Development of Intelligent Prediction System using Data Mining Techniques

DR. SHAMPA SENGUPTA

DR. ASIT KUMAR DAS

Kripa Drishti Publications, Pune.

Development Of Intelligent Prediction System Using Data Mining Techniques

Dr. Shampa Sengupta

Dr. Asit Kumar Das

Kripa-Drishti Publications, Pune.

Book Title: **Development of Intelligent Prediction System using Data Mining Techniques**

Authored by: **Dr. Shampa Sengupta, Dr. Asit Kumar Das**

1st Edition

ISBN: 978-93-90847-37-2

Published: **November 2021**

Publisher:

Kripa-Drishti Publications

A/ 503, Poorva Height, SNO 148/1A/1/1A,
Sus Road, Pashan- 411021, Pune, Maharashtra, India.
Mob: +91-8007068686
Email: editor@kdpublications.in
Web: https://www.kdpublications.in

PREFACE

Now a day's, everything is being done through electronic media which generates huge amount of data in every moment. Most of the time, data are not static rather they are dynamic and transactional in nature. Retrieval of some interesting information from generated data is a very challenging task. Generally, each data set contains a large number of instances with huge number of features. Therefore, relevant feature selection and classification is one of the main objectives of data mining technique for knowledge discovery both in static and dynamic environment. Though many research works have been conducted for the data analysis of static and incremental data, still it is an ongoing research to handle with newly generated high dimensional data sets to obtain meaningful interpretations. The concerned issues are major requirements and challenges have been addressed in the book by developing optimal feature selection and classification algorithms using the concept of Rough Set Theory, Graph Theory, Genetic Algorithm, Particle Swarm Optimization, and so on. Developed algorithms have been applied in various benchmark datasets and in the real world agricultural field to classify and predict efficiently the unknown objects as a task for data analysis. All the methods are very useful with respect to Big Data analytics as it provides the optimal solutions.

Researchers are working on multi-dimensional dataset for performing different data analysis jobs. However, most of them follow a systematic approach and does some predetermined activities while analyzing dataset using pre-processing and classification techniques. The book establishes the fact that an integrated approach towards performance-based analysis is one of the necessities for designing efficient method to discover new knowledge. In order to satisfy the above requirements the book is written at contributing towards development of a hybrid framework by building performance-based pre-processing, and classification method using rough set theory, graph theory, statistical approach and evolutionary techniques to extract important and meaningful knowledge from the standard dataset.

The aim of presenting the book is to exploit data mining techniques in both the static and dynamic environment for feature selection and classification problems, such as (i) identifying the most relevant features (ii) select significant feature subset (iii) classifier construction and (iv) ensemble classifier design for classification.

Regarding the organization the chapters of the books are organized as follows. Overview of the feature selection and classification analysis is presented in *Chapter 2.* In particular, this chapter provides a thorough study of the mostly used feature selection and classification algorithms. *Chapter 3* presents the proposed static

feature selection methods for identification of important feature subset using Rough Set Theory, graph theory, clustering algorithm, and different mathematical and statistical approaches and applied to different benchmark datasets to achieve the optimum set of features. *Chapter 4* presents the proposed incremental feature selection methods for identifying the important feature subset in dynamic environment using Rough Set Theory and evolutionary algorithms. The work on feature selection methods both in static and dynamic environment provides a through comparative study of all the proposed algorithms based on the experimental results to determine the best method, which ultimately is used for constructing the classifier. *Chapter* 5 presents the construction of classification systems in static and dynamic environment based on the reduced datasets. Single classifier is not always a very good predictor for all the problems so the ensemble of classifier takes an important role for that purpose. The best combination of classifiers by fusing the different base classifiers integrating genetic algorithm and rough set theory is presented in this chapter. A novel incremental classifier designing method is proposed for the dynamic environment where data are gradually available with the varied time. *Chapter 6* discusses the application of both the static and incremental feature selection methods and classification algorithm in the agricultural field for predicting diseases. *Chapter 7* concludes the summary of the implemented concepts presented in the book along with its limitations and future scopes.

This book is primarily intended to serve as a reference book for graduate and master degree students of computer science domain and researchers of any domain of various Colleges and Universities. We hope this book will provide the necessary guidance to the students to work on this data analysis domain. The book is actually an outgrowth of our research experience for the last several years.

Acknowledgement

It is our pleasure to convey sincere thanks and acknowledge the people who supported us in different aspects for writing the book possible. First and foremost we would like to thank God for making us able to complete this book.

Our thanks go to all the colleagues who have contributed immensely to our professional time at Indian Institute of Engineering Science and Technology and MCKV institute of Engineering, for their endless help in all official and unofficial matters throughout the years. We are highly indebted to our students for providing us the necessary stimulus for writing the book.

We will feel rewarded if this book can be published for helping the development of the research studies. We also want to thank our publisher for their support during writing the book.

Finally we would like to thank our families for their endless support to complete this book.

Dedication

To my parents for their love, blessings and support

S. Sengupta

To my mother Chhabi Das for her unconditional love and encouragement

A. K. Das

INDEX

ISBN: 978-93-90847-37-2

Chapter 1

Introduction

In this digital era of e-commerce, e-governance and m-commerce, huge amount of data is produced in every moment, almost in every field. Now a day's data are not static rather it is dynamic and transactional in nature. So, to store these large amount of data, an environment is needed which is nothing but data warehouse. Data can be stored in the data warehouse, but analysis of these stored data for extraction of important, valuable, and relevant information is the key feature of knowledge discovery process known as knowledge discovery in databases (KDD). In the past decades, discovering of knowledge was done by manual analysis, and as a result interpretation of data often prone to error and very time consuming. Recent exponential growth of data volume demands an efficient, scalable, and expert knowledge discovery method.

A large number of database technology/architecture have been designed in aid of analyzing those stored data and provide support for making decisions. However, for extraction of meaningful and relevant knowledge, which is simply not available by only querying the system from the datasets, an in-depth analysis of data is needed. Data Mining is an established important data analysis [1, 2] technique used for discovering interesting knowledge from huge amount of data in search of consistent patterns and/or systematic relationships between the variables. Several other research areas such as neural network [3], evolutionary algorithms [4], decision trees [5], support vector machines [6] and Bayesian methods [7] have been developed to address the problem and discover the hidden patterns [8] automatically.

Data mining is an iterative process that typically involves phases like problem definition, data exploration, data preparation, modeling, evaluation and finally deployment of the results. By performing data mining operations interesting knowledge, regularities and high-level information are extracted from databases which can be applied to decision making, process control, information management and query processing.

So, data mining is visualized as hybridization of techniques that develops promising interdisciplinary research and interesting findings by analyzing voluminous data. In real world, it is highly susceptible that the databases of huge size contain noisy, missing, and inconsistent data [9]. As a first step of data mining, data preprocessing techniques [10, 11] play an important role to improve the efficiency and ease of the mining process. There are a number of data preprocessing techniques, which have been used as and when necessary.

This chapter focuses on the characteristics of feature selection techniques [12, 13] and classification [14, 15] methods and their hybridization technique explored in the research work in order to discover the hidden patterns [8] in the dataset both in static and dynamic environment. Feature selection [12] of huge volume of static and dynamic data and at the same time preserving important features [13] of the dataset and how it influences the designing of classifiers has been analyzed in this chapter.

The need of ensemble of classifiers [16] towards achieving more accuracy has been discussed here. The book is presenting to develop an intelligent, efficient and optimized data analysis [1, 2] and management system that tackles the limitations of the existing systems by involving rough set theory [17-20], graph theory [21], clustering algorithm [22], evolutionary algorithms [4] such as genetic algorithm (GA) [23] and particle swarm optimization (PSO) [24] algorithm and different mathematical and statistical approaches [25, 26] to knowledge discovery. Rough set theory, a data mining tool based on the mathematical concept, is used for feature selection for both the static and dynamic environment in the thesis by generating minimal subsets of attributes called reducts. From the reducts, all possible rules are obtained, which are filtered using different optimized and statistical approaches and thus a competent set of rules [14] are generated for classifying the data objects. As single classifier always does not give the best result, so an ensemble classification technique is also proposed for the static environment using rough set theory and GA. In the book, an integrated approach using rough set theory and GA is used for feature selection in dynamic environment by generating optimal reducts in incremental way. An incremental classification model using PSO algorithm is also devised in the thesis. This chapter explains the major contributions made in this thesis. At the end, it presents an overview of the organization of the book.

1.1 Scope of the Book:

The book focuses on following data mining techniques for feature selection [12, 13] and classifier construction [14, 15] applied on various experimental benchmark datasets [27, 28] for knowledge extraction in an efficient way both for static and dynamic environment.

1.1.1 Feature Selection:

Feature selection [12, 13] is frequently used as a pre-processing step to data mining and knowledge discovery. Pre-processing is the task of diagnosing and correcting or removing damaged, corrupt, or inaccurate data from a dataset. As a first step of data mining, data pre-processing techniques [10, 11] play an important role to improve the efficiency and ease of the mining process.

There are a number of data pre-processing schemes [10, 11], which have been used in the proposed work as and when necessary. Several feature selection methods are proposed for the selection of important features from the dataset. It includes the following issues. Normalization [29], discretization [30] and concept hierarchy generation [31] are different types of data transformation technique and each contributes towards the accomplishment of the effective data mining process.

Data normalization is to scale the data to fit within a smaller range which can improve the accuracy and efficiency of mining process involving distance measurements. Data discretization is a pre-processing step where the raw values of a numeric attribute are replaced by interval labels or conceptual labels to achieve more generalized information. In pre-processing step, the missing values are handled, noises are smoothed and subsequently the data are transformed into a compact, concise, and more generalized form so that the patterns found as a result of classification algorithms may be more efficient and easier to understand.

In recent years, dimension of datasets is growing rapidly in many applications which bring great difficulty to data mining [1] and pattern recognition [8]. Feature selection [12, 13] is regularly used as a pre-processing step to data mining and knowledge discovery. It selects an optimal subset of features from the feature space according to a certain evaluation criterion. Also, all the measured variables of these high-dimensional datasets are not relevant for understanding the underlying phenomena of interest. This enormity may cause serious problems to many machine learning algorithms with respect to scalability and learning performance. Therefore, it is very effective for removing irrelevant and redundant features, increasing efficiency in data analysis like clustering [22] and classification techniques [14, 15].

A key objective of machine learning [32] research is the dimension reduction of both the static and dynamic dataset for relevant feature selection applied prior to extract interesting rules and patterns from the large repository of data. Feature selection methods [12, 13] select a minimal subset of K features from the original set of D features ($K \leq D$), so that the feature space is optimally reduced. It improves the performance of the learning algorithms, reduces the computational cost, and provides better understanding of the datasets. As a result, efficiency and acceptability of the system increases.

As datasets changes with time, it is very time consuming or even infeasible to run repeatedly the same feature selection algorithm on whole dataset. Incremental learning [33] is a technique where the learning process occurs on new data comes incrementally together with the existing data. As an application of incremental learning, the field of agriculture is considered in the thesis and incremental feature selection algorithm is devised integrating rough set theory and genetic algorithm to select important features of rice disease from rice disease dataset [34], which may change over time.

1.1.2 Classification Analysis:

Classification [14, 15] is a data mining technique used to predict class label of objects in unseen dataset based on the training provided earlier to classify the instances. Classification is a supervised learning method, where number of class labels and their values are known a priori.

Classification and prediction are two forms of data analysis methods to building models describing important data classes or to predict future data trends.

a. Classification Algorithm:

Data classification [14, 15] is a two-step process. In the first step, a model (or classifier) is built describing a predetermined set of data classes or concepts. The model needs to validate and finally used for classification of new datasets. Prediction can be viewed as the use of a model to assess the class of an unlabelled sample, or to assess the range of values of an attribute within which a given sample may belong. In this perspective, classification, and regression [15] are used to deal two major types of prediction problems. Classification is used to predict unseen objects with discrete values while regression is used for continuous or ordered values.

There are various classification methods [14] but none of them can efficiently handle voluminous datasets. Building efficient classifier to extract meaningful knowledge from the huge amount of data is the primary concern to the data mining research community. A promising technique is Rough Set Theory (RST) [17-20] which is used to analyze voluminous data, based on the concept of granular objects where each granule represents similar classes.

It is a new mathematical tool used to handle vagueness in data without any prior information unlike fuzzy set theory [35] or probability theory [36]. The model generated by the learning algorithm should both fit the input data well and correctly predict the class labels of samples. For example, decision trees [5] represent the knowledge in a tree structure, instance-based algorithms, such as nearest neighbour [37], use the instances to represent what is learned, Bayes method [7] represents the knowledge in the form of probabilistic approach and so on.

b. Classification Evaluation:

Evaluation is the major issue to measure the success of classification model in data mining field. To evaluate performance of classification algorithms [37], one way is to divide samples into two sets, training samples and test samples. Training samples are used to build a learning model while test samples are used to judge the performance of the classifier. The test samples are supplied to the model, having their unknown class labels, and then their predicted class labels assigned by the model are compared with their corresponding original class labels to calculate prediction accuracy. Holdout and cross-validation are most common techniques [37] for validation of classifiers and performance metrics such as sensitivity, specificity, precision, recall, and F-Measure [38] are often used to evaluate the performance of the classifier. Some statistical measures like Wilcoxon's Rank sum test [39], t-test [40], chisquare test [41] etc. are also very useful to check if the generated models are statistically significant or not.

1.2 Book Contribution:

Researchers are working on multi-dimensional dataset for performing different data analysis jobs. However, most of them follow a systematic approach and does some predetermined activities while analyzing dataset using pre-processing [10-11] and classification techniques [14, 15]. The book establishes the fact that an integrated approach towards performance-based analysis is one of the necessities for designing efficient method to discover new knowledge. In order to satisfy the above requirements, the book aims at contributing towards development of a hybrid framework by building performance-based preprocessing, and classification method using rough set theory, graph theory, statistical approach and evolutionary techniques to extract important and meaningful knowledge from the standard dataset.

The aim of the book is to exploit data mining techniques in both the static and dynamic environment for feature selection and classification problems, such as (i) identifying the most relevant features (ii) select significant feature subset (iii) classifier construction and (iv) ensemble classifier design for classification.

The contributions made in the book are summarized below.

1.2.1 Feature Selection in Static Environment:

Feature selection is become very necessary job for data analysis when facing high dimensional data. Rough Set Theory (RST) [17-20], a new mathematical approach to imperfect knowledge, is popularly employed to evaluate significance of features and helped to find out the optimal number of sufficient features called reduct. The main advantage of rough set theory in data analysis is that it does not need any preliminary or additional information about data like probability in statistics [36], basic probability assignment in Dempster-Shafer theory [42], grade of membership or the value of possibility in fuzzy set theory [35] and so on. But finding reduct by exhaustive search of all possible combinations of features is an NP-Complete problem and so some heuristic approaches need to be applied.

a. A novel single reduct generation method (SRG) [43] has been presented in the book for selection of important features or attributes from the decision system. Method proposes a new kind of indiscernibility relation called relative indiscernibility based on rough set theory [17-20]. In SRG, relative indiscernibility gives relatively indiscernible objects based on an attribute, relative to decision attribute. Here, relative indiscernibility relation induces partitions of objects from which degree of similarity or similarity factor between two attributes is measured and an attribute similarity (ASS) set is obtained. The similarities with similarity factor less than average similarity value are removed from ASS and an attribute similarity table is constructed. From the table, the attribute similar to maximum number of attributes is selected so that the resultant minimum set of selected attributes (called reduct) cover all attributes of the attribute similarity table. It is observed that average accuracy by SRG method is much higher than that by some existing standard feature selection technique.

b. In the book, a graph based reduct generation method (GRG) [44] is proposed by combining the concept of relative indiscernibility relation [17-20] of RST and Minimal Spanning Tree (MST) [45] of graph theory [21]. Similar to SRG method, relative indiscernibility relation induces partitions of objects from which degree of similarity or similarity factor between two attributes is measured and an attribute similarity (ASS) set is obtained. Attribute similarities of ASS with similarity factor less than average similarity value are removed and a directed weighted graph is constructed based on the reduced ASS set, where weight of an edge is the inverse of the corresponding similarity factor. Then a minimal spanning tree is obtained from the directed graph using Chu-Liu/Edmonds algorithm [46]. The tree represents all important similarities of attributes by its edges which help to find out all the information-rich attributes. To generate reduct, a root (which has no incoming edge) of the spanning tree is selected first and all its outgoing edges are removed. Then another vertex of the maximum out-degree is selected and associated outgoing edges are removed. This process continues until the edge set of the tree becomes empty and all the selected vertices form a reduct. The results show that the new method is good enough and often gives better accuracy than some existing methods in most of the cases.

c. In the book, a novel heuristic approach for the generation of multiple reduct set (FSBR) [47] is proposed. The method computes the compact reduct set based on the concepts of discernibility relation and attribute dependency of the rough set theory [17-20]. Firstly, a discernibility matrix is constructed from the decision system based on which the core and noncore attributes are identified. Then, rank of all noncore attributes is calculated from their frequency in the discernibility matrix.

The heuristic of this method is that the attribute with higher rank attribute is more important than the lower ranked attributes. The higher ranked noncore attribute is added to the core in each iteration provided attribute dependency of the resultant set increases and subsequently a reduct (final resultant set) is formed after certain iteration when dependency of the decision attribute on the resultant set is equal to that of the decision attribute on the whole condition attribute set.

The same process is repeated with the core and remaining noncore attributes for generating other reducts using forward selection method. Then an efficient backward removal process is applied to generate a compact set of reducts removing irrelevant attributes selected during forward selection. The experimental result shows that, the accuracy given by various classifiers is also comparable with that of the popular existing methods.

d. A multiple reduct generation method (MRG) [48] is proposed using the concept of Rough Set Theory [17-20] and clustering algorithm [22] to select the multiple feature subset from a decision system.

Here, projection of dataset based on two conditional attributes such as C_i and C_j is taken and *K*-means or *K*-prototype clustering algorithm is applied on it based on the nature of the dataset, where K = number of distinct values of decision attribute set D of the dataset to obtain K clusters. Also, the dataset is clustered into K-groups using Indiscernibility relation applied on the decision attribute set D. Then the connecting factor k of combined conditional attributes (C_iC_j) with respect to D is calculated using two cluster sets and attribute connecting set $ACS = \{(C_iC_j \overset{k}{\rightarrow} D)$ for all $C_i, C_j \in C$, Conditional attribute set, and D (Decision attribute set)$\}$ is formed. Each element $(C_iC_j \overset{k}{\rightarrow} D) \in ACS$ implies that C_i and C_j connecting together partition the objects that yields (k*100) % similar partitions yield by D. Then an undirected weighted graph with weights as the connecting factor k is constructed using attribute connecting set *ACS*. Finally based on the weight associated with edges, and the degree of the vertex, the reducts are generated. The results show that the MRG method is good enough and often gives better accuracy than some existing methods in most of the cases. A comparative study of all the proposed feature selection methods [43, 44, 47, 48] is performed to demonstrate their merits and demerits so that it will be easier to decide which one is better for designing classifier for class label prediction of unknown objects in the datasets. Statistical analysis is done to measure the significance of all the proposed methods in comparison with other competitive methods.

1.2.2 Feature Selection in Dynamic Environment:

a. A novel dynamic reduct generation technique (DRED) [49] has been discussed in the book for generation of multiple reduct for incremental data where new data are added continuously to the existing data. The proposed method discovers the knowledge from this incremental data [50-56]. The method analyses the new dataset, when it becomes available, and modifies the reduct accordingly to fit the entire dataset. The concepts of discernibility relation and attribute dependency of Rough Set Theory [17-20] are used for the generation of dynamic reduct set. The DRED method is applied on the static dataset after converting it into incremental dataset, the original decision system is divided into two subsystems such as old and new subsystems.

When the algorithm is first run for the initial subsystem, no previous reduct information is available; so FSBR algorithm [47] is applied on old subsystem to generate a set of reducts for the old dataset. Subsequently, when newly arrived decision subsystem is become available then the previous reduct set with the new subsystem determines a set of dynamic reducts of the whole system using DRED algorithm. The method is compared with various state-of-the-art methods to demonstrate the effectiveness of the proposed method.

b. A genetic algorithm (GA) based group incremental feature selection method (IFS) [57] is also discussed in the book, where the method selects the features dynamically using the concept of rough set theory and the genetic algorithm. Here GA [23] is applied only on newly added group of objects of small to moderate size on regular basis so the great issue of using it for its larger complexity may be optimized in most of the applications. The novelty of the IFS algorithm is that it can select features both in static and dynamic environment and no prior statistical information of the data is required.

The method generates a population of size M randomly, where the length of each binary chromosome is $|A| = N$ (Total number of features). Let $A = \{B_1, B_2, B_3, \ldots, B_N\}$, and the i-th bit of a chromosome *ch* corresponds to attribute B_i. All '1's' in chromosome *ch* correspond to an attribute subset, i.e., $A_1 \subseteq A$. To check if A_1 is a reduct of the new subsystem, the fitness value of the chromosome *ch* is computed.

The fitness function of GA [23] is defined using the concept of positive region overlap in rough set theory for new group of added data and the reduct obtained from the old existing data. The fitness function determines the quality of a solution in the population and thus, a strong fitness function is imperative for obtaining good results. The algorithm has been applied on experimental benchmark datasets to demonstrate its effectiveness.

A comparative analysis of both the proposed incremental feature selection methods [49, 57] with other existing non-incremental and incremental method is performed to show the efficacy of the proposed methods. Statistical analysis is done to measure the significance of both the proposed incremental methods in comparison with other competitive methods.

