, Key West, Florida, May 2007.

141. Micheline Kamber, Han Jiawei and Chiang Jenny Y., “Using Data Cubes for Metarule-Guided Mining of Multi-Dimensional Association Rules”, Technical Report CS-TR 97-10, School of Computing Science, Simon Fraser University, May 1997.

142. Micheline Kamber, Winstone Lara, Gon Wang and Han Jiawei, “Generalization and Decision Tree Induction: Efficient Classification in Data Mining”, RIDE, 1997.

143. Milewski B., “Distributed Source Control System”, Proceedings of the Seventh International Workshop on SoRware Configuration Management, New York, 1997.

144. Mitchell T., Buchanan B., DeJon G. G., Dietterich T., Rosenbloom P. and Waibel A., "Machine Learning", Annual Review of Computer Science, Vol. 4, pp. 417-433, 1990.

145. Mobasher Bamshad, Han E., Karypis G. and Kumar V., “Clustering in a High-Dimensional Space Using Hypergraph Models”, Technical Report, Department of Computer Science, Univeresity of Minnesota, 1998.

146. Mobasher Bamshad, Leszczylowski J., Pigozzi D. and Slutzki G., “Negation as Partial Failure”, Proceedings of the 2nd International Workshop on Logic Programming and Non-monotonic Reasoning, L. M. Pereira and A Nerode eds., MIT Press, pp. 244-262, 1993.

147. Mobasher1Bamshad, Pigozzi Don and Slutzki Giora, “Algebraic Semantics for Knowledge-based Logic Programs”, Proceedings of the Workshop on Uncertainty in Databases and Deductive Systems WUDDS-94, Ithaca, NY, November 1994.

148. Nauman J. D. and Jenkins M., “Prototyping: The New Paradigm for Systems Development”, MIS Quarterly, Vol. 6, No. 3, pp. 29-44, 1982.

149. Ng R. and Han J., “Efficient and Effective Clustering Method for Spatial Data Mining”, Proc. of Int'l Conf. on Very Large Data Bases VLDB'94, Santiago, Chile, pp. 144-155, Sep 1994.

150. Niculescu-Mizil A. and Caruana R., “Predicting good probabilities with supervised learning”, Proc. 22nd International Conference on Machine Learning ICML'05, 2005.

151. Nishisato S., “Analysis of Categorical Data: Dual Scaling and its Applications”, University of Toronto Press, Toronto, 1980.

152. Palvia P. and Nosek J. T., “An Empirical Evaluation of System Development Methodologies”, Information Resources Management Journal, Vol. 3, pp. 23-32, 1990.

153. Park Jong Soo and Kim Do-Hyung, “An Effective Algorithm for Sub-dimensional Clustering of High Dimensional Data”, KIPS, 2003.

154. Park Jong Soo, Chen Ming-Syan and Yu Philip S., “An Effective Hash Based Algorithm for Mining Association Rules”, SIGMOD Conference, pp. 175-186, 1995.

155. Perlich C., Provost F. and Simono J. S., “Tree induction vs. logistic regression: a learning-curve analysis”, J. Mach. Learn. Res., Vol. 4, pp. 211-255, 2003.

156. Piatetsky-Shapiro Gregory, Matheus Christopher, Smyth Padhraic and Uthurusamy Ramasamy, “Progress and Challenges in Knowledge Discovery in Databases”, AAAI, KDD-93, 1993.

157. Piatetsky-Shapiro Gregory, “Knowledge Discovery in Real Databases”, IJCAI-89 Workshop, AI Magazine, Vol. 11, Issue 5, January 1991.

158. Platt J., “Probabilistic outputs for support vector machines and comparison to regularized likelihood methods”, Adv. in Large Margin Classifiers, 1999.

159. Poulin J. S., “Measuring Software Reuse: principles, practices, and economic models”, Reading, MA, Addison-Wesley, 1997.

160. Provost F., Jensen D. and Oates T., “Efficient progressive sampling”, Fifth ACM SIGKDD, International Conference on Knowledge Discovery and Data Mining, San Diego, USA, 1999.