1.2.3 Classifier Construction:

Analysis of huge data is utmost important though development of new mechanisms to handling the data lag behind with the enormous growth, resulting a huge volume of data retained without being studied. Modeling static and time variant or dynamic data for classification plays an important role in data mining. In the thesis, several different classification models for both the static and dynamic environment have been presented to classify the dataset.

a. In the book, a classification model (CGRG) [58] for the static data is discussed based on a single feature subset obtained by the feature subset selection method GRG [43]. Based on these selected features, classification rules are generated by constructing decision matrix [59], a concept of rough set theory [17-20]. The algorithm has been applied on benchmark experimental datasets and compared with various existing classification algorithms to demonstrate the effectiveness of the constructed classification model.

b. In the book, a novel ensemble classifier system (ECS) [60] is also constructed. Here, an ensemble classification algorithm has been designed and discussed for construction of an optimal ensemble classification system (ECS) using the concept of Rough Set Theory (RST) [17-20] and Genetic Algorithm (GA) [23]. In the first phase, a best performing feature selection algorithm is used to select the important features from the decision system. Here, multiple reduct generation algorithm (MRG) based on the concept of Rough Set Theory and clustering algorithm, is used to select only the important feature subset called reducts of the dataset. Now, from each reduct, rule-based classifier is constructed using the concept of association rule mining [61, 62]. In this way, base classifier models, one for each reduct are generated. In the second phase, base classifiers are fused, and an optimal ensemble classifier system (ECS) is developed using GA with a suitable fitness function having the objective to maximize the classification accuracy of the ensemble classification system. Performance of the classifier is measured to express its effectiveness. Here, combination of the best performing classifiers performs better compared to a single one, as objects which are not classified by one classifier may be classified by another classifier. The ECS method can use any number of classifiers.

c. In the book, a classifier for incremental data has been constructed (IPSO) [63] with an objective to develop a rule based incremental classifier [64-66] for the incremental datasets. In the developed method, the incremental classifier is designed with the aim that the number of classification rules will be optimal. In the proposed incremental classifier, optimized classification rules are generated for the incremental data dynamically using the concept of association rule mining and Particle Swarm Optimization algorithm (PSO) algorithm with a novel fitness function. The algorithm handles incremental data effectively by modifying the existing knowledge base whenever new data are available. In the method, firstly PSO [67-69] based training process is performed on the existing dataset to find out the initial optimal classification rules for existing dataset. When a new group of data arrives, incremental PSO (IPSO) is run using existing classifier and new group of data to develop a dynamic classifier. So, the proposed IPSO algorithm analyzes the new dataset in every interval of time and updates the previous knowledge base dynamically with a reduced training time. To judge the performance of the incremental classifier IPSO, proposed method has been applied on experimental benchmark datasets and compared with various state of the art non incremental and incremental classification algorithms.

In classification system, each of the categories has their own learning strategies to classify the instances. A comparative study and performance analysis in terms of statistical measures of all these classification methods and existing state of the arts methods are presented in the book. Statistical analysis is done to measure the significance of all the developed static and incremental classification methods in comparison with other existing competitive methods.

1.2.4 Application of the Data Mining Methods in the Field of Agriculture:

Application of data mining technology in agricultural field for disease prediction is a challenging task due to the wide variation of crops, associated diseases, and dependency on human being to collect information from the field. In this chapter of the book, developed feature selection and classifier construction methods have been applied in rice images to predict different rice plant diseases.

The incremental algorithms are very useful in this field as day-by-day the characteristics of the diseases change with the time due to changes of climate, biological, and geographical factor. In this dynamic environment new disease data are added with the existing data, so to predict the rice diseases in this dynamic environment, an efficient incremental automated intelligent system is necessary. Opinion of experts processing and analyzing of information to extract knowledge are the key issues addressed in this chapter of the book for developing automated cost-effective rice disease classification system.

Here, rice disease dataset [34] is considered to select important features in incremental way for disease classification. For the rice disease dataset, features based on color, shape, position, and texture are extracted [34] from the infected rice plant images [34]. However, it is observed that classification accuracy to detect diseases is not proportional to the number of features, rather a smaller number of features but more significant ones. Therefore, feature selection is an important step for accurately classifying rice diseases based on the extracted features. So, the best feature selection methods in static and dynamic environment have been used for feature selection by removing irrelevant and redundant features for classification of rice diseases. Then the reduced dataset with the selected features is fed into the developed classifiers to generate the important classification rules in static as well as in dynamic environment to predict the different rice diseases to take the precautionary measures at an early stage to protect the crop as well as to provide help to the farmers.

1.3 Organization of the Book:

The chapters of the book are organized as follows. Overview of the feature selection and classification analysis is presented in *Chapter 2.* In particular, this chapter provides a thorough study of the mostly used feature selection and classification algorithms. *Chapter 3* presents the static feature selection methods for identification of important feature subset using Rough Set Theory, graph theory, clustering algorithm, and different mathematical and statistical approaches and applied to different benchmark datasets to achieve the optimum set of features. *Chapter 4* presents the incremental feature selection methods for identifying the important feature subset in dynamic environment using Rough Set Theory and evolutionary algorithms.

The work on feature selection methods both in static and dynamic environment provides a through comparative study of all the proposed algorithms based on the experimental results to determine the best method, which ultimately is used for constructing the classifier. *Chapter* 5 presents the construction of classification systems in static and dynamic environment based on the reduced datasets. Single classifier is not always a very good predictor for all the problems, so the ensemble of classifier takes an important role for that purpose. The best combination of classifiers by fusing the different base classifiers integrating genetic algorithm and rough set theory is presented in this chapter. A novel incremental classifier designing method is proposed for the dynamic environment where data are gradually available with the varied time. *Chapter 6* discusses the application of both the static and incremental feature selection methods and classification algorithm in the agricultural domain for predicting diseases of rice crops. *Chapter 7* concludes the summary of the work along with its limitations and future scopes.

ISBN: 978-93-90847-37-2

Chapter 2

Data Mining Tools and Techniques- Overview and Concept

2.1 Introduction:

In this era of big data, every real-world dataset contains huge number of attributes and objects. Processing and analyzing of such high dimensional dataset are a challenging task for traditional machine learning system. Rapid advancement of database and knowledge discovery tools and technologies plays a fundamental role for intelligently and automatically transform the processed data into useful information and knowledge for data mining and pattern recognition. The data mining process mainly consists of data preprocessing (such as missing value estimation, data dimension reduction and feature selection), pattern generation, knowledge prediction, and its interpretation.

Many datasets have missing data values due to the error in manual data entry processes, measurements, equipments etc. Due to missing values in the dataset, it becomes difficult to handle and analyze the dataset without any biasness and thus efficiency of the system degrades.

Many existing techniques are available to deal with such missing values. Missing values are handled by different estimation methods [70, 71, 72, 73, 74] including missing value imputation by zero (0) value, most common value, mean or median and data mining algorithms like k-nearest neighbor [37, 74,], neural networks [3, 74], and association rules [61] etc.

Dimension reduction and feature selection [12, 13, 75-78] play a vital role in research interests in many application domains in recent years. Real world dataset contains large number of attributes and objects. Learning algorithm gives poor performance when these huge datasets are given as input to it for proper analysis. So, from these huge dataset most useful attributes/features need to be extracted for better understanding of the data with lesser computational complexity.

Dimension reduction solves this problem by removing the irrelevant, redundant, and noisy features. The aim of feature selection is to find a minimum set of relevant features that preserves all the essential information of the system and contribute the maximum to the decision system. Feature selection [12, 13, 75 - 78] has been widely used in many progressive research areas such as bioinformatics, agriculture, social network, image, and signal processing and so on.

Clustering and classification are the two most commonly used data mining technology to analyze the dataset. The partitions or grouping of objects into different categories is the subject of cluster analysis.

Cluster analysis [22, 79] searches the structures in data and classifies these structures into categories such that the degree of association is high among structures of same category and low between structures of different categories. Cluster analysis plays an important role for understanding various phenomena underneath in the datasets by applying either hard or soft partitioning of data. Classification analysis [15, 80] like cluster analysis [22, 79] plays an important role for understanding the various intrinsic properties hidden in the datasets.

Classification is a form of data analysis used in decision making by extracting data models from the huge repositories. Such models describe important data classes and provide a better understanding of data. Classification is a process used to predict future data trends by categorical (discrete, unordered) labels. Recent data mining techniques [2] has developed several scalable classification techniques [81, 82] capable of handling large amount of data

In the chapter, initially description of experimental datasets together with the most important data preprocessing activity such as different feature selection techniques related to the work is reviewed. Then for analysis of the data, different clustering algorithms, classification algorithms and classifier validation method based on statistics, computer science and machine learning are reviewed.

2.2 Experimental Dataset Description:

The book focuses on analysis of different benchmark datasets and one simulated dataset using data mining techniques [1, 2], such as feature selection [12, 13], and classification [14, 15] of objects with an aim of making researchers aware of the benefits of such techniques when analyzing these dataset [27, 28, 34].

In the experiment, the datasets $M_{n\times m}$ are represented as decision system and expressed in the form of a matrix according to which each row represents an object, and each column represents an attribute or feature. The datasets consist of two types of attributes namely conditional attributes and decision attribute.

Each object is characterized by a class label represents the decision value. The used experimental datasets conform to the standard data format of machine learning algorithm and data mining. The ten popular benchmark datasets and one simulated rice disease dataset is used in the thesis for experiment purpose are summarized below.

2.2.1 Wine Dataset:

Raw Data: The raw data is available in UCI ML repository [27].
Description: The dataset consists of 13 conditional attributes, 1 decision attribute with 3 decision classes and 178 instances. All attributes are continuous.

2.2.2 Heart Dataset:

Raw Data: The raw data is available in UCI ML repository [27].
Description: The dataset consists of 13 conditional attributes, 1 decision attribute with 2 classes and 270 instances. Attributes are categorical, integer, real in nature.

2.2.3 Glass Dataset:

Raw Data: The raw dataset is available in UCI ML repository [27].

Description: The dataset consists of 9 attributes, 1 decision attribute with 6 decision classes and 214 instances. All attributes are continuous.

2.2.4 Zoo Dataset:

Raw Data: The raw data is available in UCI ML repository [27].
Description: The dataset consists of 16 conditional attributes, 1 decision attribute with 7 classes and 101 instances. Attributes are real in nature.

2.2.5 Dermatology Dataset:

Raw Data: The raw data available in UCI ML repository [27].
Description: The dataset consists of 33 attributes 1 decision attribute with 6 classes and 366 instances. Attributes are real in nature.

2.2.6 Mushroom Dataset:

Raw Data: The raw data available in UCI ML repository [27].
Description: The dataset consists of 21 conditional attributes, 1 decision attribute with 2 classes and 5644 instances. All attributes are categorical in nature.

2.2.7 Coil20 Dataset:

Raw Data: The raw data available in scikit-feature feature selection repository [28].
Description: The dataset consists of 1024 conditional attributes, 1 decision attribute with 20 classes and 1440 instances. Attributes are real in nature.

2.2.8 Orl Dataset:

Raw Data: The raw data available in scikit-feature feature selection repository [28].
Description: The dataset consists of 1024 conditional attributes, 1 decision attribute with 40 classes and 400 instances. Attributes are real in nature.

2.2.9 Allaml Dataset:

Raw Data: The raw data available in scikit-feature feature selection repository [28].
Description: The dataset consists of 7129 conditional attributes, 1 decision attribute with 2 classes and 72 instances. Attributes are real in nature.

2.2.10 Leukemia Dataset:

Raw Data: The raw data available in scikit-feature feature selection repository [28].

Description: The dataset consists of 7070 conditional attributes, 1 decision attribute with 2 classes and 72 instances. Attributes are discrete in nature.

2.2.11 Rice Disease Dataset:

Raw Data: The raw data is available in [34].
Description: The dataset consists of 37 conditional attributes, 1 decision attribute with 3 classes and 500 instances. Attributes are continuous in nature.

2.3 Feature Selection:

The complexity of any classifier depends on the number of the given inputs. This determines both the time and space complexity and the necessary number of training examples to train such a classifier. So, dimensionality of the problem needs to be reduced. Decreasing dimension also decreases the complexity of the inference algorithm during testing. When data can be explained with fewer features, a better idea about the process that underlies the data can be achieved and this allows knowledge extraction. When data can be represented in a few dimensions without loss of information, it can be plotted and analyzed visually for structure and outliers. Recently, an increasing number of applications in different fields produces massive volumes of very high dimensional data [83, 84, 85] under a variety of experimental conditions, which cause trouble in clustering and classification. Dimensionality reduction and feature selection [12, 13, 75-78] is an important prepossessing step before clustering and classification. The reduced feature set should have the same characteristics as the entire feature set in the system.

2.3.1 Issues Regarding Feature Selection:

Feature selection algorithms [12, 13, 75-78, 86] perform a search through the space of feature subsets, and address mainly three basic issues affecting the nature of the selection [86]. The general feature selection method is shown in Figure 2.1.

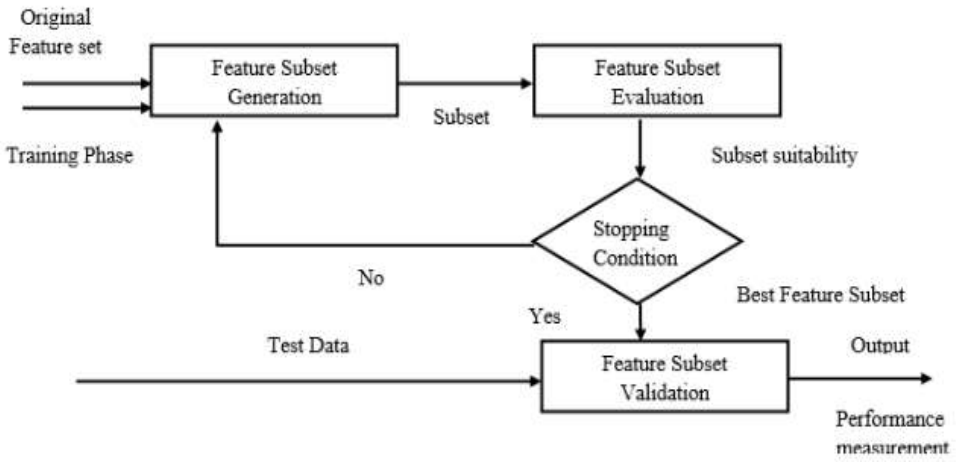


Figure 2.1: General feature selection process

In this stage, a search method is followed to generate a subset of features for evaluation. Searching can be done in forward direction where search process starts with no features and iteratively add another feature.

On the other hand, Searching can be done in backward direction where search process starts with all features and iteratively remove them. Another alternative can be there where search process starts somewhere in the middle and move both direction for the generation of feature subset.

An exhaustive search in the feature subspace is exorbitant for all but a little number of initial features. In the exhaustive search, with an initial feature there exist 2^n possible feature subsets. So heuristic search approaches are more practical than exhaustive ones and often give good results, but there is no guarantee for obtaining the optimal feature subset.

a. Evaluation Strategy:

Suitability of a feature subset is evaluated by the evaluation function of the feature selection algorithms for machine learning. In the *filter* [86] method without involvement of any learning algorithm, irrelevant features are filtered out from the dataset before learning begins. These algorithms [86] use heuristics based on general properties of the dataset to evaluate the goodness of feature subsets. In the *wrapper* [86], method an induction algorithm along with a statistical re-sampling approach is used to estimate the final accuracy of the feature subsets.

b. Stopping Criterion:

A stopping criterion is always checked in each iteration regarding the termination of the search process. According to the evaluation criteria, a feature selection algorithm might stop adding or deleting features when none of the candidate feature subset improves upon the goodness of a current feature subset. The algorithm continues to modify the feature subset without degrading the quality.

2.3.2 Feature Selection Methods:

Naturally a feature selection method will employ on a given training dataset in order to make a decision about which feature subset to be selected. There are so many feature selection methods available in the literature [86]. These are discussed in the following sections.

a. Filter Method:

The filter method selects the best feature subset using a feature ranking function which calculates the relevance score of each feature. Naturally, the features having higher rank are more relevant. When the conclusion variable is binary and the features are numeric in nature, usual ranking functions are the signal-to-noise ratio (SNR) [87] for each feature, defined in Equation (2.1) and the Fisher discriminant ratio (FDR) [88] given for each feature defined in Equation (2.2).

$$SNR_i = \frac{|\mu_1 - \mu_2|}{\sigma_1 + \sigma_2} \quad (2.1)$$

$$FDR_i = \frac{(\mu_1 - \mu_2)^2}{{\sigma_1}^2 + {\sigma_2}^2} \quad (2.2)$$

Where μ_1 and μ_2 are the mean of the feature for class 1 and class 2, respectively and σ_1 and σ_2 are the standard deviation for class 1 and class 2, respectively.

The Figure 2.2 shows the general filter method for feature selection.

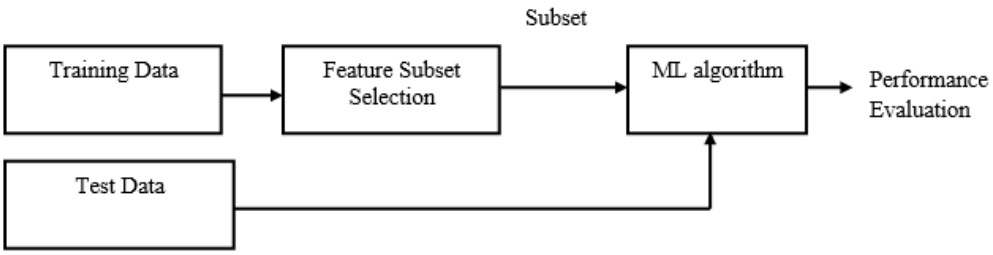


Figure 2.2: Filter approach for feature selection

The filter approach is successful in most of the applications but disadvantage of choosing this method is that features are evaluated independently, no correlation between the features are considered. But it may happen that individually two features are ineffective in classification but the combination of those two features is effective and useful [89].

b. Wrapper Method:

In the wrapper method [86, 90, 91], feature subsets are selected by the searching of the whole of feature subsets and test performance of every subset is done with the direct involvement of the learning algorithm.

The feature subset that provides the best performance is selected for final use. Clearly, if there are n features in total, 2^n possible subsets to be searched. Such an exhaustive search is impractical, so most of the wrapper algorithms incorporate a heuristic function to reduce the search space.

This procedure involves either forward selection, addition of features one at a time or backward selection, removing of features one at a time, until some stopping condition is achieved. Furthermore, a bidirectional selection method is also available that involves addition or deletion of a feature at every step. The Figure 2.3 shows the general feature selection technique by wrapper method.

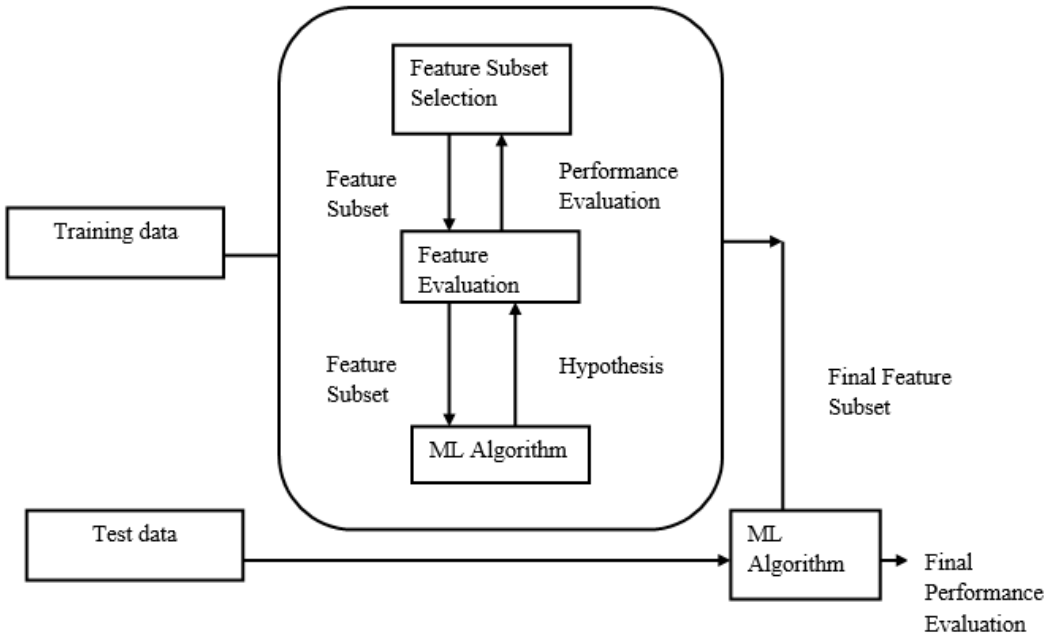


Figure 2.3: Wrapper approach for feature selection

The application of wrapper method in machine learning is relatively current [92]. The disadvantage of the wrapper method is that it tends to be computationally exhaustive and the use of a heuristic function to improve the search space can be ad hoc.

Generally, the filter methods outperform in terms of prediction accuracy, but are relatively more computationally expensive.

Wrappers often achieve better results than filters due to the fact that they are tuned to the definite dealings between an induction algorithm and its training data.

c. Principal Component Analysis:

Principal Component Analysis (PCA) [93, 94] is a popular statistical data pre-processing method that selects features as principal components by transforming the original high dimensional features into lesser number of uncorrelated features.

PCA finds principal components that are linear combinations of original features such that they are orthogonal to each other and capture maximum amount of variance in the data. Generally, it is possible to capture high variance using only a small number of principal components.

In order to find principal components, covariance matrix of original data is calculated and all the eigen values are derived. Eigenvectors those are associated with the largest eigen values are selected as principal components.

Though PCA is a popular dimension reduction method used in many applications, but now a day's non-linear dimension reduction methods are also getting popularity due to certain limitation of PCA. PCA suffers from some of the following limitations.

- It considers linear relationships between variables. PCA is less efficient for bioinformatics datasets where data have inherently non-linear structure.
- Its interpretation is valid if all of the variables are thought to be scaled at the numeric value.
- It does not consider probabilistic model organization which is significant in many contexts such as mixture modelling and Bayesian decision.
- If PCA tries to transform non-linear structure into low dimensional space, most of the structure information is lost due to the use of linear distance measures like Euclidean and Manhattan distance.

b. Correlated Feature Subset Selection:

The importance of a feature or a feature subset is judged by the parameters such as its *redundancy* and *relevancy*. A feature is redundant if it is very much correlated with other features. So, the redundancies can be detected by the correlation analysis. Considering any two features, correlation analysis can measure how strongly one feature is related to the other, based on the available information.

Correlation between two features can be computed by calculating correlation coefficient for numerical datasets. On the other hand, a relevant feature is always predictive of the decision feature, otherwise features are considered as irrelevant. So objective is to find out a good feature subset consists of the features that are uncorrelated with each other and extremely correlated with decision feature then it will be regarded as a good feature for the classification task. In this logic, the problem of feature selection needs a suitable measure of finding the correlations between features and a novel method to select features based on this measure. There exist many methods [95, 96] to evaluate the correlation between two arbitrary variables. One is based on classical linear correlation coefficient measure, defined in Equation (2.3).

$$R = \frac{n(\sum a \times b) - (\sum a)(\sum b)}{\sqrt{\left(n\sum a^2 - (\sum a)^2\right)\left(n\sum b^2 - (\sum b)^2\right)}} \qquad (2.3)$$

Where, a and b are the two relative features, n is the total number of features, and the value of R is in the range $-1 < R < +1$.

c. Information Gain based Feature Selection:

The method removes the redundant and irrelevant features from the feature space selecting suitable and relevant features of the dataset. It brings the direct effects on speeding up of data-mining algorithm, improving the dataset quality and the performance of data mining process.