161. Provost Foster J. and Kohavi Ron., “On Applied Research in Machine Learning”, Machine Learning, pp. 127-132, 1998.

162. Provost F. and Domingos P., “Tree induction for probability-based rankings”, Machine Learning, 2003.

163. Ripley B., “Statistical aspects of neural networks”, Chaos and Networks - Statistical and Probabilistic Aspects, Chapman and Hall, 1993.

164. Robertson T., Wright F. and Dykstra R., “Order restricted statistical inference”, John Wiley and Sons, New York, 1988.

165. Rolland C., “L’e-lyee: L’escritoire and Lyeeall”, Information and Software Technology, Vol. 44, pp. 185-94, 2002.

166. Rolland C., “Reasoning with goals to engineer requirements”, Proceedings of ICEIS’03– Fifth International Conference on Enterprise Information Systems, Angers, France, pp. 11-19, 2003.

167. Ronald Fagin, Ravi Kumar and D. Sivakumar, “Comparing Top K Lists”, Siam J. Discrete Math, Society for Industrial and Applied Mathematics, Vol. 17, No. 1, pp. 134–160, 2003.

168. Royce W. W., “Managing the Development of Large Software Systems: Concepts and Techniques”, Proceedings WESCON, Aug 1970.

169. Rumbaugh J., Blaha M., Premerlani W., Eddy F. and Lorensen W., “Object Oriented Modeling and Design”, Prentice-Hall, New York, NY, 1991.

170. Shadmehr R. and D’Argenio Z., “A comparison of a neural network based estimator and two statistical estimators in a sparse and noisy environment”, IJCNN-90: proceedings of the international joint conference on neural networks, Ann Arbor, MI. IEEE Neural Networks Council, pp. 289–292,1990.

171. Shasha Dennis, Cantone Domenico, Ferro Alfredo, Pulvirenti Alfredo and Reforgiata Diego, ”Antipole Tree Indexing to Support Range Search and K-Nearest Neighbor Search in Metric Spaces”, IEEE Transactions on Knowledge and Data Engineering, Vol. 17, No. 5, pp. 535-550, Apr 2005.

172. Shasha Dennis, Wang Xiong, Wang Jason T-L, Shapiro Bruce, Rigoutsos Isidore and Zhang Kaizhong, “Finding Patterns in Three Dimensional Graphs: Algorithms and Applications to Scientific Data Mining”, IEEE Transactions on Knowledge and Data Engineering, pp. 731-749, 2002.

173. Shevade S. K., Keerthi S. S., Bhattacharyya C. and Murthy K. R. K., “Improvements to the SMO Algorithm for SVM Regression”, IEEE Transactions on Neural Networks, 1999.

174. Shrikant Ramakrishnan, Vu Quoc and Agrawal Rakesh, “Mining Association Rules With Item Constraints”, AAAI, 1997.

175. Spikovska L. and Reid M. B., “An empirical comparison of id3 and honns for distortion invariant object recognition”, In TAI-90: tools for artificial intelligence: proceedings of the 2nd international IEEE conference, Los Alamitos, CA, IEEE Computer Society Press, 1990.

176. Stake Robert E., “Case Studies”, Handbook of qualitative Research, Chapter 14, pp. 236-247, Sage, 1994.

177. Stasko John and Robert Amar, "Knowledge Precepts for Design and Evaluation of Information Visualizations", IEEE Transactions on Visualization and Computer Graphics, Vol. 11, No. 4, pp. 432-442, Jul/Aug 2005.

178. Sumner Marc, Frank Eibe and Hall Mark, “Speeding up Logistic Model Tree Induction”, 9th European Conference on Principles and Practice of Knowledge Discovery in Databases, pp. 675-683, 2005.

179. Tao Li, Shenghuo Zhu and Mitsunori Ogihara, “Using Discriminant Analysis for Multi-class Classification”, Proceedings of the Third IEEE International Conference on Data Mining ICDM, 2003.

180. Tate G. and Verner J., “Case Study of Risk Management, Incremental Development and Evolutionary Prototyping”, Information and Software Technology, Vol. 42, No. 4, pp. 207-214, 1990.