There are many features evaluation functions [97] such as information gain, gain ratio, symmetrical uncertainty, relief-F, one-R and chi-squared.

Entropy [97] is a commonly used metric in the information theory, which characterizes the purity of an arbitrary collection. It is in the foundation of Information gain attribute ranking methods. The entropy measure [97] is considered as a measure of irregularity of system. The entropy of the attribute Y is defined in Equation (2.4).

$$H(Y) = -\sum_{y \in Y} p(y) \log_2(p(y)) \tag{2.4}$$

Where, *p(y)* is the probability density function for the random variable $y \in Y$. If the observed values of *Y* in the training dataset are partitioned according to the values of second feature *X*, and entropy of *Y* with respect to the partitions induced by *X* is less than the entropy of *Y* prior to partition, there is a relationship between the features *Y* and *X*. The entropy of *Y* after observing *X* is defined in Equation (2.5).

$$H(Y/X) = -\sum_{x \in X} p(x) \sum_{y \in Y} p(y/x) \log_2(p(y/x)) \tag{2.5}$$

Where, $p(y/x)$is conditional probability of *y* given *x*. Given the entropy as a criterion of impurity in a training set, additional information about *Y* provided by *X* is measured that represents the amount by which the entropy of *Y* decreases. This measure is known as information gain (*IG*), given by Equation (2.6).

$$IG = H(Y) - H(Y/X) = H(X) - H(X/Y) \tag{2.6}$$

IG is a symmetrical measure. The information gained about *Y* after viewing *X* is equal to the information gained about *X* after observing *Y*. A weakness of the information gained criterion is that it is influenced in support of features with more values even when they are not more informative.

d. Rough Set Theory based Feature Selection:

Rough Set Theory is a soft computing technique proposed by Z. Pawlak [17] for handling vague, inconsistent and uncertain data. Rough Set Theory is very useful technique to select important features from an information system in data mining field.

An information system can be represented as *I* = (*U, A*), where *U* is the universe of discourse with a finite number of instances or objects and *A* is the number of attributes defined on *U*.

The information system becomes a decision system when a decision attribute *D* is present in the system. Then the system can be represented as a decision system *DS* = (*U, A, D*), where *U* is the universe of discourse where *A* and *D* denotes the conditional attributes and the decision attributes respectively.

Rough set is defined in terms of a pair of sets, namely lower approximation, and upper approximation of the original set. Indiscernibility relations and set approximations are the fundamental concepts of the Rough Set Theory (RST) [17-20].

- **Indiscernibility Relation:** The indiscernibility relation can be described as an equivalence relation where two instances are equivalent if they are not discernible by their properties. Suppose the universe $U = \{x_1, x_2..., xn\}$ consists of n instances and for any subset of attributes, $K \subseteq A$, there is an associated K-indiscernibility relation $IND(K)$, given in equation (2.7).

$$IND(K) = \{(x, y) \in U^2 | \forall a \in K, a(x) = a(y)\} \quad (2.7)$$

If $(x, y) \in IND(K)$, then x and y are indiscernible with respect to attribute set K. $IND(K)$ actually partitions the set of instances into equivalence classes $[x]_K$. The instances in an equivalence class are indiscernible with respect to the attribute set K and any two instances of different equivalence classes are discernible with respect to K.

- **Lower and Upper Approximation:** The lower approximation of a target set X with respect to any subset of attributes, says $K \subseteq A$, is the set of all instances that certainly belong to X, defined mathematically in equation (2.8).

$$\underline{K}X = \{x | [x]_K \subseteq X\} \quad (2.8)$$

To determine the lower approximation of a target set X for the attribute subset K, U is partitioned into equivalence classes $[x]_K$ using $IND(K)$, given in equation (2.7). In the same way, equivalence classes $[x]_D$ is formed using equation (2.7) for the decision attribute set D. Let, $U/K = \{[x]_K / [x]_K$ is an equivalence class induced by $IND(K)\}$ and $U/D = \{[x]_D / [x]_D$ is an equivalence class induced by $IND(D)\}$ are the two partitions of instances in U. Now, each class $X \in U/D$ is considered as the target set. The lower approximation set $\underline{K}X$ with respect to K is computed using equation (2.8), whose elements are certainly members of U/K. The positive region $POS_K(D)$ is calculated by taking the union of the lower approximations $\underline{K}X$ under K for all targets set $X \in U/D$, given in equation (2.9).

$$POS_K(D) = \cup_{X \in U/D} \underline{K}X \quad (2.9)$$

The upper approximation of the target set X with respect to K is the set of all instances that can possibly belong to X, as defined in equation (2.10).

$$\overline{K}X = \{x | [x]_K \cap X \neq \emptyset\} \quad (2.10)$$

Boundary region of a target set X is defined as the region of uncertainty consists of those instances that neither be considered nor be ignored as the member of the target set X. Thus, the boundary region $\boldsymbol{BND_K(D)}$for all target sets of the decision system (actually, all target sets are elements of set U/D) is defined by the equation (2.11).

$$\boldsymbol{BND_K(D)} = \cup_{X \in U/D} \overline{K}X - \cup_{X \in U/D} \underline{K}X \quad (2.11)$$

Obviously, if $\boldsymbol{BND_K(D)}$ is empty, then the decision system is precisely defined; otherwise, there are some uncertainty or impreciseness in the system. Due to this impreciseness, there is some lack of information to fully characterize the decision system. To handle such impreciseness of the system, rough set theory is considered for many features selection method.

- **Attribute Dependency and Reduct:** In RST, the concept of attribute dependency is stated very clearly to determine which attributes are strongly related to which other attributes in the decision system. Let us consider, two disjoint attribute subsets, *K* and *Q* of *A* and find out the degree of dependency present between them. Each attribute subset *K* and *Q* induces two equivalence classes $[x]_K$ and $[x]_Q$ respectively. Then, the dependency of *Q* on *K* is denoted by $\gamma_K(Q)$ and is defined in equation (2.12).

$$\gamma_K(Q) = \frac{\sum_{i=1}^{N} |\underline{K}X_i|}{|U|} \tag{2.12}$$

Where, X_i is a class of instances in $[x]_Q \forall i = 1, 2, \ldots, N$.

The dependency value of *D* on *K* (i.e., $\gamma_K(D)$) which is in the range [0, 1], is calculated using equation (2.12). From the definition, it is clear that more objects in the positive region imply fewer objects in the boundary region and thus the dependency value increases. More dependency of the decision attribute with respect to an attribute subset implies that the attributes are more significant.

A reduct [17, 18] can be considered as a complete set of attributes represents the class structure of a decision system. By considering these attributes, the decision system has the same equivalence class structure as it is expressed by the full conditional attribute set *A*. Considering the equivalence classes **in set** *U*/*D* obtained using indiscernibility relation *IND* (*D*), as the target sets and *R* as the minimal attribute subset of *A*, *R* is called the reduct if it satisfies equation (2.13). In other words, *R* is a reduct if the dependency of *D* on *R* is exactly equal to that of *D* on *A*.

$$\gamma_R(D) = \gamma_A(D) \tag{2.13}$$

There may be multiple reducts present in a decision system, but each should preserve the equivalence-class structure expressed by the decision system. So, feature selection using RST is performed by selecting a reduct or a set of reducts from the original feature set by satisfying the above mentioned criteria.

- **Genetic Algorithm based Feature Selection:**

Genetic algorithm (GA) [23] is an adaptive heuristic search technique for finding global optimal solution. It simulates genetic and evolutionary process of natural evolution. It is first proposed by Professor Holland in the University of Michigan of the United States. Its search technique is not along a single direction of search space. It considers a number of individual solutions and tests for convergence within the overall scope of the search space, thus leading to a greater possibility of finding the global optimal solution.

It is very useful for solving optimization problems because of its robustness in the sense that it works fine even if the input parameters are slightly changed, or in the presence of reasonable noise. Also, the method offers significant benefits while searching for a solution in a large state-space, multi-modal state-space, or n-dimensional surface, over more typical search of optimization techniques like linear programming [98], depth-first [99], breath-first [100], praxis [101], and so on. The genetic algorithm-based optimization starts with a population of randomly generated chromosomes where each chromosome represents a candidate solution of the concrete problem being solved. In each generation, the fitness of each chromosome is evaluated, and the more fitted solutions are selected to form a mating pool. Two parents are randomly selected from the pool and undergo cycle crossover [23] and mutation [23] to form two offspring. This process of selection [23], crossover [23] and mutation [23] is repeated until the new population is generated. The new populations members are evaluated based on the fitness function [23] and participate for inclusion in the mating pool, and the process continues until either a predefined number of generations are completed, stagnation, or termination criteria are satisfied. The mutation operation is helpful to avoid premature convergence and to explore broader search space. Thus, the process searches for the better solutions in each generation and is continued until the population converges to a globally optimal solution in the solution space. Configurable parameters in the implementation include termination criterion, tournament size to select parents, crossover probability, and mutation probability.

The optimization problems generally have one or more feasible solutions obtained using one or more objective functions. There are two types of genetic algorithm, single objective genetic algorithm [102] and multi objective genetic algorithm [103]. In case of a single objective genetic algorithm [102], the real-world optimization problem is modeled involving only one objective function for finding the unique optimal solution. The main goal of using single objective genetic algorithm is to find the optimal solution, which corresponds to the optimum value of the single objective function. On the other hand, multi objective genetic algorithm [103] models the optimization problem involves more than one competing or conflicting objective functions for finding many optimal solutions. Most of the real-world optimization problems involve multiple objectives, and if they are conflicting in nature, there is no single optimal solution, rather a number of pareto optimal solutions [104]. Generally, all pareto optimal solutions are treated as equally good and the goal may be to find a representative set of pareto optimal solutions or finding a single solution by the decision maker based on the application. Thus, although the fundamental difference between these two optimization techniques lies in the cardinality in the set of optimal solutions, but in reality, if a user needs only one solution, no matter whether the associated optimization problem is single objective or multi objective.

- **Particle Swarm Optimization Algorithm based Feature Selection:**

Particle Swarm Optimization (PSO) [24] is an evolutionary optimization algorithm proposed by Kennedy and Eberhart in 1995. In PSO [24, 105-110], a population, called a swarm, of candidate solutions are encoded as particles in the search space. PSO starts with the random initialization of a population of particles. The whole swarm moves in the search space to search for the best solution by updating the position of each particle based on the experience of its own and its neighboring particles. In PSO, a potential solution to a problem is represented by a particle $X(i) = (x_{(i,1)}, x_{(i,2)}, \ldots, x_{(i,n)})$ in an n-dimensional search space.

The coordinate $x_{(i, d)}$ of the particle $X(i)$ have a rate of change of position i.e. the velocity $v_{(i,d)}$ where $d = 1, 2, ..., n$. Every particle keeps a record of the best position that it has ever visited. Such a record is called the particle's previous best position and denoted by B_i. The global best position G attained by any particle so far is also recorded. Iteration comprises evaluation of each particle with the adjustment of $v_{(i, d)}$ in the direction of particle $X(i)$'s previous best position and the previous best position of any particle in its neighborhood. The set of phases that govern PSO are evaluate, compare, and evolve. The evaluation phase measures how well each particle or candidate solution solves the problem. The comparison phase identifies the best particles and the evolve phase produces new particles based on some of the best particles previously found. These three phases are repeated until a given stopping criterion is matched. The objective of the method is to find the best particle, which gives the optimal solution of the problem. Important concepts in PSO are velocity and neighborhood topology. Each particle $X(i)$ is associated with a velocity vector. This velocity vector is updated at every generation. The updated velocity vector is then used to generate a new particle $X(i)$. The neighborhood topology defines how other particles in the swarm, such as $B(i)$ and G, interact with $X(i)$ to modify its respective velocity vector and consequently, its position as well.

2.4 Cluster Analysis:

Cluster analysis [22, 79] is one of the major data analysis techniques generally used in various real-life applications in the field of machine learning. Clustering is the grouping of objects where similar type of objects is placed in same group and dissimilar objects are placed in different groups.

Clustering method which generates high quality clusters with less inter-cluster similarity and more intra-cluster similarity is recognized as a good clustering method.

2.4.1 Concern Regarding Cluster Analysis:

The goal of data mining research is to develop efficient cluster analysis [22, 79] technique applied in large databases [27, 28, 111-113]. Active themes of data mining research [114, 115] focus on the scalability of clustering methods, effectiveness of the methods while clustering complex shapes, high-dimensional clustering techniques and methods of clustering applicable on heterogeneous datasets. Various issues that are related to clustering are discussed below.

a. **Scalability:**

Scalability, as a property of systems, is generally difficult to describe and in any particular case it is essential to define the specific requirements for scalability on those dimensions, which are deemed important.

Many clustering algorithms [22] perform well on small datasets but it is not so efficient when the algorithm is applied to large datasets. Clustering on a sample of large dataset [111-113] may lead to biased results; therefore, in such cases highly scalable clustering algorithms [116] are needed.

b. **Ability to Handle Different Types of Attributes:**

Many existing clustering algorithms are developed to cluster interval-based data only [117]. Clustering methods are needed for binary, categorical, and ordinal data or combination of these data types depending on the application domains [117, 118].

c. **Discovery of Clustering with Arbitrary Shape:**

Many clustering algorithms determine clusters based on Euclidean or Manhattan distance [119, 120] measures and which results spherical clusters with similar size and density. Although, a cluster can be of any shape, it is important to develop algorithms that can form clusters of arbitrary form.

d. **Domain Knowledge to Determine Input Parameters:**

For the cluster analysis, many clustering algorithm seeks input parameters (such as the number of cluster) and the clustering results are quite sensitive to those input parameters. But for datasets containing high-dimensional objects, it is very tough to decide the input parameters as well as to feed the parameters in the algorithm to maintain the quality of the cluster. Apriori selection of parameters is possibly avoided when designing algorithms in order to provide quality cluster [121].

e. Ability to Deal with Noisy Data:

Most real-world datasets contain outliers, missing, unknown, or erroneous data. Some clustering algorithms are sensitive to such data and may lead to clusters of poor quality.

f. **Insensitivity to the Order of Input Records:**

Some clustering algorithms are sensitive to the order of feeding input data. For example, the same set of data, when presented with different orderings, the algorithms may generate different clusters. It is important to develop algorithms that are insensitive to the order of input.

g. **High Dimensionality:**

The data in data warehouse generally is of high dimensional. Most of the clustering algorithms [22, 79] are good enough to handle the low dimensional datasets. It is challenging to cluster objects in high-dimensional space [27, 28], particularly considering very sparse and highly skewed data.

2.4.2 Clustering Algorithm:

The mostly used general clustering algorithms are presented in [22, 79]. Murtagh has discussed the advances in hierarchical clustering algorithms [122] and Baraldi investigated several models for fuzzy and neural network clustering [123].

A number of review papers are found in [124-125]. Some frequently used and popular clustering algorithms are briefly discussed below:

a. Partition Based Clustering:

Given a dataset of N objects, a partitioning method [22] constructs k ($k \leq N$) partitions each represents a cluster. The algorithm classifies the data into k groups, which together satisfy the following:

- Each group must contain at least one object, and
- Each object must belong to exactly one group (hard clustering).

The algorithm uses an iterative relocation technique based on some objective functions that attempt to grow the partitioning by moving objects from one cluster to another cluster. The general criterion of a good partitioning is that objects in the same group are closed to each other, whereas objects of different clusters are far distant from each other. There are several kinds of other criteria [126] for judging the quality of partitions. Partition based clustering algorithm needs to modify for dataset with difficult shape of clusters. Most common partition based clustering algorithms are (i) *k*-means clustering algorithm [127-128] which follows a simple iterative approach to cluster a dataset into k groups or clusters where k is set a priori. (ii) *k*-medoids clustering algorithm [129] instead of taking the mean value of the objects in a cluster as a point of reference, the medoid is used as it is the most centrally located object in a cluster. Thus, the partitioning of objects can still be performed based on the principle of minimizing the sum of the dissimilarities between every object and its related reference point (iii) *k*-Prototype clustering algorithm [130] is similar to *k*-means clustering algorithm but *k*-means clustering algorithm can handle only numeric attributes as its cost function is numerically measured. So, to handle categorical attributes *k*-prototype algorithm [130] can be used where the intra class similarity of objects can be measured for both numerical and categorical attributes (iv) Clustering LARge Applications (CLARA) [131]. CLARA draws multiple samples of the dataset, applies PAM [129] on each sample, and its best clustering returns as the output.

b. Hierarchical Clustering:

Hierarchical clustering (HC) algorithms [132, 133] organize data into a hierarchical structure according to the proximity matrix [134]. The results of HC are usually depicted by a binary tree or dendrogram. The whole dataset is represented by the root node of the dendrogram, and every leaf node is considered as a data object. The intermediate nodes, thus, describe the extent so that the objects are proximal to each other; and the height of the dendrogram usually expresses the distance between each pair of clusters or objects, or a cluster and an object. The clustering results can be obtained by cutting the dendrogram at different levels.

The HC algorithms are generally classified as agglomerative methods [133] and divisive methods [133]. Agglomerative clustering starts with N number of clusters and each of them include exactly one object. A series of merge operations are then followed that finally lead all objects of the same group. On the other hand, divisive clustering proceeds in a reverse direction.

At the primary step, the whole dataset belongs to a cluster and then successively divides it until all clusters are singleton clusters. For a cluster with N objects, there are $(2^{(N-1)} - 1)$ possible two-subset divisions, which is very expensive in computation, and not commonly used in practice.

In recent years, with the requirement for handling large-scale datasets in data mining and other fields, many new HC techniques such as CURE [135], ROCK [136], Chameleon [137], and BIRCH [138] have appeared and greatly improved the clustering performance. Though divisive clustering is not commonly used in practice, some of its application can be found in [139]. Two divisive clustering algorithms, named *MONA* and *DIANA*, are described in [139].

c. Density-based Clustering:

Density-based clustering methods [22] have been applied to discover clusters with arbitrary shape. Typically, these clusters are dense regions of objects in the space of dataset that are separated by regions of low density (representing noise). Two popular density-based spatial clustering techniques are DBSCAN [140] and DENCLUE [140].

DBSCAN searches for clusters by checking the $\in$-neighborhood of each point in the database. If the $\in$-neighborhood of a point p contains more than a threshold, a new cluster with p as a core object is created. DBSCAN then iteratively directly collects density-reachable objects from the core objects, which may involve the merge of a few density-reachable clusters. This process terminates when no new points can be added to any clusters. DENCLUE is a clustering algorithm based on a collection of density distribution functions, described below-

- The control of each data object can be formally modeled with the help of a mathematical function, called an influence function that describes the impact of a data point within its neighborhood.
- The overall density of a data space can be modeled logically as the sum of the influence functions of all data objects.
- Clusters can then be determined mathematically by identifying density attractors, where density attractors are neighboring maxima of the overall density function.

There are several advantages of DENCLUE method in comparison with other clustering algorithms such as it has a solid mathematical foundation and generalizes other clustering methods, including partition-based, hierarchical, and locally based methods. However, the method requires careful selection of the density parameter and the noise threshold, as the selection of such parameters may significantly influence the quality of the clustering solutions.

d. Fuzzy *c*-Means Clustering:

Fuzzy c-means (FCM) [141, 142, 143] is a method of clustering which allows one piece of data to belong to two or more clusters. This clustering method is mostly used in pattern recognition problem. It is based on minimization of the objective function defined in Equation (2.14).

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}^{m}\left\|x_i - c_j\right\|^2, \quad 1 \le m < \propto \tag{2.14}$$

Where m is any real number greater than 1, u_{ij} is the degree of membership of x_i in cluster j, x_i is the d-dimensional measured data, c_j is the d-dimension center of the cluster, and ||*|| is any norm stating the similarity between any measuring data and the center.

Fuzzy partitioning accepts the objective function through an iterative optimization with the update of membership u_{ij} and the cluster centers c_j by Equation (2.15) and Equation (2.16).

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{\left\|x_i - c_j\right\|}{\left\|x_i - c_k\right\|}\right)^{\frac{2}{m-1}}} \tag{2.15}$$

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m}.x_i}{\sum_{i=1}^{N} u_{ij}^{m}} \tag{2.16}$$

This iteration will stop when, $\max_{ij}\left\{\left|u_{ij}^{k+1} - u_{ij}^{k}\right|\right\} < \varepsilon$, where ε is a termination criterion between 0 and 1, while k is the iteration steps. This method converges to a local minimum or a saddle point of J_m.

2.4.3 Cluster Validation:

The clustering algorithms [22] partition data into an appropriate number of subsets. Although for some applications, the number of clusters, k can be determined from the domain knowledge but in most of the cases, k is unknown and needs to be estimated exclusively from the data itself. Many clustering algorithms [127, 129, 130] take k as an input parameter, and it is true that the quality of the resulting clusters is dependent on the estimation of k value.

Cluster validation, a very important issue in cluster analysis, is the solution of these problems. It is the measurement of the goodness or quality of a cluster relative to other clusters generated by clustering algorithms using different parameter values. There are many approaches to find the natural number of clusters. Some cluster validation measures like, compactness, connectedness, separation, and combinations, take a clustering method and the underlying dataset as the input, and employ information intrinsic to the dataset to review the quality of the clustering.

a. Compactness:

Compactness is an indicator of the scattering of the data within a particular cluster. It measures clusters compactness or homogeneity, with intra-cluster variance as their most popular representative. Numerous variants of measuring intra-cluster homogeneity are possible such as the evaluation of maximum or average pair-wise intra-cluster distances, maximum or average center-based similarities or the use of graph-based methods.

b. Connectedness:

It attempts to assess how well a given partitioning agrees with the conception of connectedness, i.e., to what degree a partitioning examines local densities and groups data items together with their nearest neighbors in the data space.

c. Separation:

Separation is an indicator of the isolation of clusters from one another. It measures the degree of separation between individual clusters. For example, a general rating for a partitioning can be stated as the average weighted inter-cluster distance, where the distance between two individual clusters is computed as the distance between cluster centroids, or as the minimum distance between data objects belonging to different clusters.

d. Combinations:

The above measures can be combined according to the particular idea of clustering quality that they occupy. Combinations of compactness and separation are particularly admired, as the two classes of measures show opposite tendency while intra-cluster homogeneity progress with a rising number of clusters, the distance between the clusters tends to deteriorate.

Thus, a number of procedures assess both inter-cluster separation and intra-cluster homogeneity, and calculate a final score as the linear or nonlinear combination of the two measures. Dubes called the difficulty of determining the clusters number "the fundamental problem of cluster validity" [144].

There are many validation indices like *Davies-Bouldin* (DB) index [145], *Dunn* index [145], *WB-index* [146], *H-index* [146], *Silhouette* index [147], and *CS-index* [148] for predicting quality of the clusters.

2.5 Classification Analysis:

Classification analysis [14, 15, 80] like cluster analysis plays an important role for understanding the intrinsic properties present in the datasets. Classification is a type of data analysis technique used in decision making to predict the future data trends.

Data classification is a two-step procedure, namely learning and classification. In the first phase, a classification system is developed depicting a predetermined set of data classes or concepts. In the second phase, the classifier model is used for classification of data. In this section, the various classification algorithms related to the work is discussed.

2.5.1 Issues Regarding Classification Analysis:

Different issues and the measures regarding the data classification are discussed in this section. The following preprocessing steps are applied to improve accuracy, efficiency, and scalability of the classifiers.

a. Data Cleaning:

Real-world data is often incomplete, noisy, and inconsistent. Data cleaning procedures [1] attempt to predict the missing values, remove noise and tackle inconsistencies in the data, which helps to reduce the ambiguity during learning and classification.

b. Feature Selection:

Dataset contains many attributes which are irrelevant and not significant in decision-making process. Hence, relevance analysis may be done on the data for removing any irrelevant or redundant attributes in order to make the dataset ready for learning. In machine learning, this step is known as feature selection [12, 13, 86].

It is often useful, and sometimes necessary, to reduce the dimension of dataset with as less information loss as possible. Ideally, total time spent on relevance analysis and learning using the "reduced" feature subset is less than the time that would have been spent on learning using the original set of features. Hence, such analysis improves classification efficiency.

c. Data transformation:

Data are transformed or consolidated into forms appropriate for mining. One of such form is generalization to higher-level concepts using concept hierarchies. This is particularly useful for continuous valued attributes.

d. Data Normalization:

If neural networks used for distance measurement in the learning step, then generally data are normalized.

Normalization does scale of attribute values within a specified range. For distance-based methods, normalization manages to handle large attribute ranges by outweighing attributes with initially smaller ranges.

2.5.2 Classification Algorithms:

A classification method is a systematic approach to developing classification models of a dataset. There are various classification algorithms such as Decision Tree classifiers [5], Rule-based classifiers [149], Naïve Bayes classifiers [7], Neural Networks [3], Support Vector Machines [6] and rough set Theory [150, 151] based classifier.

Each method has their own learning strategy to identify a model that best fits the relationship between the attribute set and the class (decision) label of the input dataset.

The model generated by the learning process should both fit the training data correctly and at the same time predict the class labels of the test data accurately. In this section, different types of classification algorithms are discussed.

a. Decision Tree based Classifier:

It is a tree, in which a choice between numbers of alternatives is represented by each branch node and each leaf node represents a classification or decision.

In principle, there is exponential growth of decision trees that can be created from a given set of features, while a number of trees are more precise than others.

Searching the optimal tree is computationally infeasible due to exponential size of the search space.

- **Decision Tree Classifier Algorithm:**

The basic algorithm of decision tree induction is a greedy algorithm that generates decision trees in a top-down recursive divide-and-conquer manner.

The algorithm presented below is a version of ID3 [152], a well-known decision tree induction algorithm. The basic principle of the algorithm is as follows:

i. The tree starts as a single node representing the training samples.

ii. If the samples are all of the same class, then the node becomes a leaf node, and it is labeled with that class.

iii. Otherwise, the algorithm uses an entropy-based measure [153] known as information gain, a heuristic function [154] for selecting the attribute that best separate the samples into individual classes. This attribute becomes the test or decision attribute at the node. All attributes are categorical or discrete-valued and the continuous-valued attributes are discretized in this algorithm.

iv. A branch is created for each known value of the test attribute, and the samples are partitioned accordingly.

v. The algorithm uses the same process recursively to form a decision tree for the samples at each partition. Once an attribute has occurred at a node, it need not be considered in any of the node's descendents.

The partition using recursion stops only when any one of the following conditions is true-

i. All samples for a particular node belong to identical class, or
ii. There is no attribute left on which the samples may be further partitioned. Here, majority voting is used and involves changing the given node into a leaf node and labeling it with the decision label in majority among samples. On the other hand, the class distribution of the node samples may be stored.
iii. No samples are for the branch test-attribute. In this situation, a leaf is formed with the majority class in samples.

- **Characteristics of Decision Tree Classifiers:**

Following are some of the important characteristics of decision tree induction algorithms.

a. Decision tree induction is a nonparametric approach [155] for building classification models. It does not require any prior assumptions like probability distributions fulfilled by the class label and other attributes.
b. Finding an optimal decision tree is an NP-complete problem [156]. Many decision tree algorithms employ a heuristic-based technique to direct their search in the large hypothesis space. The ID3 algorithm [152] uses a greedy, top-down, and recursive partitioning strategy for growing the decision tree.
c. The techniques developed for constructing decision trees are computationally expensive, making it possible to quickly construct models even when the training set size is bulky. In addition, once a decision tree has been built, classifying a test sample is very fast, with a worst-case complexity of O(*w*), here *w* is the maximum depth of the tree.
d. Decision trees, especially smaller-sized trees, are comparatively easy to understand. The classification accuracies of the trees are also comparable to other classification methods for many datasets.
e. The occurrence of redundant attributes does not unfavorably affect the classification accuracy of decision trees. One of the two redundant attributes must not be utilized for splitting once the other attribute has been selected. Several feature selection methods can help to improve the accuracy of decision trees by removing the redundant features during preprocessing.
f. A sub-tree can be replicated several times in a decision tree, as shown in Figure 2.4. This makes the decision tree more complex. Such circumstances can happen from decision tree implementations that rely on a single attribute test-condition at every internal node. While a good number of decision trees use a divide-and-conquer partitioning approach, the same test condition can be applied to different parts of the attribute space, thus leading to the sub-tree replication problem [157].

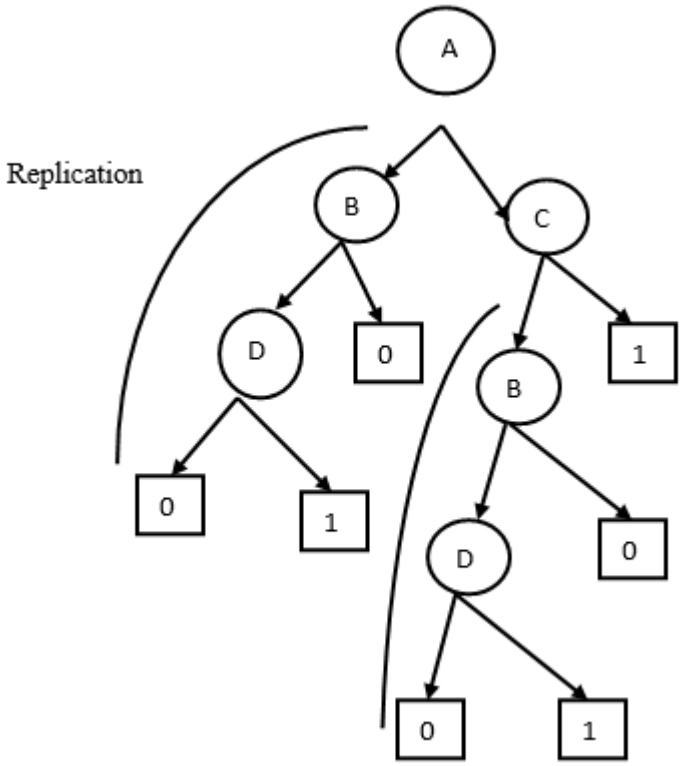


Figure 2.4: Tree Replication Problem

g. Since most decision tree algorithms employ a top-down, recursive partitioning approach, the number of records becomes smaller as the tree is traversed down. At the leaf nodes, the number of records may be too small to make a statistically significant [158-160] decision about the class representation of the nodes. This is known as the data fragmentation [161] problem. One possible solution is to disallow further splitting when the number of records falls below a certain threshold.

b. Rule-based Classifier:

A rule-based classifier [149] actually classifies objects using a collection of *IF-THEN* rules. The rules for the model are represented in a disjunctive normal form, $RS = (R_1 \vee R_2 \vee \ldots \vee R_k)$, where *RS* is known as the rule set and R_i ($i = 1, 2, \ldots, k$) are the classification rules. Each classification rule can be represented by R_i: $(condition_i) \rightarrow y_i$, where the left-hand side of a rule is called as antecedent part and the right-hand side is named as consequent part of the rule to achieve the predicted class y_i. The rule set, generated by a rule-based classifier, satisfies the following two important properties-

- **Mutually Exclusive Rules:** The rules in a rule set (*RS*) are mutually exclusive if antecedent of no two rules match for the same dataset record. This property guarantees that each record is covered by at most one rule in *RS*.

- **Exhaustive Rules:** A rule set (*RS*) is exhaustive if there is a rule for each combination of attribute values. This property ensures that every data record is covered by at least one rule in *RS*.

These two properties together ensure that every data record is covered by exactly one rule. Unfortunately, many rule-based classifiers do not have such properties.

I. Characteristics of Rule-based Classifiers:

The rule-based classifier has the following characteristics:

i. The expressiveness of a rule set is more or less comparable to that of a decision tree. Both rule-based and decision tree classifiers construct rectilinear partitions of the feature space and allocate a class label to each partition. However, if the rule-based classifier permits several rules to be triggered for a particular record, then a more complex decision boundary can be constructed.
ii. Rule-based classifiers are generally used to produce descriptive models that are easier to interpret and give comparable performance to the decision tree classifier.

c. Bayesian Classifier:

The relationship between the attribute set and the class variable is non-deterministic in most applications. The class label of a test-record cannot be expected with certainty even though its attribute set is identical to some of the training instances. The principle of the Bayes theorem [162] for solving classification problem is described, and then a brief description of Naïve Bayes classifier [7] is also provided.

- **Basics of Bayes Theorem:**

Let X is a data sample whose class label is unknown, and H hypothesizes that X belongs to class C. For classification problems, it is required to determine posterior probability $P(H|X)$, the probability of hypothesis H, given the observed data sample X while $P(H)$ is the prior probability of hypothesis H, described in Equation (2.17).

Bayes Rule: Given a training data sample X, posteriori probability of a hypothesis H, $P(H|X)$ follows the Bayes theorem as defined by Equation (2.17).

$$P(H|X) = \frac{P(H|X)P(H)}{P(X)} \qquad (2.17)$$

- **Naïve Bayes Classifier:**

The steps of Naïve Bayes classifier are described below:

I. Each data sample is presented by an n-dimensional feature vector, $X = (x_1, x_2, \ldots, x_n)$, describing n measurements made on the sample using n attributes $A_1, A_2, \ldots, A_n$ respectively.

II. Suppose there are *m* classes, C_1, C_2, …, *Cm.* Given an unknown data sample *X*, the classifier predicts that *X* belongs to the class having the highest posterior probability. The naïve Bayes classifier assigns an unknown sample *X* to the class C_i (for $i = 1, 2, ..m$) if and only if Equation (2.18) holds.

$$P(C_i|X) > P(C_j|X), for\ i \leq j \leq m\ and\ j \neq i. \qquad (2.18)$$

The class C_i ($i = 1, 2, ..m$), for which $P(C_i|X)$ is maximum, is computed by Equation (2.18) using Bayes Rule, defined in Equation (2.19).

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)} \qquad (2.19)$$

III. As $P(X)$ is constant for all classes, only $P(X \mid C_i)\, P(C_i)$ required to be maximized. If the class prior probabilities are not known, then it is assumed that the classes are equally likely, means $P(C_1) = P(C_2) = \ldots = P(C_m)$, and therefore maximize $P(X \mid C_i)$, otherwise, maximize $P(X \mid C_i)\, P(C_i)$. It is noted that the class prior probabilities $P(C_i)$ may be computed by s_i / s, where s_i is the number of training samples of class C_i and *s* is the total number of training samples.

IV. It is extremely expensive to compute $P(X \mid C_i)$ for a datasets with several attributes. In order to reduce computation cost in evaluating $P(X \mid C_i)$, the naïve theory of class conditional independence [163] is estimated as shown in Equation (2.20). It assumes that the values of attributes are conditionally independent of one another, given the class label of the sample, which implies that there are no dependence relationships among the attributes.

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i) \qquad (2.20)$$

The probabilities $P(x_1 \mid C_i)$, $P(x_2 \mid C_i)$, …, $P(x_n \mid C_i)$ can be estimated from the training samples, where if

i. A_k is categorical, then $P(x_k \mid C_i) = s_{ik} / s_i$, s_{ik} is the number of training samples of class C_i having value x_k for A_k, and s_i is the number of training samples belonging to C_i.
ii. A_k is continuous-valued then the attribute is normally assumed to have a Gaussian distribution, thus $P(x_k \mid C_i)$ is calculated by Equation (2.21), where $g(x_k, \mu_{Ci}, \sigma_{Ci})$ is the Gausian (normal) density function [164] for attribute A_k, while μ_{Ci} and σ_{Ci} are the mean and standard deviation, respectively, for the given values of attribute A_k for training samples of class C_i.

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}) = \frac{1}{\sqrt{2\pi}\,\sigma_{C_i}} e^{-\frac{(x_k-\mu_{C_i})^2}{2\sigma_{C_i}}} \qquad (2.21)$$

(v)In order to classify an unknown sample X, $P(X \mid C_i)\, P(C_i)$ is estimated for every class C_i. Sample X is assigned to class C_i if and only if $P(X \mid C_i)\, P(C_i) > P(X \mid C_j)\, P(C_j)$ for $1 \leq j \leq m,\ j \neq i$. Alternatively, it is assigned to class C_i for which $P(X \mid C_i)P(C_i)$ is maximum.

- **Effectiveness of Bayesian Classifiers:**

Bayesian classifiers have minimum error rate in comparison to other classifiers, but it is not always true due to inaccuracies in assumptions and lack of available probability of data.

Nevertheless, several experimental studies of this type of classifier in comparison to decision tree [5] and neural networks [3] have observed showing comparable results in some domains.

Bayesian classifiers are also useful because they offer a theoretical justification for alternative classifiers, based on neural networks [3] and curve-fitting [165] algorithms.

d. K-Nearest Neighbor Classifiers:

K-nearest neighbor (KNN) classifiers [37] classify objects based on learning by analogy. It is one of the most fundamental classification methods and applicable for classification in absence of a prior knowledge or insufficient knowledge about the distribution of the data. *K*-nearest neighbor classification was developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine.

- **Classification Method:**

The training samples are described by *d*-dimensional numeric attributes where each sample represents a point in a *d*-dimensional space.

When an unknown sample is given, a KNN classifier investigates the object space for the *K* training samples that are closest to the unknown sample.

These *K* training samples are the *K* nearest neighbors of the unknown sample. Euclidean distance [119] is computed to define the closeness of two points.

The representation diagram of KNN classifier is given in Figure 2.5.

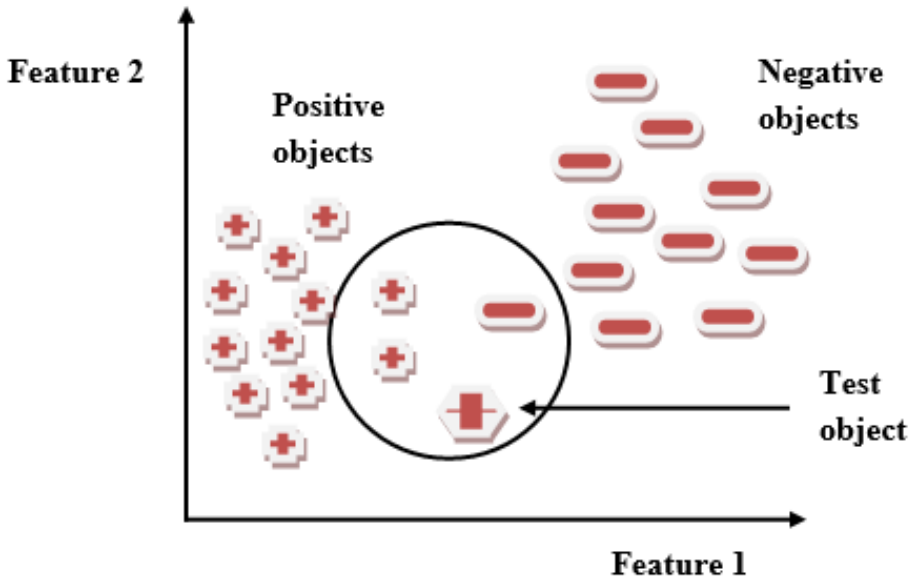


Figure 2.5: Representation diagram of KNN classifier

- **Characteristics of Nearest Neighbor Classifiers:**

The characteristics of nearest neighbor classifier are summarized below:

i. It is a kind of instance-based learning that uses specific training instances to make the predictions. In this method no abstraction or model is derived from data.
ii. Classifying a test data is quite expensive in order to compute the proximity values individually between the training and test data.
iii. The prediction is based on local information, so for small values of K the classifiers are quite susceptible to noise.
iv. The classifiers can produce arbitrary shaped decision boundaries, which have high variability because they depend on the composition of training samples. By increasing the number of nearest neighbors, variability may be reduced considerably.

The classifiers can produce wrong predictions unless the appropriate proximity measure and data preprocessing steps are invoked.

e. Support Vector Machines:

Support vector machine (SVM), pioneered by Vapnik [6, 166] is a supervised learning technique. A hyper plane or a set of hyper planes in a high dimensional space are constructed by SVM classifier, applied for classification analysis of dataset based on the Structural Risk Minimization principle [167]. The SVM conceptually implements the idea where (i) input vectors are non-linearly mapped to a very high-dimensional feature space and (ii) a linear decision surface is constructed in the feature space.

Intuitively, a good separation is achieved by the hyper plane that has the largest distance to the nearest training data points of any class (called functional margin), since in general larger the margin, lower is the generalization error of the classifier. Suppose some given data points each belongs to one of two classes, and the goal is to decide in which class a new data point belongs.

- **Formalization:**

SVM is a large-margin classifier, which aims at finding a decision boundary between classes that is maximally far from any point in the training data. Figure 2.6 shows an example of the classification using SVM. Unlike other linear machine learning methods, SVM defines the class separation criterion by looking for a decision hyper plane that maximizes the distance from any data point. The distance from the decision boundary to the closest data points determines the margin of the classifier. These points are called support vectors and they are the only ones defining the position of the separating hyper plane. This makes the SVM classifier particularly robust and well suited for classification with low or unbalanced training data. Despite the fact that SVM is inherently binary classifier, it can be used for multiclass problems as well. In multiclass problem, the simple "one-versus-all" classification scheme can be applied. Another possibility is to train n(n−1)/2 classifiers and choose the class of a given document that is selected by most of the classifiers.

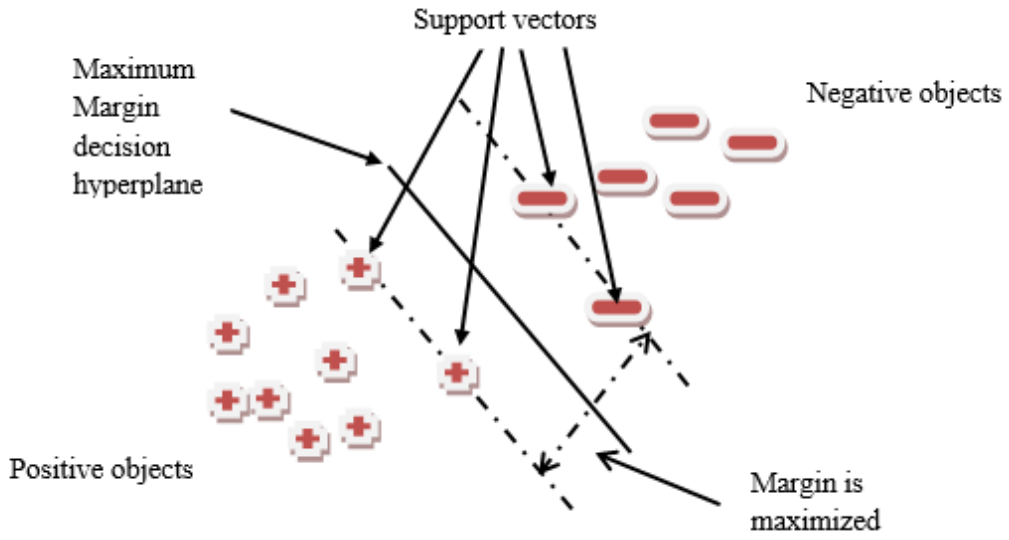
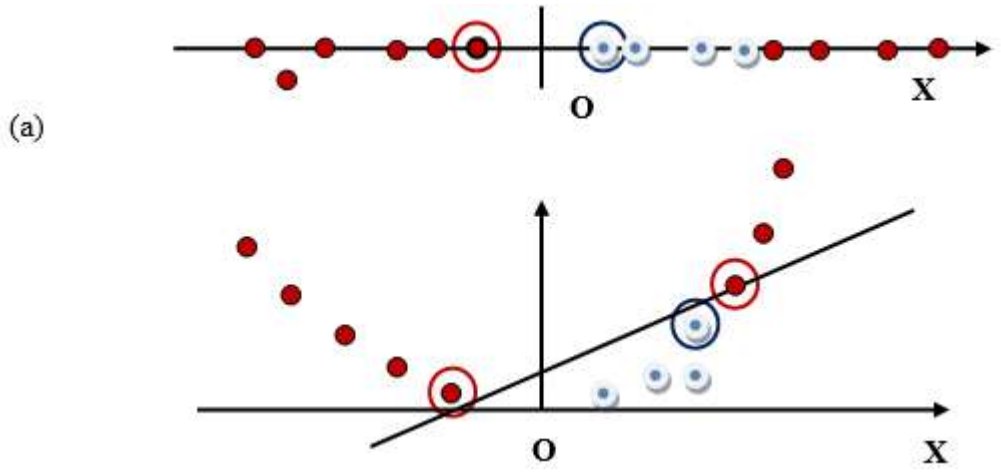


Figure 2.6: Classification in a linearly separable domain using SVM.

SVM is also applicable when the classes are not linearly separable in principle. A general solution is to map the feature space into a higher-dimensional feature space where the training set is linearly separable. Figure 2.7 shows an example of mapping one-dimensional space, where in Figure 2.7 (a) the points are not linearly separable, to a two-dimensional space Figure 2.7 (b) to make it linearly separable.



(b)

Figure 2.7: Projection of data into higher dimensional space to make linearly separable.

This approach makes a linear classification in the high-dimensional space, corresponding to the non-linear classification in the original space. However, the mapping function has to preserve the relatedness between data points in the higher-dimensional space.

This is often called the *kernel trick* [168]. A kernel function is a function that corresponds to a dot product in some feature space and satisfies the *Mercer's condition* [169].