181. Thomas C., Landgrebe W. and Robert P.W. Duin, “Approximating the multiclass ROC by pairwise analysis”, Pattern Recognition Letters, Vol. 28, pp. 1747–1758, Elsevier, 2007.

182. Thomas C., Landgrebe W. and Robert P.W. Duin, “Efficient multiclass ROC approximation by decomposition via confusion matrix perturbation analysis”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007.

183. Tichy W. F., “Tools for software configuration management”, Proceedings of the International Workshop on Software Version and Configuration Control Grassau, Germany, J. F. H. Winkler, Ed., Teubner Verlag, pp. 1–20, 1988.

184. Ting-Fan Wu, Chih-Jen Lin and Ruby C. Weng, “Probability Estimates for Multi-class Classification by Pairwise Coupling”, NIPS, 2003.

185. Toivonen Hannu T. T., Onkamo Pa¨ivi, Vasko Kari, Ollikainen Vesa, Sevon Petteri, Mannila Heikki, Herr Mathias and Kere Juha, “Data Mining Applied to Linkage Disequilibrium Mapping”, Am. J. Hum. Genet. 67, pp. 133–145, 2000.

186. Wang Y. and Witten I. H., “Modeling for optimal probability prediction”, Proceedings of the Nineteenth International Conference in Machine Learning, Sydney, Australia, pp. 650-657, 2002.

187. Weill P., “The relationship between investment in information technology and firm performance: a study of the valve-manufacturing sector”, Information Systems Research, Vol. 34, pp. 307-333, 1992.

188. Witten I. H. and Frank E., “Data Mining – Practical Machine Learning Tools and Techniques”, Elsevier Inc., 2005.

189. Witten I. H. and Frank E., “Data Mining: Practical machine learning tools and techniques with java implementations”, Morgan Kaufmann, 2000.

190. Yair Even-Zohar and Dan Roth, “A Sequential Model for Multi-Class Classification”, EMNLP, 2001.

191. Yip Kevin Y., Cheung David W. and Ng Michael K., “HARP: A Practical Projected Clustering Algorithm”, IEEE Transactions on Knowledge and Data Engineering, 2004.

192. Yoav Freund, Robert E. Schapire, “Experiments with a new boosting algorithm”, Thirteenth International Conference on Machine Learning, San Francisco, pp. 148-156, 1996.

193. Zadeh L. A., “Fuzzy Sets”, Information and Control, Vol. 8, pp. 338-353, 1965.

194. Zadrozny B. and Elkan C., “Obtaining calibrated probability estimates from decision trees and naïve bayesian classifiers”, ICML, 2001.

195. Zadrozny B. and Elkan C., “Transforming classifier scores into accurate multi-class probability estimates”, KDD, 2002.

196. Zaïane O. R. and Han J., “Resource and Knowledge Discovery in Global Information Systems: A Preliminary Design and Experiment”, Proc. 1st Int'l Conf. on Knowledge Discovery and Data Mining KDD'95, Montreal, Canada, pp. 331-336, Aug 1995.

197. Zaki Mohammed J. and Pan Yi, “Introduction: Recent Developments in Parallel and Distributed Data Mining”, Distributed and Parallel Databases, Kluwer Academic Publishers, Vol. 11, pp. 123–127, 2002.

198. Zaki Mohammed J., “Parallel Sequence Mining on Shared-Memory Machines. Parallel and Distributed Computing”, Vol. 61, pp. 401-426, 2001.

199. Zaki Mohammed J., “Scalable Algorithms for Association Mining”, IEEE Transactions On Knowledge And Data Engineering, Vol. 12, No. 3, May/June 2000.

200. Zaniolo Carlo and Wang Haixun, “ATLaS: A Native Extension of SQL for Data Mining”, SIAM International Conference on Data Mining 2003, San Francisco, CA, May 2003.

201. Zaniolo Carlo and Wang Haixun, “CMP: A Fast Decision Tree Classifier Using Multivariate Predictions”, ICDE, pp. 449-460, 2000.