Some common kernels are Radial Basis Function (RBF) [170], Polynomial [171, 172] and Sigmoidal [173]. A special property of SVM is that they simultaneously minimize the empirical *classification error* and maximizes the *geometric margin*.

- **Characteristics of SVM:**

SVM is one of the most widely used classification algorithm for its many attractive qualities. The general characteristics of SVM are summarized below-

i. The SVM learning problem can be formulated as a convex optimization problem [174], in which efficient algorithms are available to find the global minimum of the objective function.
ii. SVM performs capacity control by maximizing the margin of the decision boundary. The user must still provide other parameters such as the type of kernel function to use and the cost function for introducing each slack variable.
iii. SVM is applied to categorical dataset by introducing dummy variables for each categorical attribute value present in the dataset.
iv. Though SVM is only directly applicable for two-class tasks, but it can be extended for multi-class task, called Multi-class SVM.

- **Merits and Demerits of SVM:**

The extensive time to train is actually the major weakness of SVM. The theoretical complexity of SVM training is cubic with respect to the size of the training set. The recent research on the SVM training has focused on reducing the time, often by approximate solutions.

The empirical complexity of the current approaches is about $O(|D|^{1.7})$, where $|D|$ is the number of instances in the training set.

On the positive side, many studies show that SVM consistently achieves good performance on classification tasks, often substantially outperforming existing methods. The ability, to generalize well in high-dimensional feature space, eliminates the need for feature selection, which makes the application considerably easier. SVM are robust and do not require any parameter modification, because they can discover good parameter settings automatically.

f. Classification using Backpropagation:

Artificial neural networks, a nonlinear statistical data modeling tools, can be used to model complex relationships between inputs and outputs or to find patterns in data by simulating the structure and/or functional aspects of biological neural networks.

The first learning algorithm came in 1959 by Rosenblatt [175] who suggested that if a target output value is provided for a single neuron with fixed inputs, one could incrementally change weights to learn to produce these outputs using the perceptron-learning rule.

Backpropagation or propagation of error [176], pioneered by Remelhart, is a common method to train the artificial neural networks to classify the samples.

As the algorithm's name implies, the errors between the output and the target value is propagated backwards from the output nodes to the inner nodes of the network consists of an interconnected group of artificial neurons and processes information using a connectionist approach, shown in Figure 2.8.

Backpropagation is used to calculate the gradient of error of the network with respect to the weights which are modified based on the learning rule to minimize the error. Backpropagation usually allows quick convergence on satisfactory local minima for error in the kind of networks to which it is suited. Generally, backpropagation networks are considered as multilayer perceptrons typically with one input, one hidden, and one output layer. In order for the hidden layer to serve any useful function, multilayer networks must have nonlinear activation functions [177] for the multiple layers.

To classify the objects, the attributes for each training instances are fed simultaneously into the input layer.

The weighted output traverses from one intermediate layer (hidden layer) to another and reach to the output layer, which generates the network's prediction for a given sample.

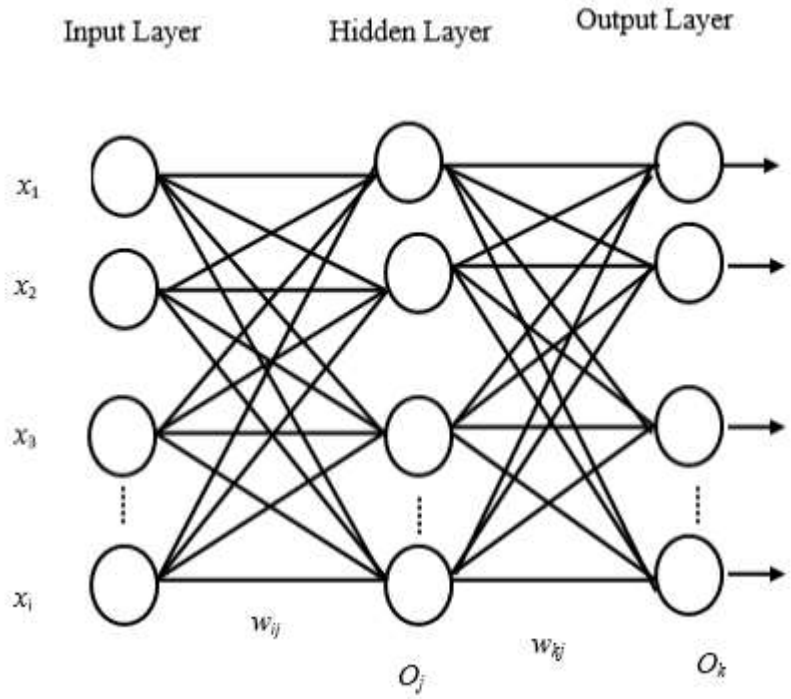


Figure 2.8: A multilayer feed-forward neural network

Acquired knowledge in the form of a network of units connected by weighted links is difficult for humans to interpret which motivated research in extracting the knowledge embedded in trained neural networks and in representing that knowledge symbolically. Various algorithms for extraction of rules from networks and sensitivity analysis [178] have been proposed, where restrictions are imposed regarding procedures used in training, the network topology, and the discretization of input values. Fully connected networks are difficult to articulate and as an initial step, network pruning is required to extract rules from neural networks. This consists of removing weighted links that do not result in a decrease in the classification accuracy of the given network. Once the trained network has been pruned, some approaches then applied to perform link, unit, or activation value clustering. Rules are derived relating combinations of activation values with corresponding output unit values. Similarly, the sets of input values and activation values are studied to derive rules describing the relationship between the input and hidden layers. Finally, the two sets of rules may be combined to form IF-THEN rules.

Neural nets have been successfully used to solve many complex and diverse tasks, ranging from autonomously flying aircraft to detecting credit card fraud.

2.5.3 Ensemble of Classifiers:

An ensemble classifier system is obtained by combining multiple and diverse classification models, also known as multiple classifier systems [16, 179].

In statistics and machine learning, ensemble methods use multiple models to obtain improved analytical performance than that obtained from single constituent models [180]. Ensemble systems are useful to deal with large volumes of data or lack of adequate data.

When the amount of training data is too large to train a single classifier, the data is strategically partitioned into smaller subsets. Each partition is used to train the separate classifiers, which are combined using an appropriate combination rule. On the other hand, if there are smaller amount of data, then bootstrapping [16] can be used to train different classifiers using different bootstrap samples of the data. Each bootstrap sample is a random sample of the data drawn with replacement and treated as if it is autonomously drawn from the fundamental distribution [181]. Research in ensemble systems have been expanded rapidly, that includes composite classifier systems [182], mixture of experts [183], stacked generalization [184], combination of multiple classifiers [179, 185-187], dynamic classifier selection [188], classifier fusion [189, 190], classifier ensembles [16, 192], and many others. In *classifier fusion* strategy, all classifiers are trained over the entire feature space. In this case, the combination of classifiers involves merging the individual classifiers to achieve a stronger expert of superior performance. Examples of this approach include bagging predictors [191], boosting [192], AdaBoost [193] and many variations of those.

a. Ensemble Combination Rules:

An ensemble classifier system can be trained basically on different subsets of the training dataset, different parameters of the classifiers, or even with different feature subsets as in arbitrary subspace models. The classifiers can then be combined using one of the several different combination rules. Some of these combination rules functions on class labels only, but other classifiers need continuous outputs that are interpreted as support given by the classifier to each of the classes.

b. Algebraic Combiners:

Algebraic combiners are *non-trainable combiners*, where continuous valued outputs of classifiers are combined using an algebraic expression, such as minimum, maximum, sum, mean, product, median etc. In each case, the final ensemble decision $H_{final}(x)$ is the class i that receives the largest support $\mu_i(x)$ after the algebraic expression is applied to specific supports obtained by each class. Specifically, $H_{final}(x) = argmax_i \mu_i(x)$, where the final class supports are calculated as follows-

- **Mean rule:** $\mu_i(x) = \frac{1}{T}\sum_{t=1}^{T} d_{t,i}\,(x)$
- **Sum rule:** $\mu_i(x) = \sum_{t=1}^{T} d_{t,i}\,(x)$ (identical final decision as the mean rule)
- **Weighted sum rule:** $\mu_i(x) = \sum_{t=1}^{T} w_t d_{t,i}\,(x)$(where w_t is the weight assigned to the t-th classifier according to some measure of performance)
- **Product rule:**$\mu_i(x) = \prod_{t=1}^{T} d_{t,i}(x)$
- **Maximum rule:**$\mu_i(x) = \max_{t=1,2,\ldots,T}\{d_{t,i}(x)\}$
- **Minimum rule:**$\mu_i(x) = \min_{t=1,2,\ldots,T}\{d_{t,i}(x)\}$
- **Median rule:**$\mu_i(x) = med_{t=1,2,\ldots,T}\{d_{t,i}(x)\}$

i. Voting Based Methods:

Voting based methods function on class labels only, where $d_{t,i}$ is 1 or 0 depending on whether classifier *t* chooses *i* or not, respectively. Then the ensemble system chooses class *i* that accepts the largest total vote using following voting techniques-

- **Majority (Plurality) Voting**

$$\sum_{t=1}^{T} d_{t,i}(x) = \max_{t=1,2,\dots,C} \sum_{t=1}^{T} d_{t,i} \qquad (2.22)$$

When the classifier outputs are independent, the majority voting combination always leads to a performance improvement. For a two-class problem, if there are a total of *T* classifiers, the ensemble decision will be correct if at least $\left\lfloor \frac{T}{2} + 1 \right\rfloor$ classifiers decide the proper class. At present, suppose that every classifier has a probability *p* of making a correct decision. After that, the probability of ensemble making an accurate decision has a binomial distribution, specifically, the probability of choosing k (>⌊T/2 + 1⌋) number of correct classifiers out of *T* is defined in Equation (3.18).

$$P_{ens} = \sum_{k=\left(\frac{T}{2}\right)+1}^{T} \binom{T}{k} p^k (1-p)^{T-k} \qquad (2.23)$$

Then, $P_{ens} \to 1, as\ T \to \propto\ if\ p > 0.5$ and $P_{ens} \to 0, as\ T \to \propto\ if\ p < 0.5$

Note that $p > 0.5$ is required and adequate for a two-class problem, whereas it is sufficient, but not necessary for multi class problems.

- **Weighted Majority Voting:**

$$\sum_{t=1}^{T} w_t d_{t,i}(x) = \max_{t=1,2,\dots,C} \sum_{t=1}^{T} w_t d_{t,i} \qquad (2.24)$$

The best possible weights for the majority voting (weighted) rule, is expressed by the relation; $w_t \propto \frac{p_t}{1-p_t}$, if the *T* classifiers are class-conditionally self-sufficient with classification accuracies $p_1, p_2, \dots, p_T$.

c. Ensemble Learning Algorithms:

Most of the classification techniques (except nearest neighbor classification) predict the class labels of unknown examples using a single classifier induced from the training data, resulting poor accuracy of the classifier. Ensemble of classifiers is the technique for improving classification accuracy by aggregating the predictions of multiple classifiers.

This method constructs a set of base classifiers using the training dataset and makes classification by taking a vote on the predictions, made by each classifier. A logical view of the ensemble method is presented in Figure 2.9. The main idea is to build multiple classifiers from the original dataset and then combined their output predictions when classifying unknown examples. The ensemble of classifiers can be constructed in many ways [194-196].

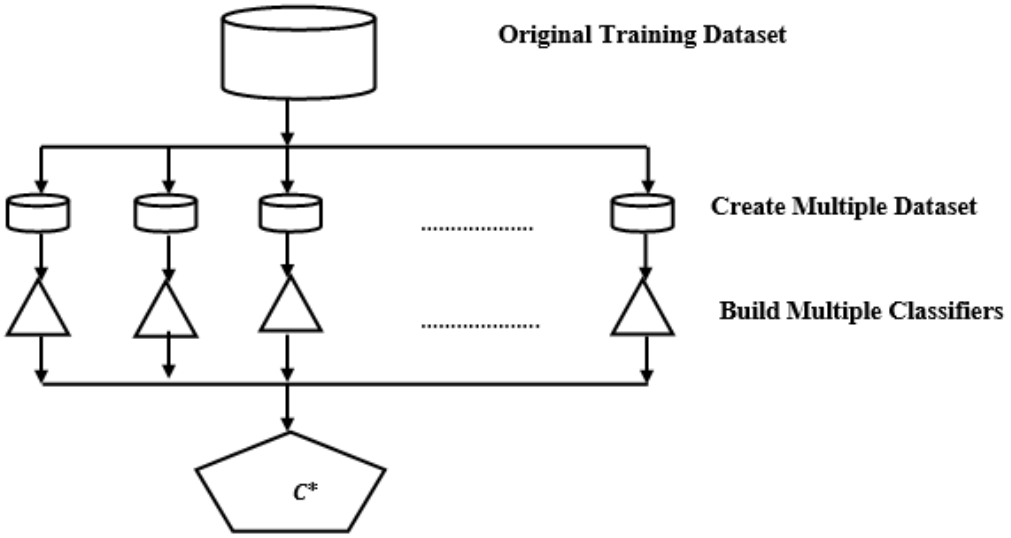


Figure 2.9: A logical view of the ensemble learning method

- **Bagging:**

Bagging or bootstrap aggregating [191] is a technique that repeatedly samples a dataset with replacement where uniform probability distribution is used. Every bootstrap sample has the same size as the original dataset. As the sampling is done with replacement, several times some instances may appear in the same training set, while others may be omitted from the training set.

On an average, a bootstrap sample D_i of size N, contains approximately 63% of the original training datasets in every sample has a probability $[1 – (1 – 1/N)^N]$ of being selected in each D_i. If N is sufficiently large, this probability converges to $(1 – 1/e) \approx 0.632$. If k is the number of bootstrap samples, then train a base classifier C_i on D_i, for each $i = 1, 2, \ldots, k$. After training of k classifiers, a test instance is assigned to the class that receives the highest number of votes.

Bagging improves generalization error by reducing the variance of the base classifiers. The performance of bagging depends on the stability of the base classifier. If a base classifier is unstable, bagging reduces the errors related with random fluctuations in the training dataset. If a base classifier is stable, and robust to minor perturbations in the training set, then the error of the ensemble is primarily caused by bias in the base classifier. In this situation, bagging may not be able to improve the performance of the base classifiers significantly.

- **Boosting:**

Boosting [192] is an iterative method used to adaptively modify the distribution of training examples so that the base classifiers focus on examples that are hard to classify. Unlike bagging, boosting assigns a weight to each training example and may adaptively modify the weight at the end of every boosting round. The weights assigned to the training examples can be used as a sampling distribution to draw a set of bootstrap samples from the original dataset or by the base classifier to study a model that is biased to higher weight examples.

Initially, equal weights, 1/*N*, is assigned to each of the *N* examples thus they are equally likely to be selected for training. A sample is taken according to the sampling distribution of the training examples to obtain a new training set. Next, a classifier is induced from the training set and used to classify all examples in the original data. The weights of the training examples are updated at the end of each boosting round. The algorithm [192] differs in terms of how the weights of the training examples are updated at the end of each boosting round and how the predictions made by each classifier are combined.

- **Stacked Generalization:**

In Wolpert's stacked generalization [184] method, an ensemble classifier system is first trained using bootstrapped samples of the training dataset, making *Tier* 1 *classifiers*, whose outputs are used to train a *Tier* 2 *classifier (meta-classifier)*. The underlying idea is to get idea about whether the training data have been properly learned. For example, if a specific classifier wrongly learned a certain region of the feature space, then consistently misclassifies instances will come from that region. The *Tier* 2 classifier in such circumstances able to learn these activities, and beside with the learned activities of other classifiers, it can correct such improper training.

- **Mixture of Experts:**

Jordan and Jacobs' mixture of experts [183] method generates several classifiers whose outputs are combined through a generalized linear rule. The weights of the combination are determined by a gating network typically trained by the expectation maximization (EM) algorithm [197]. Both the experts and the gating network require the input instances for training. Also, some mixture-of-experts models can be further combined to obtain a hierarchical mixture of experts [183]. Mixture of experts is particularly useful when different experts are trained on different components of the feature space, otherwise when heterogeneous feature sets are available to be used for a data fusion problem.

2.6 Classification Validation:

Estimating classifier accuracy is important in the sense that it allows one to evaluate how accurately a given classifier level data, which are not included in the training data set.

2.6.1 Issues Regarding Classification Validation:

Classification methods are compared and evaluated according to the following criteria.

a. Predictive Accuracy:

The accuracy refers to how well a given classifier correctly predicts the class label of new or previously unseen data. There are various methods [37] to estimate classifier accuracy. If the accuracy of the classifier is considered acceptable, the classifier can be employed to classify future dataset for which the class label is unknown.

b. Speed and Scalability:

Speed refers to the complexity and computational costs in generating and using the classifier system. Scalability of a method refers to the ability to construct classifier efficiently with the given large amount of data.

c. Robustness and Interpretability:

Robustness refers to the ability of classifier to make correct prediction in presence of noisy data or data with missing values. Interpretability is the level of understanding and insight provided by the classifier model and compactness of classification rules.

2.6.2 Classifier Validation Methods:

Using training data, a classifier is derived and then the accuracy estimation of the classifier can result in misleading more optimistic estimates for over-specialization of the learning algorithm to the dataset. Holdout [37] and cross-validation [37] are two common techniques for assessing classifier accuracy, based on randomly sampled partitions of the given data.

a. Holdout Method:

Here, given data are randomly partitioned into two independent sets, a training set, and a test set. Normally two thirds of the dataset are used as the training set, and the remaining one third is allotted to the test set. The training set is employed to develop the classifier, whose classification accuracy is predicted with the test set. The estimation is pessimistic because only a part of the initial dataset is used to develop the classifier.

b. *k*-fold Cross Validation:

In *k*-fold cross-validation, the initial dataset is at random partitioned into *k* mutually exclusive folds (S_1, S_2, …, Sk), and each fold is of equal size approximately. The training and testing are performed *k* times. In iteration *i*, subset S_i is used as the test set and the rest subsets are collectively used to train the classifier.

The accuracy estimation is the overall number of accurate classifications from the *k* iterations, divided by the total number of samples in the initial dataset. In stratified cross-validation [37], the folds are stratified so that the class distribution of the samples in every fold is approximately equal as that in the initial dataset. The use of such methods to estimate classification accuracy increases the overall computation time, however, is valuable for selecting along with several classifiers.

c. Alternative Accuracy Measure:

To know the alternative measurements of accuracy, the *'sensitivity'* or *recall* and *'specificity'* measures [37, 38, 198] are used. In addition, *fall_out* and *F1_Score* may also be used to assess the performance of the classifier. These four measures are defined by Equation (2.25), (2.26), (2.27) and (2.28) respectively.

$$Recall = \frac{TP}{P} = \frac{TP}{TP+FN} \quad (2.25)$$

$$Fall\_out = \frac{FP}{N} = \frac{FP}{FP+TN} \quad (2.26)$$

$$Specificity = \frac{TN}{N} = \frac{TN}{FP+TN} = 1 - Fall\_out \quad (2.27)$$

$$F1\_score = \frac{2 \times TP}{2 \times TP+FP+FN} \quad (2.28)$$

Where *TP*, *FP*, *TN*, *FN*, *P*, and *N* are the number of positive objects classified as positive, negative objects classified as positive, negative objects classified as negative, positive objects classified as negative, total positive objects, and total negative objects respectively.

The accuracy can be defined as a function of sensitivity and specificity given in Equation (2.29).

$$Accuracy = Sensitivity \times \frac{P}{(P+N)} + Specificity \times \frac{N}{(P+N)} \quad (2.29)$$

In classification, it is generally assumed that all samples are uniquely classifiable and therefore, each training sample belongs to only one class. But because of the variations of dataset in large databases, it would not be sensible to assume that all samples are uniquely classifiable. Rather, it would be feasible to assume that each sample may belong to more than one class. That time appropriate accuracy may not be achieved since it fails to consider the possibility of samples belonging to more than one class.

2.6.3 Statistical Analysis of Classifier:

To judge the performance of the classifier, statistical analysis [41] is also made to demonstrate that the method is statistically significant with respect to the different competitive algorithms. Any characteristics or measure of sample items is known as a statistic. Obtaining the estimate of an unknown parameter using a statistic and studying the properties of the obtained estimate is the prime objective of the statistical inference. In most of the research work the usual approach is to make generalization or to draw inferences based on samples about the parameters of the population from which samples are taken.

Statistical inference is concerned with two things such as Hypothesis testing and Estimation. In hypothesis testing, test the claims made about unknown population parameters using sample. Estimation means estimating unknown population parameters using a sample.

A research hypothesis is a predictive statement capable of being tested by scientific methods that relates an independent variable to some dependent variables. If two methods M_1 and M_2 are compared about its superiority and the process starts with the assumption that both the methods are equally good, then this assumption is called null hypothesis.

On the other side if the process starts with the assumption that the method M_1 is superior or the method M_2 is inferior then it is termed as an alternative hypothesis. The null hypothesis is symbolized as H_0 and the alternative hypothesis is H_1. The various steps associated with the hypothesis testing are

(i) Setting up the hypotheses (ii) Selecting a significance level (iii) Test statistics (iv) critical value and (v) decision about rejecting or not rejecting the null hypothesis. If the test of significance is based on certain parameters and their estimation then these tests are recognized as parametric test and examples are chisquare test [41], t-test [40, 41] etc. On the other side non- parametric test do not make any assumption about the parameter of the population and thus do not make use of the parameters of the distribution and the examples are Wilcoxon rank sum test [39, 41], Mann Whitney U test [41], Run test [41] etc. As the Wilcoxon rank sum test [39, 41] is a non-parametric test and it is valid for data of any distribution and much less sensitive to outliers compare to other testing methods so it is used in the thesis work to evaluate the statistical significance of the proposed methods.

2.7 Summary:

This chapter discusses major data mining tools and techniques related to data preprocessing activities such as missing value estimation and feature selection and different clustering and classification issues, algorithms, and their validations.

The chapter provides the description of experimental datasets. Missing value estimation is an important prepossessing step in data mining process.

The presence of missing values can influence the performance of clustering and classification algorithms. Several references are provided in the chapter to treat missing values to process the dataset.

Feature selection has been an active and fertile field of research area in data mining for data preprocessing.

The main objective of feature selection is to choose a subset of relevant and important features. In the chapter, a wide variety of feature selection algorithms have been reviewed. These algorithms are developed by different research communities to solve different problems, and those have their own merits and demerits.

At the preprocessing and post-processing phase, feature selection is an important supervised learning step to find a feature subset that produces higher classification accuracy. Cluster analysis is an essential tool for dataset analysis which includes a series of steps, ranging from preprocessing and algorithm development, to solution evaluation and validity. In the chapter, clustering algorithms and validation methods are reviewed.

These algorithms are developed by different research communities to solve different real-world problems and have their own merits and demerits. Therefore, it is concluded that, there is no clustering algorithm that can be universally acceptable for all kind of dataset.

Classifier is an important tool for exploration of unlabeled data. The classification method includes a series of steps, ranging from preprocessing and algorithm development, to ensemble of classifiers.

All steps are challenging, and researchers are working for quite long time, generating new issues and avenues in different disciplines. In the chapter, a wide variety of classification algorithms have been reviewed.

These algorithms are developed by different research communities, aiming to solve different real-world problems, and have their own merits and demerits.

Ensemble of classifiers generally improves the performance of classification and prediction of unknown datasets.

Using an ensemble of classifiers, instead of choosing just and combining their outputs can reduce the risk of a wrong selection of a particularly poorly performing classifier.

To judge the performance of the classifier, different classifier validation methods along with the classification performance metric are also reviewed in the chapter.

ISBN: 978-93-90847-37-2

Chapter 3

Feature Selection in Static Environment

3.1 Introduction:

Huge amount of data is being generated and collected in every moment in almost every field. As a result, the size of the datasets is growing along with accumulating a large number of features, which are not equally important in decision-making. The objective of feature selection is to select minimal set of important features from the large feature space by avoiding the selection of too many or too few features than is necessary.

Too few features may result loss of information while too many irrelevant features dominate important features that surpass the information present in the system. Thus, a trade-off is essential to find only the relevant features that preserve all the characteristics of the system. So, the features contribute the most to the decision must be retained.

The benefit of the feature selection is to save the learning time of the learning process by reducing the interference of irrelevant features. The irrelevant and redundant features should be removed prior for building the accurate classifier in order to achieve better performance and reducing complexity of the systems.

Rough Set Theory (RST) [17-20], a purely mathematical approach to imperfect knowledge, is popularly employed for evaluating significance of features and helps to find the important attributes or features in terms of reduct. Feature selection and reduct generation are frequently used as a pre-processing step to data mining and knowledge discovery. It selects an optimal subset of features from the feature space according to a certain evaluation criterion. It has been a fertile field of research and shown very effective in removing irrelevant and redundant features, increasing efficiency in data analysis like clustering [22, 79] and classification techniques [15, 80].

The main advantage of rough set theory in data analysis is that it does not need any preliminary or additional information about data like probability in statistics [36], basic probability assignment in Dempster-Shafer theory [42], grade of membership or the value of possibility in fuzzy set theory [35] and so on. But finding reduct by exhaustive search of all possible combinations of features is an NP-Complete problem and so some heuristic approaches are applied. Rough set based heuristic feature selection algorithm mainly has three components such as feature subset generation, selection of rough sets-based evaluation criterion and the termination condition. Feature subset generation is a process that either starts with all features, or some selected features and then features are iteratively removed or added respectively.

The evaluation criterion evaluates the fitness of a feature subset produced by the generation procedure.

Many evaluation criteria are designed based on rough set theory such as distinct measures of significance of attributes [199], discernibility matrix [200] based algorithm, dependency based [201] algorithm, mutual information [202, 203] based algorithm. A typical terminal condition depends on the evaluation criterion to terminate the process when a certain feature subset is reached. After satisfying the terminal condition, a selected feature subset is generated as the output. In rough set theory, the feature subset is termed as reduct. In reality, there are multiple reducts in a given information system used for developing classifiers, amongst which the best performer is chosen as the final solution to the problem. But this is not always true and according to the Occam's razor and minimal description length principle [204-206], the minimal reduct is preferred. However, Roman [207] has found that the minimal reduct is good for ideal situations where a given dataset fully represents a domain of interest. But for real life situations and limited size datasets, those other than the minimal reducts might be better for prediction. Selecting a reduct with good performance is time expensive, as there might be many reducts of a given dataset. Hu et al. [208] developed two new algorithms to compute core attributes and reducts for feature selection from the dataset. The algorithm can be extensively applied to a wide range of real-life applications with very large datasets. Jensen et al. [209] developed the Quick-reduct algorithm to compute a minimal reduct exclusive of exhaustively creating all feasible subsets and also developed Fuzzy-Rough attribute reduction with application to web categorization. Zhong et al. [210] applies Rough Sets with Heuristics (RSH) and Rough Sets with Boolean Reasoning (RSBR) for reduction of attributes and discretization of real-valued attributes. Komorowska et al. [211] developed an application of rough sets to modeling prognostic power of cardiac tests. Carlin et al. [212] presents an application of rough sets to diagnosing suspected acute appendicitis.

In this chapter, four different approaches to feature selection methods [43, 44, 47, 48] based on filter approach [86] have been proposed, each of which has its novelty in feature selection.

The first method [43] generates single reduct using the property of relative indiscernibility of RST called *single reduct generation using RST* (SRG) method. The second method [44], called *generation of reduct constructing directed minimal spanning tree* (GRG) using the concept of relative indiscernibility relation of RST and Minimal Spanning Tree (MST). Next, RST based multiple reduct generation algorithm, called *compact reduct set generation by forward selection and backward removal techniques* (FSBR) [47] based on the concept of indiscernibility relation and attribute dependency of Rough Set Theory is proposed. Finally *multiple reduct generation algorithm by integrating clustering algorithm and RST* (MRG) [48] is proposed using the concepts of rough set theory, graph theory and clustering algorithm.

There are many existing standard approaches [12, 13, 75-78, 86] used for feature selection as well as dimensionality reduction of data. The common standard dimension reduction methods 'Cfs Subset Eval'(CFS) [95], 'Consistency Subset Evaluator'(CON) [213], 'Classical Attribute Reduction based on Shannon's information entropy'(CAR) [214], Relief-F [215], Singular value Decomposition (SVD) [216] and one popular genetic algorithm-based feature selection methods such as MOGA [217] have been considered in the thesis for performance analysis of the proposed methods. CFS method assesses the predictive ability of each attribute individually and the degree of redundancy among them. The method iteratively adds attributes that have the higher correlation with the class, provided that the set does not already contain an attribute whose correlation with the attribute in question is even higher.

On the other hand, CON method evaluates attribute sets by the degree of consistency in class values when the training instances are projected onto the set. CON seeks the smallest subset whose consistency is the same as that of the full attribute set. CAR is a common heuristic classical attribute reduction algorithm based on information entropy for decision tables. The method selects the core attributes by checking the significance of all attributes using the concept of Shannon's conditional entropy and based on the core attributes, CAR algorithm can find reduct by gradually adding selected attributes to the core. Relief-F is an existing feature selection algorithm that searches for the nearest neighbors of the objects of different classes and assigns weight to the features according to how effectively they discriminate objects of different classes. Singular value decomposition (SVD) [216] is a widely used method of feature construction. The goal of SVD is to form a set of features that are linear combinations of the original features, which provide the best possible reconstruction of the original data in the least square sense. A multi objective genetic algorithm (MOGA) [217] is used for feature selection of Microarray datasets which optimizes three objectives: maximize the sensitivity, maximize the specificity, and minimize the number of genes. The dimension reduction methods CFS [95], CON [213] and Relief-F [215] are available at "Weka" tool [218] while CAR [214], SVD [216] and MOGA [217] are implemented in Matlab.

Results of the existing and the proposed feature selection methods are evaluated and compared on the basis of classification accuracies and other statistical measures by the considered base classifiers to judge the effectiveness of the proposed methods for experimental benchmark datasets [27, 28].

In our work, considered base classifiers are Naïve Bayes (NB) [7], Support vector machine (SVM) [6], K-nearest neighbors K-NN [37], Bagging [191], Tree based classifier (J48) [5], Multilayer Perceptron (MLP) [3] and an incremental classifier IPSO [63]. SVM is used with RBF Kernel, K value of K-NN is set to the square root of sample size of data.

The specification of the computer in these experiments are, Computer Model: ACER machines D725; CPU: Pentium(R) Dual-Core CPU T4400 @ 2.20GHz × 2; Memory: 1GB; OS: Ubuntu 12.04 LTS - 32 bit.

The remaining part of the chapter is organized as follows: The single feature subset selection methods based on RST concepts and graph theory are described and their performances are compared in Section 3.2.

In Section 3.3, multiple feature subset selection methods based on RST, graph theory, and clustering algorithm are described and their experimental results are evaluated and compared. Finally, the chapter is summarized in Section 3.4.

3.2 Single Feature Subset Selection:

The important and relevant feature selection [86] is primarily very important in data mining research as the irrelevant features degrade the classification accuracy. Accurate prediction can be achieved only by identifying the most informative features(s) from a large feature space by removing irrelevant and redundant feature.

As finding important features by exhaustive search of all possible combination of features is an NP-complete problem [156], so efficient heuristics are proposed [86] in important feature selection. In this section, two single feature subset selection methods [43, 44] are proposed by which single reduct is generated from the dataset.

First method [43] provides single reduct based on the concepts of RST only while the other method [44] generates single reduct based on the concepts of RST and graph theory. In the subsequent sections the terms attribute and feature has been used synonymously.

3.2.1 Single Reduct Generation Using Rough Set Theory (SRG):

Feature selection methodology is essential to determine features responsible for classifying any object, which be included in learning network and provide information about class related features. Successful feature selection method helps to classify different objects, lead to a better understanding of the internal structures of the data. Here, a feature selection method (SRG) [43] has been proposed for selecting a single important feature subset preserves the property of the whole decision system as reduct of the decision system for classifying objects of the datasets.

In the method, a new kind of indiscernibility, called relative indiscernibility of an attribute with respect to another attribute is introduced. In SRG, relative indiscernibility gives relative indiscernible objects based on a feature, relative to decision attribute.

To find relative indiscernible objects based on a feature, the dataset is partitioned using that feature and the decision attribute separately. The partitions (using that feature and the decision attribute) with same set of objects are placed in the same class.

To obtain the minimal feature subset using SRG method which can fully characterize the dataset, following steps are performed:

- Define a Relative Indiscernibility relation for each conditional attribute relative to decision attribute
- Measure attribute similarity between every pair of attributes and compute the similarity factors
- Find the Attribute Similarity Set ASS = {Ai®Aj} using computed similarity factors
- Modify the Attribute Similarity Set removing weakly similar pair of attributes
- Compute the single reduct

The detail procedure of reduct generation is discussed below.

a. Define a Relative Indiscernibility Relation for a Conditional Attribute Relative to Decision Attribute:

To understand the concept, a decision system *DS* is considered as $DS = (U, A)$ where U is the universe (a finite set of objects, $U = \{x_1, x_2, ., x_n\}$) and A is the set of attributes such that $A = C \cup D$ and $C \cap D = \emptyset$ where C and D are the set of conditional attributes and the decision attributes, respectively.

For each attribute $a \in A$ defines an information function: f_a: $U \rightarrow V_a$, where V_a is the set of values of a, called the domain of attribute. Every subset of attributes P determines an indiscernibility relation over U, and is denoted as $IND(P)$, which can be defined by Equation (2.7).

$IND(P)$ is called the P-indiscernibility relation and the equivalence classes of the P-indiscernibility relation are denoted by $[x]_P$.

Here relative indiscernibility relation gives relative indiscernible objects based on each conditional attribute, relative to decision attribute D. Every conditional attribute A_i of C determines a relative indiscernibility relation (RIR) over U relative to D, and is denoted as $RIR_D(A_i)$, which can be defined by Equation (3.1).

$$RIR_D(A_i) = \{(x, y) \in \Pi_{A_i}[x]_D \times \Pi_{A_i}[x]_D \mid f_{A_i}(x) = f_{A_i}(y) \forall [x]_D \in U/D\} \quad (3.1)$$

Example: To illustrate the concept of relative indiscernibility, a sample dataset is considered in Table 3.1 with eight objects $\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$, four conditional attributes $\{i, e, l, r\}$ and one decision attribute (D) with their respective values.

Table 3.1: Sample Dataset

Objects	***Diploma* (*i*)**	***Experience* (*e*)**	***French* (*l*)**	***Reference* (*r*)**	***Decision* (*D*)**
x_1	MBA	Medium	Yes	Excellent	Accept
x_2	MBA	Low	Yes	Neutral	Reject
x_3	MCE	Low	Yes	Good	Reject
x_4	MSc	High	Yes	Neutral	Accept
x_5	MSc	Medium	Yes	Neutral	Reject
x_6	MSc	High	Yes	Excellent	Reject
x_7	MBA	High	No	Good	Accept
x_8	MCE	Low	No	Excellent	Reject

Here, equivalence classes by indiscernibility relation IND (P) defined in Equation (2.7) for each attribute are:

$U/D = \{\{x_1, x_4, x_7\}, \{x_2, x_3, x_5, x_6, x_8\}\}$

$U/i = \{\{x_1, x_2, x_7\}, \{x_3, x_8\}, \{x_4, x_5, x_6\}\}$

$U/e = \{\{x_1, x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_6, x_7\}\}$

$U/l = \{\{x_1, x_2, x_3, x_4, x_5, x_6\}, \{x_7, x_8\}\}$

$U/r = \{\{x_1, x_6, x_8\}, \{x_2, x_4, x_5\}, \{x_3, x_7\}\}$

The equivalence classes by relative indiscernibility relation $RIR_D(A_i)$ defined in Equation (3.1) for each conditional attribute are:

$RIR_D(i) = \{\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\}, \{x_4\}, \{x_5, x_6\}\}$

$RIR_D(e) = \{\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}\}$

$RIR_D(l) = \{\{x_1, x_4\}, \{x_2, x_3, x_5, x_6\}, \{x_7\}, \{x_8\}\}$

$RIR_D(r) = \{\{x_1\}, \{x_6, x_8\}, \{x_2, x_5\}, \{x_4\}, \{x_{3\}}, \{x_7\}\}$

b. Attribute Similarity MeasurementL:

An attribute A_i is similar to another attribute A_j in context of classification of objects if they induce the same equivalence classes of objects under their respective relative indiscernible relations. But in real situation, it rarely occurs and so similarity of attributes is measured by introducing the similarity measurement factor which indicates the degree of similarity of one attribute to another attribute. Here, an attribute A_i is said to be similar to an attribute A_j with degree of similarity (or similarity factor) $\delta_f^{i,j}$ and is denoted by $A_i \rightarrow A_j$ if the probability of inducing the same equivalence classes of objects under their respective relative indiscernible relations is ($\delta_f^{i,j} \times 100$)%, where $\delta_f^{i,j}$ is computed by Equation (3.2).

$$\delta_f^{i,j} = \frac{1}{|U_D/A_i|} \sum_{[x]_{A_{i/D}} \in U_D/A_i} \frac{1}{|[x]_{A_{i/D}}|} \max_{[x]_{A_{j/D}} \in U_D/A_j} ([x]_{A_{i/D}} \cap [x]_{A_{j/D}}) \quad (3.2)$$

The details for computation of similarity measurement for the attribute similarity $A_i \rightarrow A_j$ ($A_i \neq A_j$) is described in algorithm "SIM_FAC" below.

Algorithm: SIM_FAC (A_i, A_j) /* Similarity factor computation for $A_i \rightarrow A_j$ */

Input: Attributes A_i and A_j

Output: Similarity factor $\delta_f^{i,j}$

Begin

for each conditional attribute A_i do

Compute relative indiscernibility *RIRD* (A_i) using Equation (3.1)

RIR_D (A_i) induces equivalence classes $U_D/A_i = \{[x]_{A_{i/D}}\}$

end-for

/* similarity measurement of A_i to A_j */
$\delta_f^{i,j} = 0$
for each $[x]_{i/D} \in U_D/A_i$ do
max_overlap = 0
for each $[x]_{j/D} \in U_D/A_j$
overlap = $|\ [x]_{i/D} \cap [x]_{j/D}\ |$
if (overlap > max_overlap) then
max_overlap = overlap
end-for
$\delta_f^{i,j} = \delta_f^{i,j} + \frac{\max\_overlap}{|\ [x]_{i/D}\ |}$
end-for
$\delta_f^{i,j} = \frac{\delta_f^{i,j}}{|\ U_D/A_i\ |}$
End.

To illustrate the attribute similarity computation process, attribute similarity and its similarity factor are listed in Table 3.2 for all attributes of Table 3.1.

Table 3.2: Degree of similarity of all pair of attributes

Attribute Similarity ($A_i \rightarrow A_j$)	**Equivalence Classes by $RIR_D(A_i)$ (U_D/A_i)**	**Equivalence Classes by $RIR_D(A_j)$ (U_D/A_j)**	**Similarity factor of A_i to A_j ($\delta_f^{i,j}$)**
$i \rightarrow e$	$\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\}, \{x_4\}, \{x_5, x_6\}$	$\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}$	$\delta_f^{i,e} = 0.80$
$i \rightarrow l$	$\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\}, \{x_4\}, \{x_5, x_6\}$	$\{x_1, x_4\}, \{x_2, x_3, x_5, x_6\}, \{x_7\}, \{x_8\}$	$\delta_f^{i,l} = 0.80$
$i \rightarrow r$	$\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\}, \{x_4\}, \{x_5, x_6\}$	$\{x_1\}, \{x_6, x_8\}, \{x_2, x_5\}, \{x_4\}, \{x_3\}, \{x_7\}$	$\delta_f^{i,r} = 0.70$
$e \rightarrow i$	$\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}$	$\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\}, \{x_4\}, \{x_5, x_6\}$	$\delta_f^{e,i} = 0.83$
$e \rightarrow l$	$\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}$	$\{x_1, x_4\}, \{x_2, x_3, x_5, x_6\}, \{x_7\}, \{x_8\}$	$\delta_f^{e,l} = 0.83$
$e \rightarrow r$	$\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}$	$\{x_1\}, \{x_6, x_8\}, \{x_2, x_5\}, \{x_4\}, \{x_3\}, \{x_7\}$	$\delta_f^{e,r} = 0.76$
$l \rightarrow i$	$\{x_1, x_4\}, \{x_2, x_3, x_5, x_6\}, \{x_7\}, \{x_8\}$	$\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\}, \{x_4\}, \{x_5, x_6\}$	$\delta_f^{l,i} = 0.75$
$l \rightarrow e$	$\{x_1, x_4\}, \{x_2, x_3, x_5, x_6\}, \{x_7\}, \{x_8\}$	$\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}$	$\delta_f^{l,e} = 0.75$
$l \rightarrow r$	$\{x_1, x_4\}, \{x_2, x_3, x_5, x_6\}, \{x_7\}, \{x_8\}$	$\{x_1\}, \{x_6, x_8\}, \{x_2, x_5\}, \{x_4\}, \{x_3\}, \{x_7\}$	$\delta_f^{l,r} = 0.75$
$r \rightarrow i$	$\{x_1\}, \{x_6, x_8\}, \{x_2, x_5\},$	$\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\},$	$\delta_f^{r,i} = 0.70$

Attribute Similarity ($A_i \rightarrow A_j$)	**Equivalence Classes by $RIR_D(A_i)$ (U_D/A_i)**	**Equivalence Classes by $RIR_D(A_j)$ (U_D/A_j)**	**Similarity factor of A_i to A_j ($\delta_f^{i,j}$)**
	$\{x_4\}, \{x_3\}, \{x_7\}$	$\{x_4\}, \{x_5, x_6\}$	
$r \rightarrow e$	$\{x_1\}, \{x_6, x_8\}, \{x_2, x_5\}, \{x_4\}, \{x_3\}, \{x_7\}$	$\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}$	$\delta_f^{r,i} = 0.70$
$r \rightarrow l$	$\{x_1\}, \{x_6, x_8\}, \{x_2, x_5\}, \{x_4\}, \{x_3\}, \{x_7\}$	$\{x_1, x_4\}, \{x_2, x_3, x_5, x_6\}, \{x_7\}, \{x_8\}$	$\delta_f^{r,l} = 0.80$

The computation of $\delta_f^{i,j}$ of each attribute similarity using Equation (3.2) in Table 3.2 can be understood by Table 3.3, in which similarity $i \rightarrow e$ in first row of Table 3.2 is considered, where, $U_D/i = \{\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\}, \{x_4\}, \{x_5, x_6\}\}$ and $U_D/e = \{\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}\}$.

Table 3.3: Similarity factor computation for $i \rightarrow e$

$[x]_{i/D}$ of U_D/i	**Overlapping $[x]_{e/D}$ of U_D/e with $[x]_{i/D}$ of U_D/i**	$[x]_{i/D} \cap [x]_{e/D}$	$T = \frac{1}{\lvert[x]_{i/D}\rvert} \max_{[x]_{e/D} \in U_D/e} ([x]_{i/D} \cap [x]_{e/D})$
$\{x_1, x_7\}$	$\{x_1\}$ $\{x_4, x_7\}$	$\{x_1, x_7\} \cap \{x_1\}$ $\{x_1, x_7\} \cap \{x_4, x_7\}$	$\frac{1}{2}$
$\{x_2\}$	$\{x_2, x_3, x_8\}$	$\{x_2\} \cap \{x_2, x_3, x_8\}$	$\frac{1}{1}$
$\{x_3, x_8\}$	$\{x_2, x_3, x_8\}$	$\{x_3, x_8\} \cap \{x_2, x_3, x_8\}$	$\frac{2}{2}$
$\{x_4\}$	$\{x_4, x_7\}$	$\{x_4\} \cap \{x_4, x_7\}$	$\frac{1}{1}$
$\{x_5, x_6\}$	$\{x_5\}$ $\{x_6\}$)	$\{x_5, x_6\} \cap \{x_5\}$ $\{x_5, x_6\} \cap \{x_6\}$	$\frac{1}{2}$
$\delta_f^{ie} = \frac{1}{\lvert[x]_{i/D}\rvert} \sum_{[x]_{i/D} \in U_D/i} T = \frac{1}{5} (\frac{1}{2} + \frac{1}{1} + \frac{2}{2} + \frac{1}{1} + \frac{1}{2}) = \frac{4}{5} = 0.80$			

c. Computation of Attribute Similarity Set:

For each pair of conditional attributes (A_i, A_j), similarity factor is computed by "SIM_FAC" algorithm, described in section 3.2.1.2. Higher the similarity factor of $A_i \rightarrow A_j$ is higher means that the relative indiscernibility relations $RIR_D(A_i)$ and $RIR_D(A_j)$ produce more similar equivalence classes. This implies that both the attributes A_i and A_j have almost similar

classification power and so $A_i \rightarrow A_j$ is considered as strong similarity of A_i to A_j. Since, for any two attributes A_i and A_j, two similarities $A_i \rightarrow A_j$ and $A_j \rightarrow A_i$ is computed, only one with higher similarity factor is selected in the list of attribute similarity set ASS. Out of these similarities, the similarity with $\delta_f^{i,j}$ less than the average (δ_f) value are discarded from ASS and rest is considered as the modified set of attribute similarity. So, each element x in ASS is of the form x: $A_i \rightarrow A_j$ such that Left(x) = A_i and Right(x) = A_j. The algorithm **"ASS_GEN"** described below, computes the attribute similarity set ASS.