202. Zhang Y., Li X. Rong, Zhu Z. and Zhang H., “A new clustering and training method for radial basis function networks”, Proc. of the Intern. Conf. on Neural Networks - ICNN, pp. 311–316, 1996.

203. Zhong-Hui W, Wan-Gui Li, Yun-Ze Cai And Xiaoiming Xu., “An Empirical Comparison of Ensemble Classification Algorithms with Support Vector Machines”, Proceedings of the Third International Conference on Machine Laming and Cybernetics, Shanghai, pp. 26-29, Aug 2004.

204. Zhu Ji, Rosset Saharon, Zou Hui and Trevor Hastie, “Multiclass AdaBoost”, Statistics and Its Interface, Vol. 2, pp. 349–360, 2009.

List of Publications

International Journals: Seven

1. Manchanda Sanjeev , Dave Mayank and Singh S. B., “Metrics Directed Selection of Supervised Learning Algorithms”, International Journal of Computing, Ukraine (Accepted and scheduled to be published in year 2010).
2. Manchanda Sanjeev , Dave Mayank and Singh S. B., "Metrics Directed Selection Of Supervised Learning Algorithms For Multi Class Problems", International Journal of Computer Science and Security, Malaysia. (Accepted and scheduled to be published in year 2010).
3. Manchanda Sanjeev, Dave Mayank and Singh S. B., “Change Management And Software Reuse Supportive ‘Genetic Information System Development And Maintenance’ Model”, International Journal of Software Engineering and Knowledge Engineering (IJSEKE), Vol. 19, No. 1, pp. 113-136, USA, Feb. 2009.
4. Manchanda Sanjeev , Dave Mayank and Singh S. B., " Exploring Knowledge for a Common Man Through Mobile Services and Knowledge Discovery in Databases ", International Journal of Computer Science and Security, Volume 3, Issue 1, Malaysia, pp. 43 – 61,January 2009.
5. Manchanda Sanjeev , Dave Mayank and Singh S. B., “’Genetic Information System Development and Maintenance’ Model For Effective Software Maintenance and Reuse”, International Journal of Engineering, Volume 1, Issue 1, pp. 1 – 20, Malaysia, May 2007.
6. Manchanda Sanjeev , Dave Mayank and Singh S. B., “An Empirical Comparison Of Supervised Learning Processes”, International Journal of Engineering, Volume 1, Issue 1, pp. 21 – 38, Malaysia, May 2007.
7. Manchanda Sanjeev , Dave Mayank and Singh S. B., “Customized and Secure Image Steganography Through Random Numbers Logic”, Signal Processing: An International Journal, Volume 1, Issue 1, pp. 1 – 16, Malaysia, May 2007.

International Conferences: Three

1. Manchanda Sanjeev, Dave Mayank and Singh S. B., “An Empirical Comparison of Supervised Learning Algorithms Through Cross Validation, Calibration and Stacking”, International Conference on Software, Knowledge, Information Management and Applications (SKIMA), sponsored by IEEE and IAS, Kathmandu, Nepal, pp. 18-23, 18-21, March 2008.
2. Manchanda Sanjeev, Dave Mayank, Singh S.B., “Pseudo Random Numbers Based Methods for Customized and Secure Image Steganography”, IRMA International Conference, Vancouver, Canada, organizaed by IRMA, USA, pp. 1608-1814, 19-23, May 2007.
3. Manchanda Sanjeev, Dave Mayank, Singh S.B. and Kasana H.S., “Knowledge: The Base For Decision-Making Anything Anytime and Anywhere”, International Conference On Computing, Communications and Control Technologies (CCCT '05), organized by University of Texas, Austin, Texas, USA, 24-27, July 2005.

Appendix A: Glimpses of Datasets

Datasets used for experimentation in this thesis are listed here. First few instances are displayed and few columns are removed to fit datasets in this page.

1. Adult Datset

Age	Work Class	Fnlwgt	Education	Education-num	Marital-Status	…	Hours-per-week	Native-country	Class
25	1	226802	3	7	3	…	40	1	0
38	1	89814	4	9	1	…	50	1	0
28	5	336951	6	12	1	…	40	1	1
44	1	160323	2	10	1	…	40	1	1
18	0	103497	2	10	3	…	30	1	0
34	1	198693	13	6	3	…	30	1	0
29	0	227026	4	9	3	…	40	1	0
63	2	104626	5	15	1	…	32	1	1
24	1	369667	2	10	3	…	40	1	0

2. Cov Type Dataset

Elevation	Aspect	Slope	Horizontal Distance To Hydrology	…	Horizontal Distance To Fire Points	Wilderness Area_1	…	Wilder-ness Area_4	Soil Type 1	…	Soil Type 40	Cover Type
1	1	1	1	…	1	1	…	1	1	…	1	0
1	1	1	1	…	1	1	…	1	1	…	1	0
2	2	1	1	…	1	1	…	1	1	…	1	0
2	3	2	1	…	1	1	…	1	1	…	1	0
1	1	1	1	…	1	1	…	1	1	…	1	0
1	2	1	2	…	1	1	…	1	1	…	1	0
1	1	1	1	…	1	1	…	1	1	…	1	0
1	1	1	1	…	1	1	…	1	1	…	1	0
1	1	1	1	…	1	1	…	1	1	…	1	0

3. DS100 Dataset

X1	X2	X3	X4	X5	…	X99	X100	Class
-4.1929	-2.5171	-0.47378	0.62843	2.6874	…	0.15817	0.13074	0
-6.6806	-2.9697	-0.99098	-2.4662	-2.1155	…	0.016469	0.10418	0
-0.25967	-0.42383	-0.08975	-0.10335	0.21026	…	-0.08708	-0.07456	0
-1.1564	-0.84808	0.25946	0.41142	1.4374	…	-0.05851	-0.32066	0
-0.66666	-0.21439	-0.04186	0.12996	0.36305	…	-0.05437	0.07039	0
-0.17517	-0.04397	0.31662	0.042977	-0.12249	…	0.050117	-0.07797	0
-1.3412	1.1629	0.029398	0.13513	-0.49758	…	0.26673	0.058246	0
-0.66152	-0.84928	-0.28197	-0.0304	0.29475	…	-0.1975	-0.00817	0
-1.7492	-1.7055	-0.54523	-0.6485	0.044507	…	0.00814	-0.11377	1

4. Letter Dataset

x-box	y-box	Width	High	onpix	x-bar	…	y-ege	yegvx	Class
2	4	4	3	2	7	…	5	6	1
4	7	5	5	5	5	…	7	10	1
7	10	8	7	4	8	…	5	10	1
4	9	5	7	4	7	…	0	8	0
6	7	8	5	4	7	…	3	7	0
4	7	5	5	3	4	…	1	7	0
6	10	8	8	4	7	…	1	8	1
1	0	2	0	1	6	…	4	9	1
5	9	7	6	7	7	…	2	8	0

5. Oracle Dataset

Var_1	Var_2	Var_3	Var_4	Var_5	Var_6	…	Var_26	Var_27	Class
7	3	2	90	1	40	…	776	14629	0
4	4	2	96	1	61	…	1100	18910	0
24	1	1	15	2	35	…	300	17464.5	0
17	2	2	99	2	22	…	490	20420.75	0
15	4	2	31	2	39	…	0	14053	0
13	3	2	74	1	40	…	0	21115	0
13	3	2	74	1	43	…	1200	26406	1
4	4	2	2	1	60	…	1000	24482	1
10	1	1	9	2	46	…	1000	28094	1

6. Pen Digits Dataset

V1	V2	V3	V4	V5	…	V15	V16	Class
47	100	27	81	57	…	40	98	1
0	89	27	100	42	…	100	6	0
0	57	31	68	72	…	16	0	0
0	100	7	92	5	…	67	0	0
0	67	49	83	100	…	47	0	0
100	100	88	99	49	…	20	20	1
0	100	3	72	26	…	66	0	0
0	39	2	62	11	…	0	57	0
13	89	12	50	72	…	100	100	1