Algorithm: ASS_GEN (C, δ_f) /* Computes attribute similarity set { $A_i \rightarrow A_j$ } */

Input: C = set of conditional attributes and δ_f = 2-D array of size $|C| \times |C|$ contains similarity factors between each pair of conditional attributes obtained by using equation (3.2)

Output: Attribute Similarity Set ASS

```
Begin
 ASS = {}, sum_δf = 0
 /* compute only |C| (|C| – 1)/2 elements in ASS */
 for i = 1 to |C| - 1  do
      for j = i+1 to |C|  do
            if δf(i,j) > δf(j,i) then
                  sum_δf = sum_δf + δf(i,j)
                  ASS = ASS ∪ {Ai → Aj}
            else
                  sum_δf = sum_δf + δf(j,i)
                  ASS = ASS ∪ {Ai → Aj }
            end-if
       end-for
  end-for

   /* modify ASS by only elements Ai → Aj for which δf(i,j) >avg_δf */
 ASSmod = {}

 avg_δf = (2× sum_δf) / |C|(|C|-1)

 for each {Ai → Aj } ∈ ASS  do
      if δf(i,j) >avg_δf then
      ASSmod = ASSmod ∪ {Ai → Aj }
      ASS = ASS – {Ai → Aj }
      end-if
 end-for
 ASS = ASSmod
End
```

For the sample dataset, initially **"ASS_GEN"** algorithm selects ASS = {$i \rightarrow l$, $i \rightarrow r$, $e \rightarrow i$, $e \rightarrow l$, $e \rightarrow r$, $r \rightarrow l$} and construct Table 3.4. As the similarity factor for attribute

similarities $i \rightarrow l$, $e \rightarrow i$, $e \rightarrow l$ and $r \rightarrow l$ is greater than average value 0.786, so modified attribute similarity set ASS = $\{i \rightarrow l, e \rightarrow i, e \rightarrow l, r \rightarrow l\}$.

Table 3.4: Illustrates the selection of attribute similarities

Attribute Similarity ($Ai \rightarrow A_j$; $i \neq j$ and $\delta_f^{i,j} > \delta_f^{j,i}$)	**Similarity factor of A_i to A_j ($\delta_f^{i,j}$)**	**$\delta_f^{i,j}$ >Average value**
$i \rightarrow l$	$\delta_f^{i,l} = 0.80$	Yes
$i \rightarrow r$	$\delta_f^{i,r} = 0.70$	
$e \rightarrow i$	$\delta_f^{e,i} = 0.83$	Yes
$e \rightarrow l$	$\delta_f^{e,l} = 0.83$	Yes
$e \rightarrow r$	$\delta_f^{e,r} = 0.76$	
$r \rightarrow l$	$\delta_f^{r,l} = 0.80$	Yes
Average δ_f	0.786	

d. Generation of Modified Attribute Similarity Set:

The attribute similarity obtained so far is known as simple similarity of an attribute to another attribute. But, for simplifying the reduct generation process, the elements in ASS are minimized by combining some simple similarities. The new similarity obtained by combination of the simple similarities is called compound similarity. Here, all x from ASS with same Left(x) are considered and obtained compound similarity is Left(x) $\rightarrow \cup$ Right(x) $\forall x$. Thus, introducing compound similarity, the set ASS is refined to a set with minimum elements so that for each attribute, there is at most one element in ASS representing either simple or compound similarity of the attribute. The detail algorithm for determining compound attribute similarity set is given below in "COMP_SIM" algorithm:

Algorithm: COMP_SIM (ASS)

/* Compute the compound attribute similarity of attributes*/
Input: Simple attribute similarity set ASS
Output: Compound attribute similarity set CSS
Begin
for each $x \in$ ASS do
for each y ($\neq x$) $\in$ ASS do
if Left(x) = Left(y) then
Right(x) = Right(x) $\cup$ Right(y)
ASS = ASS – $\{y\}$
end-if
end-for
end-for
CSS=ASS

End

e. Generation of Single Reduct:

Finally, from the compound attribute similarity set CSS, reduct is generated. First of all, select an element, say, x from CSS for which length of Right(x) i.e., |Right(x)| is the maximum. This selection assures that the attribute Left(x) is similar to maximum number of attributes and so Left(x) is an element of reduct *RED*. Then, all elements z of CSS for which Left(z) $\subseteq$ Right(x) are deleted and also x is deleted from CSS. This process is repeated until the set CSS becomes empty which provides the reduct *RED*. The single reduct generation algorithm "SIN_RED_GEN" is given below:

Algorithm: SIN_RED_GEN (CSS, *RED*)

Input: Compound attribute similarity set CSS
Output: Single reduct *RED*

```
Begin
    RED = ϕ
    While (CSS ≠ ϕ)   do
     max = 0
       for each x ∈ CSS   do
            if |Right(x)| > max then
              max = |Right(x)|
                 L = Left(x)
           end-if
       end-for
       for each x ∈ CSS   do
        if Left(x) = L then
                 RED = RED ∪ Left(x)
                 R = Right(x)
                 CSS = CSS – {x}
                 for each z ∈ ASS do
                    if Left(z) ⊆ R then
                     CSS = CSS – {z}
                 end-for
                    break
           end-if
       end-for
    end-while
   Return (RED)
End
```

Applying "COMP_SIM" algorithm on the sample dataset the set ASS = $\{i \rightarrow l, e \rightarrow i, e \rightarrow l, r \rightarrow l\}$ is refined to compound similarity set CSS = $\{i \rightarrow l, e \rightarrow \{i, l\}, r \rightarrow l\}$. So, the selected element from CSS is $e \rightarrow \{i, l\}$, and by applying "SIN_RED_GEN" algorithm, $e \in RED$ and CSS is modified as CSS = $\{r \rightarrow l\}$. And, in the next iteration, $r \in RED$ and CSS =ϕ. Thus, *RED* = $\{e, r\}$ is found for the sample dataset.

f. Experimental Results of the SRG Method:

The proposed method computes a single reduct for experimental benchmark dataset selected from UCI machine learning repository [27] and repository related to feature selection methods [28] mentioned in the section 2.2. At first, all the attributes of the dataset are discretized by ChiMerge [219] discretization algorithm. Then SRG method [43] and some other well-known feature selection methods such as, CFS [95], CON [213], CAR [214], Relief-F [215], SVD [216] and MOGA [217] are applied on the dataset for selecting the important features and the reduced datasets are classified based on considered base classifiers. 10-fold cross validation is used for the classification performance evaluation. Number of attributes after applying proposed and existing feature selection methods and the accuracies (%) of the base classifiers are computed and listed in Table 3.5, which shows the efficiency of the proposed method. To measure the statistical significance of the proposed SRG method [43], wilcoxon's rank sum test [39] is carried out with p value as 0.05 (or a significance level of 5%) to validate if the result obtained by the best performing algorithm differs from the others in a statistically significant way. The test confirms if the final accuracy obtained by an algorithm is statistically and significantly different from that of the best performing algorithm on some classification problem. Thus, if the performance of an algorithm is differing from the best result with a p value ≤ 0.05 then the mean error of the first one is marked with a '†' symbol, otherwise the two performances are considered as equivalent and the difference is not statistically significant, and mark the mean error with a '≈' symbol, as shown in Table 3.5. To indicate the best performing algorithm, we use a bold-faced font to write its mean.

Table 3.5: Performance analysis of proposed SRG feature selection method

Dataset	Methods (#features)	Classifiers (%)						
		NB	SVM	KNN	Bagging	J48	MLP	IPSO
Wine	CFS (8)	97.19†	97.21†	97.45†	94.94†	93.82†	93.10†	95.70†
	CON (8)	97.19†	97.31†	96.63†	94.94†	94.94†	94.30†	97.65†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.56†
	Relief-F (9)	97.19†	97.31†	96.63†	94.94†	94.97†	94.40†	97.23†
	SVD (6)	96.65†	97.56†	96.76†	95.78†	**97.45**	96.98†	97.80†
	MOGA (7)	97.87†	96.65†	95.56†	97.64≈	96.78†	96.87†	97.67†
	SRG (6)	**98.70**	**98.40**	**97.39**	**97.86**	97.00	**97.50**	**98.40**
Heart	CFS (8)	84.36≈	**84.75**	81.67†	81.11†	81.11†	81.67†	82.78†
	CON (11)	84.50≈	84.44≈	82.07†	81.48†	82.89†	79.55†	82.72†
	CAR (10)	83.36†	**84.75**	81.67†	83.11≈	82.11†	80.67†	82.34†
	Relief-F (10)	83.50†	84.44≈	82.07†	81.48†	**83.89**	79.59†	82.30†
	SVD (4)	83.45†	83.67†	**83.98**	82.21†	83.21≈	82.08†	83.87†
	MOGA (8)	84.67≈	83.32†	83.22†	**83.56**	83.87≈	**84.76**	**84.98**
	SRG (4)	**84.77**	83.77†	83.90≈	83.40≈	83.31≈	83.70†	84.51≈
Glass	CFS (6)	43.92†	57.94†	79.91†	73.83†	68.69†	70.09†	66.02†
	CON (7)	47.20†	57.48†	78.50†	71.50†	64.20†	68.60†	64.65†
	CAR (8)	56.92†	58.94†	80.91†	75.83†	69.69†	71.09†	68.54†

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
	Relief-F (8)	57.20†	57.48†	79.50†	70.50†	63.20†	72.60†	67.74†
	SVD (6)	56.67†	57.75†	77.39†	71.56†	67.50†	74.45†	75.89†
	MOGA (7)	56.54†	57.76†	76.49†	72.45†	64.76†	70.89†	76.23†
	SRG (6)	**65.73**	**62.44**	**83.57**	**76.53**	**72.30**	**77.00**	**77.78**
Zoo	CFS (9)	96.03≈	93.06†	94.05†	94.04†	93.06†	93.06†	93.54†
	CON (9)	96.03≈	93.03†	94.05†	94.04†	93.88†	94.32†	94.45†
	CAR (6)	94.05†	93.92†	93.32†	94.02†	94.07†	94.05†	95.45†
	Relief-F (7)	95.03†	93.70†	93.01†	93.01†	94.12†	94.12†	95.03†
	SVD (8)	**96.09**	94.67†	94.78†	94.67†	94.32†	94.67†	95.56†
	MOGA (6)	94.78†	94.23†	94.45†	95.21≈	94.32†	94.34†	94.54†
	SRG (8)	96.01≈	**95.04**	**95.05**	**95.48**	**95.89**	**95.12**	**96.56**
Dermatology	CFS (9)	98.76†	97.42†	97.01†	98.06†	98.07†	98.62†	99.09≈
	CON (9)	98.52†	98.25†	95.56†	98.06†	98.86†	98.67†	98.45†
	CAR (11)	98.73†	98.30†	97.42†	98.31†	98.06†	98.07†	98.54†
	Relief-F (11)	98.72†	98.45†	95.56†	97.16†	98.76†	98.46†	98.45†
	SVD (9)	97.78†	98.03†	96.75†	97.09†	98.01†	97.89†	97.65†
	MOGA (9)	97.87†	97.89†	97.90†	97.05†	98.67†	97.90†	97.50†
	SRG (9)	**99.40**	**99.27**	**98.45**	**99.48**	**99.41**	**99.33**	**99.13**
Mushroom	CFS (4)	97.52†	96.01†	96.52†	97.01†	97.01†	97.01†	97.04†
	CON (5)	**98.52**	**98.85**	98.52†	99.05≈	98.16†	**99.86**	98.54≈
	CAR (8)	98.02≈	98.32≈	99.02≈	**99.65**	**99.23**	99.01≈	98.45≈
	Relief-F (5)	97.04†	98.03≈	98.03†	98.13†	98.10†	98.10†	98.23≈
	SVD (4)	97.04†	97.23†	97.23†	97.83†	97.34†	87.64†	97.45†
	MOGA (5)	97.34†	95.45†	96.67†	96.34†	96.34†	96.45†	97.64†
	SRG (4)	98.34≈	98.82≈	**99.04**	97.78†	98.89†	99.08≈	**98.78**
Coil20	CFS (194)	78.12†	79.24†	80.01†	80.10†	79.25†	79.85†	80.98†
	CON (194)	79.60†	79.20†	78.00†	79.35†	79.34†	78.79†	80.21†
	CAR (201)	77.12†	78.24†	77.01†	80.10†	75.25†	78.85†	78.98†
	Relief-F (198)	77.60†	78.20†	77.80†	78.35†	77.34†	78.49†	80.21†
	SVD (123)	76.60†	76.20†	77.80†	75.35†	76.64†	78.49†	79.21†
	MOGA (95)	82.20†	79.99†	82.92†	83.02†	83.02†	84.20†	84.71†
	SRG (123)	**87.87**	**87.98**	**89.31**	**90.16**	**90.45**	**90.50**	**90.65**
Orl	CFS (201)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	57.65†
	CON (198)	56.60†	53.20†	52.00†	51.35†	50.70†	52.89†	54.87†
	CAR (204)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	57.65†
	Relief-F (213)	54.60†	54.20†	54.01†	54.25†	53.79†	53.89†	54.87†
	SVD (132)	52.12†	53.24†	53.01†	52.10†	53.80†	54.35†	57.65†
	MOGA (110)	59.60†	57.20†	59.00†	58.25†	59.70†	59.09†	59.87†

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
	SRG (132)	**61.89**	**60.98**	**62.06**	**61.47**	**62.07**	**63.03**	**63.43**
Allaml	CFS (221)	81.12†	81.32†	82.62†	**83.82**	82.21≈	83.01†	83.80†
	CON (230)	79.12†	80.32†	81.62†	82.82†	80.21†	82.01†	83.80†
	CAR (201)	80.02†	80.32†	81.52†	82.72†	81.21†	82.02†	82.80†
	Relief-F (210)	81.12†	81.32†	82.62†	**83.82**	82.21≈	83.11†	83.80†
	SVD (201)	80.22†	81.62†	81.27†	82.92†	**82.81**	83.19†	82.80†
	MOGA (104)	81.12†	83.32	**84.72**	81.82†	82.01≈	82.21†	84.50≈
	SRG (201)	**82.12**	**83.32**	84.62≈	**83.82**	82.21≈	**84.01**	**84.80**
Leukaemia	CFS (147)	83.23†	84.23†	82.67†	82.02†	82.05†	83.23†	84.34†
	CON (159)	84.05†	85.87†	84.23†	84.12†	84.21†	84.67†	85.01†
	CAR (147)	83.47†	85.23†	84.56†	84.53†	84.54†	84.56†	89.10†
	Relief-F (159)	84.23†	85.34†	84.12†	84.34†	84.34†	84.98†	87.50†
	SVD (102)	73.68†	76.32†	71.05†	71.02†	71.05†	73.68†	75.87†
	MOGA (97)	86.34†	87.90†	86.50†	85.78†	88.12†	88.23†	89.80†
	SRG (102)	**88.01**	**89.12**	**88.23**	**88.78**	**89.43**	**89.12**	**90.42**

The method also determines some statistical measurements defined in the Equation (2.25) to Equation (2.28) in the section 2.6.2 and the average results for all seven classifiers are listed in Table 3.6 to demonstrate the effectiveness of the method.

Table 3.6: Statistical measures for SRG and different competitive algorithm

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Wine	CFS (8)	0.96	0.04	0.95	0.96
	CON (8)	0.96	0.04	0.96	0.96
	CAR (8)	0.95	0.06	0.94	0.95
	Relief-F (9)	0.96	0.04	0.96	0.95
	SVD (6)	0.97	0.03	0.96	0.97
	MOGA (7)	0.97	0.03	0.97	0.97
	SRG (6)	**0.98**	**0.02**	**0.97**	**0.98**
Heart	CFS (8)	**0.98**	**0.02**	**0.98**	**0.97**
	CON (11)	0.82	0.18	0.83	0.82
	CAR (10)	0.83	0.16	0.82	0.83
	Relief-F (10)	0.83	0.17	0.83	0.83
	SVD (4)	0.82	0.17	0.82	0.83
	MOGA (8)	0.83	0.18	0.82	0.82
	SRG (4)	0.84	0.16	0.83	0.84
	CFS (6)	0.66	0.36	0.67	0.66

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Glass	CON (7)	0.65	0.36	0.65	0.64
	CAR (8)	0.69	0.31	0.68	0.69
	Relief-F (8)	0.68	0.32	0.69	0.68
	SVD (6)	0.69	0.31	0.69	0.68
	MOGA (7)	0.68	0.32	0.68	0.68
	SRG (6)	**0.74**	**0.25**	**0.74**	**0.74**
Zoo	CFS (9)	0.94	0.06	0.93	0.94
	CON (9)	0.94	0.06	0.94	0.93
	CAR (6)	0.94	0.06	0.92	0.94
	Relief-F (7)	0.94	0.05	0.94	0.93
	SVD (8)	0.95	0.05	**0.95**	0.95
	MOGA (6)	0.95	**0.04**	**0.95**	0.94
	SRG (8)	**0.96**	**0.04**	**0.95**	**0.96**
Dermatology	CFS (9)	0.98	**0.01**	**0.99**	**0.98**
	CON (9)	0.98	**0.01**	0.98	0.97
	CAR(11)	0.98	0.02	0.97	**0.98**
	Relief-F (11)	0.98	0.02	0.98	**0.98**
	SVD (9)	0.98	0.02	0.98	0.97
	MOGA (9)	0.98	0.02	0.97	**0.98**
	SRG (9)	**0.99**	**0.01**	**0.99**	**0.98**
Mushroom	CFS (4)	0.97	0.03	0.96	0.97
	CON (5)	**0.99**	**0.01**	0.98	0.99
	CAR (8)	**0.99**	**0.01**	**0.99**	**0.98**
	Relief-F (5)	0.98	0.02	0.97	0.98
	SVD (4)	0.96	0.04	0.96	0.95
	MOGA (5)	0.97	**0.01**	0.96	0.97
	SRG (4)	**0.99**	**0.01**	**0.99**	**0.98**
Coil20	CFS (194)	0.80	0.19	0.81	0.80
	CON (194)	0.79	0.21	0.79	0.78
	CAR (201)	0.78	0.21	0.78	0.79
	Relief-F (198)	0.78	0.22	0.77	0.78
	SVD (123)	0.77	0.23	0.77	0.78
	MOGA (95)	0.83	0.16	0.83	0.83
	SRG (123)	**0.90**	**0.09**	**0.90**	**0.91**
	CFS (201)	0.54	0.45	0.53	0.54
	CON (198)	0.53	0.45	0.53	0.52

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Orl	CAR (204)	0.53	0.46	0.52	0.53
	Relief-F (213)	0.54	0.46	0.54	0.53
	SVD (132)	0.54	0.46	0.54	0.54
	MOGA (110)	0.59	0.37	0.60	0.59
	SRG (132)	**0.62**	**0.38**	**0.62**	**0.61**
Allaml	CFS (221)	0.82	**0.17**	0.82	**0.82**
	CON (230)	**0.83**	0.18	**0.83**	**0.82**
	CAR (201)	0.81	0.19	0.80	0.81
	Relief-F (210)	0.81	0.19	0.81	**0.82**
	SVD (201)	0.82	0.18	0.81	**0.82**
	MOGA (104)	0.82	0.18	0.82	0.81
	SRG (201)	0.82	0.18	0.82	0.81
Leukemia	CFS (147)	0.83	0.16	0.81	0.83
	CON (159)	0.85	0.13	0.85	0.84
	CAR (147)	0.85	0.14	0.85	0.85
	Relief-F (159)	0.85	0.15	0.86	0.85
	SVD (102)	0.73	0.23	0.73	0.77
	MOGA (97)	0.88	0.12	0.87	**0.88**
	SRG (102)	**0.89**	**0.10**	**0.89**	**0.88**

From Table 3.5 and 3.6, it is proved that the proposed SRG method is better than most of the existing standard feature selection methods.

3.2.2 Generation of Reduct Constructing Directed Minimal Spanning Tree using Rough Set Theory (GRG):

This section describes a new method [44] of feature selection using the concepts of Rough Set Theory [17-20] and Graph Theory [21] (GRG). Here, the data mining problem is converted to graph theoretic problem and then single minimal feature subset (called reduct in RST) is generated. Many feature selection techniques use heuristics which may degrade the performance, but the proposed GRG method has a strong mathematical foundation and hence, produces good results.

The GRG method computes relative indiscernibility of the conditional attributes relative to the decision attribute using the Equation (3.1) given in section 3.2.1.1, which helps to measure the degree of similarity among the conditional attributes, using the Equation (3.2) given in section 3.2.1.2. Based on the similarity of attributes a weighted directed graph is generated and a minimal spanning tree of the graph is obtained which finally gives the reduct.

The overall flow diagram of the GRG method [44] is shown in Figure 3.1.

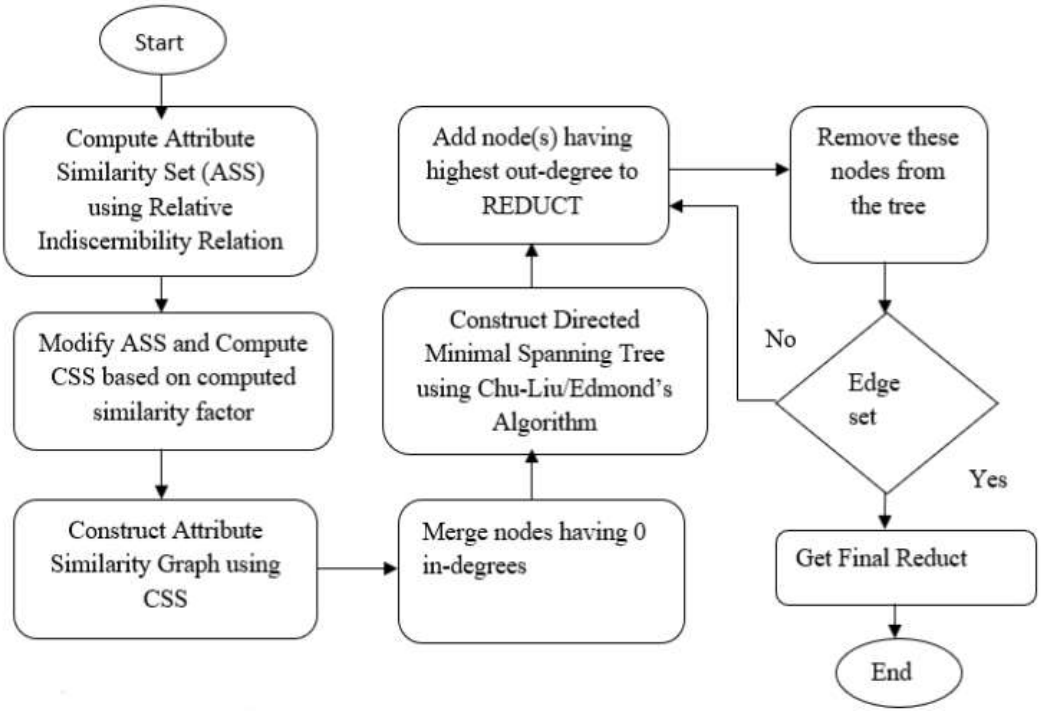


Figure 3.1: Flow diagram of GRG

Before discussing detail of the method, some basic concepts on rooted directed minimal spanning tree and Chu-Liu/Edmond's (CLE) Algorithm [46] is described in the following section.

a. Rooted Directed Minimal Spanning Tree Algorithm:

Generally, Prim's [220] or Kruskal's [221] algorithm is used to find out the minimal spanning tree (MST) from an undirected graph. But these two methods do not give the optimal result in case of a directed graph.