7. Physics Dataset

V1	V2	V3	V4	V5	…	V77	V78	Class
0	0	0	0	0	…	0	0	0
0.92	0.818	-0.646	-1	0	…	0.302	0.951	0
0.868	0.178	0.151	-1	0	…	0.00142	0.883	1
0	0	0	0	1.58	…	0	0	0
0	0	0	0	0	…	0	0	0
0	0	0	0	0	…	0	0	1
0	0	0	0	0	…	0	0	1
0	0	0	0	0	…	0	0	0
0	0	0	0	0	…	0	0	1

8. Satellite Dataset

A1	A2	A3	A4	A5	…	A35	A36	Class
92	115	120	94	84	…	113	87	0
84	102	106	79	84	…	104	79	0
84	102	102	83	80	…	104	79	0
80	102	102	79	84	…	104	79	0
84	94	102	79	80	…	109	87	0
80	94	98	76	80	…	109	87	0
76	102	106	83	76	…	104	79	0
76	102	106	87	80	…	100	79	0
76	89	98	76	76	…	93	71	1
76	94	98	76	76	…	93	67	1

9. Shuttle Dataset

VA1	VA2	VA3	VA4	VA5	VA6	...	VA9	Class
50	21	77	0	28	0	…	22	0
55	0	92	0	0	26	…	56	1
53	0	82	0	52	-5	…	2	0
37	0	76	0	28	18	…	8	0
37	0	79	0	34	-26	…	2	0
85	0	88	-4	6	1	…	80	1
56	0	81	0	-4	11	…	62	1
55	-1	95	-3	54	-4	…	2	0
53	8	77	0	28	0	…	24	1
37	0	101	-7	28	0	…	8	0

10. TIC 2000 Dataset

Mostype	Maanthui	Mgemomv	Mgemleef	Moshoofd	…	Ainboed	Abystand	Class
33	1	3	2	8	…	0	0	0
37	1	2	2	8	…	0	0	0
37	1	2	2	8	…	0	0	0
9	1	3	3	3	…	0	0	0
40	1	4	2	10	…	0	0	0
23	1	2	1	5	…	0	0	0
39	2	3	2	9	…	0	0	0
33	1	2	3	8	…	0	0	0
33	1	2	4	8	…	0	0	0
11	2	3	3	3	…	0	0	0

Appendix B: List of Abbreviations and Symbols

Abbreviation	Description
AdaBoostADTreeIso*	Fuzzy Boundaries of Regression based clusters through AdaBoost-ADTree (Classification algorithms) with Isotonic Regression (Regression algorithm)
AdaBoostADTreeLinear*	Fuzzy Boundaries of Regression based clusters through AdaBoost-ADTree (Classification algorithms) with Linear Regression (Regression algorithm)
AdaBoostADTreePace*	Fuzzy Boundaries of Regression based clusters through AdaBoost-ADTree (Classification algorithms) with Pace Regression (Regression algorithm)
AdaBoostADTreeRBF *	Fuzzy Boundaries of Regression based clusters through AdaBoost-ADTree (Classification algorithms) with RBF Network (Regression algorithm)
AdaBoostADTreeSVM*	Fuzzy Boundaries of Regression based clusters through AdaBoost-ADTree (Classification algorithms) with SVM Regression (Regression algorithm)
ADTree	Alternating Decision Tree
ADTreeIso*	Fuzzy Boundaries of Regression based clusters through ADTree (Classification algorithm) with Isotonic Regression (Regression algorithm)
ADTreeLinear*	Fuzzy Boundaries of Regression based clusters through ADTree (Classification algorithm) with Linear Regression (Regression algorithm)
ADTreePace*	Fuzzy Boundaries of Regression based clusters through ADTree (Classification algorithm) with Pace Regression (Regression algorithm)
ADTreeRBFNet*	Fuzzy Boundaries of Regression based clusters through ADTree (Classification algorithm) with RBFNet Regression (Regression algorithm)
ADTreeSVMReg*	Fuzzy Boundaries of Regression based clusters through ADTree (Classification algorithm) with SVMReg Regression (Regression algorithm)
AUC	Area Under the ROC curve
BaggADTreeIso*	Fuzzy Boundaries of Regression based clusters through Bagged-ADTree (Classification algorithms) with Isotonic Regression (Regression algorithm)
BaggADTreeLinear*	Fuzzy Boundaries of Regression based clusters through Bagged-ADTree (Classification algorithms) with Linear Regression (Regression algorithm)
BaggADTreePace*	Fuzzy Boundaries of Regression based clusters through Bagged-ADTree (Classification algorithms) with Pace Regression (Regression algorithm)
BaggADTreeRBFNet*	Fuzzy Boundaries of Regression based clusters through Bagged-ADTree (Classification algorithms) with RBF Network Regression (Regression algorithm)
BaggADTreeSVMReg*	Fuzzy Boundaries of Regression based clusters through Bagged-ADTree (Classification algorithms) with SVM Regression (Regression algorithm)
CovType	Forest Cover Type
CPE	Class Probability Estimator
CQL	Library of Customized and Quality based Processes/Services
DBA	Database Administrator
DS100	Life Sciences Dataset with 100 Attributes

Table B.1a: List of Abbreviations used in this research

*These terms are used for all classification algorithms used for this research i. e. ADTree is replaced with other names.

Abbreviation	Description
IBk	Instance-based K Nearest Neighbour learning algorithms
ID3	Iterative Dichotomiser 3
IsoFuzzy	Fuzzy Boundaries of Isotonic Regression Based Clusters
IsoFuzzy	Isotonic Regression algorithm used for Fuzzy Boundaries of Regression based clusters formation
KDD	Knowledge discovery in databases
L	Library of Processes/Services
LibSVM	A Library for Support Vector Machines
LinearFuzzy	Fuzzy Boundaries of Linear Regression Based Clusters
LinearFuzzy	Linear Regression algorithm used for Fuzzy Boundaries of Regression based clusters formation
LOC	Lines of Code
MLP	Multi Layer Perceptron
PACE Regression	Projection Adjustment by Contribution Estimation
PaceFuzzy	Fuzzy Boundaries of PACE Regression Based Clusters
PaceFuzzy	Pace Regression algorithm used for Fuzzy Boundaries of Regression based clusters formation
PART	Partial Decision Trees
Pk	k-th Process/Service
PS	Progressive Sampling
QL	Library of processes/services fulfilling Quality Criteria
RBFNet	Gaussian Radial Basis Function Network
RBFNet	RBF Network algorithm used for Fuzzy Boundaries of Regression based clusters formation
RBFNetFuzzy	Fuzzy Boundaries of RBF Network Regression Based Clusters
REPTree	Reduced Error Pruned Tree
ROC Curve	Receiver Operating Characteristic Curve
SCQL	Library of Secure, Customized and Quality based Processes/Services
SMO	Sequential Minimal Optimization
SPki	i-th Sub-Process/Service of k-th process
SVM	Support Vector Machine
SVMReg	Support Vector Machine Regression
SVMReg	SVM Regression algorithm used for Fuzzy Boundaries of Regression based clusters formation
SVMRegFuzzy	Fuzzy Boundaries of SVM Regression Based Clusters
TIC2000	The Insurance Company Dataset of Year 2000
ZeroR	Zero-attribute-rule

Table B.1b: List of Abbreviations used in this research

Symbols	Description
μ_X	a characteristic function of a crisp set
$\mu_A(x)$	The membership function of a fuzzy set
ξ	Train Set
Ψ	Test Set
Δ_i	ith incremental subset of train set
Δ	Union of records from δ' and δ"
Δ'	No. of records separated by Fuzzy Boundaries of Regression based clusters from Incremental Train set guided by CPE output
Δ"	No. of records separated by Fuzzy Boundaries of Regression based clusters from Incremental Train set guided by continuous output
Δ'	Union of records from δ and Δ_i
ξ'	Remaining Train set after applying Fuzzy Boundaries of Regression based clusters

Table B.2: List of Symbols used in this research