The Figure 3.2 shows that the tree, constructed by performing iterative greedy decision of the Prim's algorithm, is not a minimal spanning tree of the directed graph. Chu and Liu [46], Edmonds [222] and Bock [223] have independently given efficient algorithms for obtaining the MST on a directed graph.

The Chu-Liu and Edmonds algorithms are virtually identical; the Bock algorithm is similar but defined on matrices instead of on the graphs. Moreover, a distributed algorithm is proposed by Humblet [224].

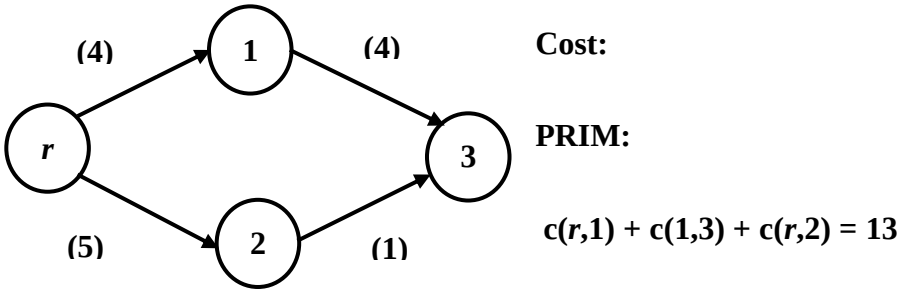


Figure 3.2: MST construction for directed graph using Prim's algorithm

The rooted spanning tree (directed) is defined as a graph which attaches, without any cycle, all vertices with n-1 edges, i.e., every vertex, except the root, has one and only one incoming edge.

Suppose, $G = (V, E)$, be a directed graph where V and E are the set of vertices and edges, respectively. A cost $c(i, j)$ is connected with each edge (i, j) between vertices i and j in E.

Let, $|V|=n$ and $|E|=m$. The algorithm is explained briefly by the following steps, which calculates a rooted minimal spanning tree $MST(V, S)$ of the graph $G(V, E)$ where S is a sub set of E such that $\sum c(i, j), \forall (i, j)$ in S is minimized.

- **Chu-Liu/Edmond's (CLE) Algorithm:**
  i. The edges entering the root if any are discarded; for each vertex other than the root, the entering edge with the smallest cost is selected. Let the selected n-1 edges be the set S.
  ii. If no cycle formed, $MST(V, S)$ is a Minimal Spanning Tree. Otherwise, go to step (iii).
  iii. For each cycle formed, contract the vertices in the cycle into a single new vertex k modifying the cost of each edge which enters a vertex j in the cycle from some vertex i outside the cycle, according to the Equation (3.3).

$$c(i, k) = c(i, j) - [c(x(j), j) - \min\{(c(x(t), t)\ \forall t \in \text{vertex set in cycle}\}] \quad (3.3)$$

Where, $c(x(j), j)$ is the cost of the edge in the cycle which enters j.

  i. For each new vertex, select the entering edge which has the smallest modified cost; replace the edge which enters the same real vertex in S by the new selected edge.
  ii. Go to step (ii) with the newly generated contracted graph.

The main idea of the algorithm [46] is to discover the replacing edge(s) which has the minimum extra cost to eliminate cycle(s), if any. The Equation (3.3) exhibits the associated extra cost.

Figure 3.3 illustrates that the contraction method finds the minimum additional cost replacing edge (2, 3) for edge (4, 3) and hence the cycle is eliminated.

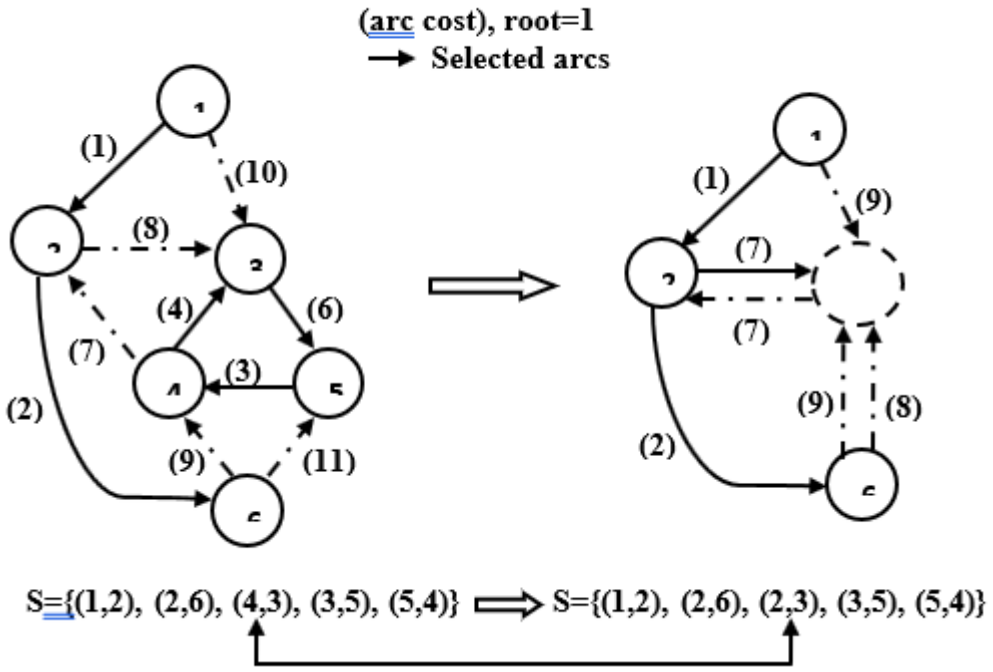


Figure 3.3: **MST construction using Chu-Liu/Edmond's algorithm**

b. Compute Attribute Similarity Set (ASS) using Relative Indiscernibility Relation:

The GRG method first computes the equivalence classes by $RIR_D(A_i)$, defined in Equation (3.1), for each conditional attribute A_i in the dataset. To illustrate the method, a sample dataset, shown in Table 3.1 is considered. Here, equivalence classes formed using $RIR_D(A_i)$ defined in Equation (3.1) are listed below:

$RIR_D(i) = \{\{x_1, x_7\}, \{x_2\}, \{x_3, x_8\}, \{x_4\}, \{x_5, x_6\}\}$

$RIR_D(e) = \{\{x_1\}, \{x_5\}, \{x_2, x_3, x_8\}, \{x_4, x_7\}, \{x_6\}\}$

$RIR_D(l) = \{\{x_1, x_4\}, \{x_2, x_3, x_5, x_6\}, \{x_7\}, \{x_8\}\}$

$RIR_D(r) = \{\{x_1\}, \{x_6, x_8\}, \{x_2, x_5\}, \{x_4\}, \{x_3\}, \{x_7\}\}$

Then, it calculates the degree of similarity ($\delta_f^{i,j}$) or similarity factor among each pair of conditional attributes (A_i, A_j) of the dataset using "SIM_FAC" algorithm discussed in section 3.2.1.2.

Now an attribute similarity set (ASS) is constructed using "ASS_GEN" algorithm discussed in section 3.2.1.3. Here ASS is represented as ASS = $\{A_i \xrightarrow{\delta_f^{i,j}} A_j\ (A_i \neq A_j)\}$ which contains the set of pairs of attributes that are most strongly related to each other, where $\delta_f^{i,j}$ is the similarity factor of attribute A_i to attribute A_j. By applying the "ASS_GEN" algorithm on the sample dataset, obtained initial ASS = $\{i \xrightarrow{\delta_f^{i,l}} l, i \xrightarrow{\delta_f^{i,r}} r, e \xrightarrow{\delta_f^{e,i}} i, e \xrightarrow{\delta_f^{e,l}} l, e \xrightarrow{\delta_f^{e,r}} r, r \xrightarrow{\delta_f^{r,l}} l\}$ and Table 3.7 gives the modified attribute similarity set , discussed in the following section.

c. Modified Attribute Similarity Set (ASS):

From Table 3.7, as the similarity factor for attribute similarities $i \xrightarrow{\delta_f^{i,l}} l$, $e \xrightarrow{\delta_f^{e,i}} i$, $e \xrightarrow{\delta_f^{e,l}} l$ and $r \xrightarrow{\delta_f^{r,l}} l$ are greater than average $\delta = 0.786$. So, the modified attribute similarity set is ASS = $\{i \xrightarrow{\delta_f^{i,l}} l, e \xrightarrow{\delta_f^{e,i}} i, e \xrightarrow{\delta_f^{e,l}} l, r \xrightarrow{\delta_f^{r,l}} l\}$ for the sample dataset.

Table 3.7: Selection of attribute similarities in ASS

Attribute Similarity ($A_i \xrightarrow{\delta_f^{i,j}} A_j$; $i \neq j$ and $\delta_f^{i,j} > \delta_f^{j,i}$)	**Similarity factor of A_i to A_j ($\delta_f^{i,j}$)**	$\delta_f^{i,j} > \delta$
$i \xrightarrow{\delta_f^{i,l}} l$	$\delta_f^{i,l} = 0.80$	Yes
$i \xrightarrow{\delta_f^{i,r}} r$	$\delta_f^{i,r} = 0.70$	
$e \xrightarrow{\delta_f^{e,i}} i$	$\delta_f^{e,i} = 0.83$	Yes
$e \xrightarrow{\delta_f^{e,l}} l$	$\delta_f^{e,l} = 0.83$	Yes
$e \xrightarrow{\delta_f^{e,r}} r$	$\delta_f^{e,r} = 0.76$	
$r \xrightarrow{\delta_f^{r,l}} l$	$\delta_f^{r,l} = 0.80$	Yes
Average Similarity (δ)	0.786	

e. Construction of Attribute Similarity Graph:

The modified and minimized attribute similarity set ASS = $\{A_i \xrightarrow{\delta_f^{i,j}} A_j\ \forall\ i\ \&\ j\}$ contains the set of pairs of attributes that are most strongly related to each other.

To generate a reduct, firstly this set is represented by a directed graph, named *attribute similarity graph (*ASG*)*.

In ASG, the vertices are the attributes present in the set ASS and weighted edge exists from attribute A_i to attribute A_j with weight $\delta_f^{i,j}$ if $A_i \xrightarrow{\delta_f^{i,j}} A_j \in$ ASS. The weight of an edge between two end vertices is the value of the similarity factor between two attributes of the dataset associated to these two vertices. Thus, attribute similarity $A_i \rightarrow A_j$ with $\delta_f^{i,j} = w$, present in set ASS is represented by a directed edge from vertex A_i to vertex A_j with weight w.

Mathematically, ASG is denoted as *G (V, E)*, where *V* and *E* defined by Equation (3.4) and Equation (3.5).

$$V = \{A_i | A_i \in (\text{Left}(x) \cup \text{Right}(x)) \forall x \in \text{ASS}\} \quad (3.4)$$

$$E = \left\{(A_i, A_j) \middle| A_i \xrightarrow{\delta_f^{i,j}} A_j \in \text{ASS}\right\} \quad (3.5)$$

The attribute similarity graph generated for Table 3.7 is shown in Figure 3.4.

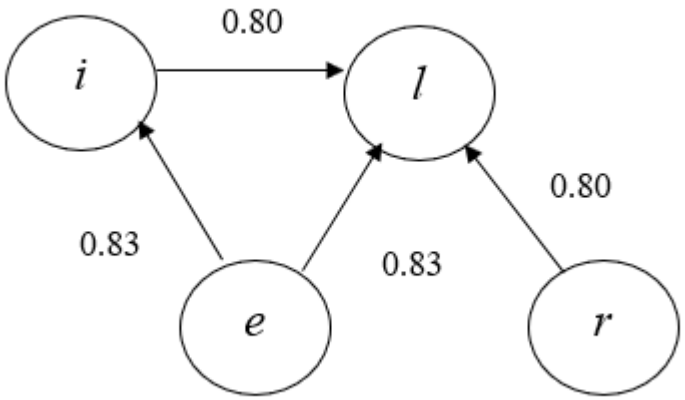


Figure 3.4: **ASG obtained from Table 3.7**

f. Construction of Directed Minimal Spanning Tree using Chu-Liu/Edmond's algorithm (CLE):

Attribute Similarity Graph (ASG) represents the overall similarity structure of the attribute similarity set ASS. Some vertices in the ASG may have multiple incoming edges which imply that a particular vertex (attribute) *v* is like many vertices (attributes). Now, the vertices of the graph, which have one or more out-going edges, represent the attributes to which some other attributes are similar. The weights of the edges between them denote the strength of their similarity. Therefore, a maximal spanning tree of this graph would give the highest similarities between two attributes. Constructing maximal spanning tree is equivalent to constructing minimal spanning tree inverting the weights of the edges.

So, to construct the minimal spanning tree, weights associated to each edge of the directed graph ASG are inversed and Chu-Liu / Edmond's Algorithm [46, 223] is applied. In the process, the vertex that has only outgoing edges and no incoming edges is considered as the root.

If more than one such vertex exists, then they are fused to form a single vertex. So, before construction of the minimal spanning tree, ASG is modified to merge all the nodes with in-degree zero to a single node and it is considered as the root of the graph. Generation of minimal spanning tree of ASG is given in "MST_GEN" algorithm.

Algorithm: MST_GEN (ASS) /* generate a minimal spanning tree of ASG */**Input**: ASS = modified attribute similarity set obtained from ASS_GEN algorithm

Output: Rooted Directed Minimal Spanning Tree M

Begin
/* Represent ASS as a graph using Equation (3.4) and Equation (3.5) */

Construct weighted graph ASG = (*V*, *E*) from ASS, where

$V = \{A_i \mid A_i \in \text{Left}(x) \cup \text{Right}(x), \forall x \in \text{ASS}\}$ $E = \{(A_i, A_j) \mid A_i \xrightarrow{\delta_f^{i,j}} A_j \in \text{ASS}\}$

/*Merge nodes with in-deg zero to create a new node*/

Root = { } for each vertex $N_i \in V$ do

if in_deg (N_i) = 0 then

Root = *Root*$\cup\{N_i\}$

Modify ASG by fusing all vertices in *Root*

end-if

end-for

for each edge $A_i \xrightarrow{\delta_f^{i,j}} A_j \in$ E do

$\delta_f^{i,j} = (\delta_f^{i,j})^{-1}$

end-for

Compute MST of ASG using CLE Algorithm

End

The algorithm modifies the attribute similarity graph shown in Figure 3.4 to a new graph, as shown in Figure 3.5 and constructs directed minimal spanning tree shown in Figure 3.6.

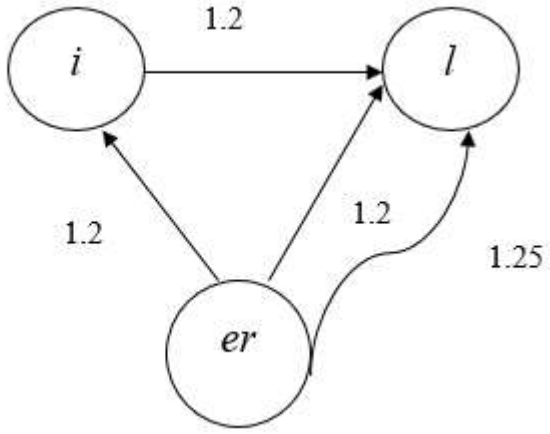


Figure 3.5: The modified ASG obtained from Figure 3.4

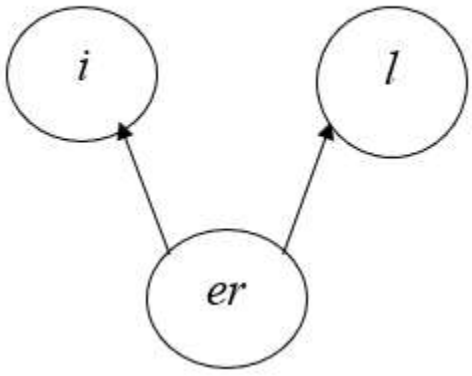


Figure 3.6: Minimal spanning tree of the graph of Figure 3.5

g. Generation of Reduct:

The above generated rooted directed minimal spanning tree would give the highest similarities between the attributes. In the final stage, the maximal spanning tree is searched to find the vertex with highest out-degree.

The vertex with highest out-degree is an attribute to which the greatest number of other attributes is similar. So, this node is added to the initially empty reduct set and its out-going edges are removed from the tree. This process of trimming the edges of the tree and adding the vertex (attribute) to the reduct set continues till the edge set of the tree becomes empty and thus final reduct is obtained.

Generation of reduct from rooted directed minimal spanning tree of ASG is given in "RED_GEN" algorithm.

Algorithm: RED_GEN (MST)

/* generates reduct from rooted directed minimal spanning tree of ASG */

Input: MST (V, S) = Rooted Directed Minimal Spanning Tree

Output: Reduct R

Begin

$R = \{\}$

order [V] = array of vertices of MST sorted in descending order of their out-degree

for $i = 1$ to $|V|$ do

Remove outgoing edges from vertex order[i]

$R = R \cup \{order[i]\}$

if ($S = \Phi$) then

return (R)

end-for

End

Reduct generated from Figure 3.6 is $\{e, r\}$ as shown in Figure 3.7.

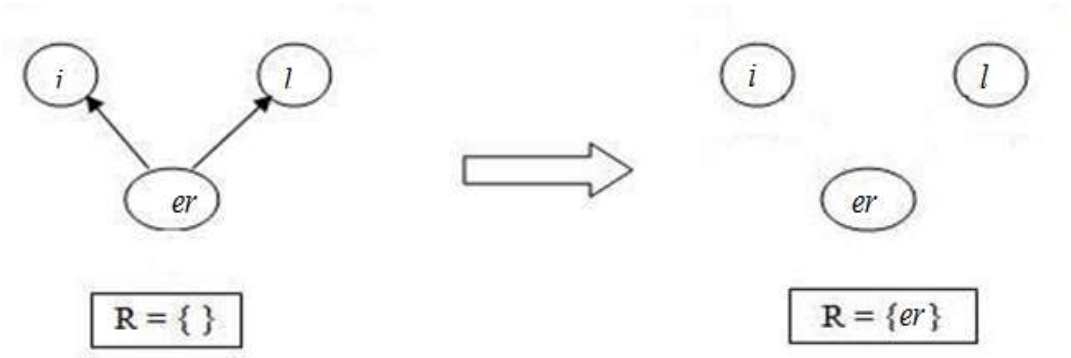


Figure 3.7: Reduct generation from minimal spanning tree

h. Experimental Results of GRG Method:

Experimental results presented here provide an evidence of effectiveness of GRG method to identify the most significant feature subset as a single reduct for experimental dataset [27, 28] summarized in the section 2.2. At first, all the attributes are discretized by ChiMerge [219] discretization algorithm. Then GRG method [44] and existing standard feature selection methods such as, CFS [95], CON [213], CAR [214], Relief-F [215], SVD [216] and MOGA [217] are applied on the dataset for selecting the important features and the reduced datasets are classified based on the considered base classifiers. 10-fold cross validation is used for the classification performance evaluation. Number of attributes after applying mentioned feature selection methods and the accuracies (%) of the datasets by base classifiers are computed and listed in Table 3.8, which shows the efficiency of the proposed GRG method. Similar to the SRG method discussed in section 3.2.1.6, the statistical analysis is done using Wilcoxon's rank sum test [39] and results are listed in Table 3.8. To indicate the best performing algorithm a bold-faced font is used.

Table 3.8: **Performance analysis of GRG and existing feature selection methods**

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†	97.19†
	CON (8)	96.19†	97.11≈	96.63†	94.94†	94.94†	94.30†	97.19†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.19†
	Relief-F (9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†	97.19†
	SVD (9)	96.65†	96.56†	96.76†	95.78†	95.45†	95.98†	96.65†
	MOGA (7)	97.17≈	96.65†	95.56†	96.64†	95.78†	95.87†	97.87†
	GRG (9)	**97.70**	**97.91**	**97.48**	**97.09**	**96.65**	**96.49**	**98.14**
Heart	CFS (8)	84.36†	84.75†	81.67†	81.11†	81.11†	81.67†	84.36≈
	CON (11)	84.50†	84.44†	82.07†	81.48†	82.89†	79.55†	84.50≈
	CAR (10)	83.36†	84.75†	81.67†	83.11†	82.11†	80.67†	83.36†
	Relief-(10)	83.50†	84.44†	82.07†	81.48†	83.89†	79.59†	83.50†
	SVD (9)	83.45†	83.67†	83.98†	82.21†	83.21†	82.08†	83.45†
	MOGA (8)	84.67†	83.32†	83.22†	83.56†	83.87†	83.26≈	84.67≈
	GRG (9)	**85.27**	**85.42**	**84.81**	**84.52**	**84.89**	**83.43**	**84.97**
Glass	CFS (6)	43.92†	57.94†	79.91†	73.83†	68.69†	70.09†	43.92†
	CON (7)	47.20†	57.48†	78.50†	71.50†	64.20†	68.60†	47.20†
	CAR (8)	56.92†	58.94†	80.91†	75.83†	69.69†	71.09†	56.92†
	Relief-F (8)	57.20†	57.48†	79.50†	70.50†	63.20†	72.60†	57.20†
	SVD (8)	56.67†	57.75†	77.39†	71.56†	67.50†	74.45†	56.67†
	MOGA (7)	56.54†	57.76†	76.49†	72.45†	64.76†	70.89†	56.54†
	GRG (8)	**67.28**	**64.48**	**83.64**	**76.63**	**70.09**	**75.23**	**77.92**
	CFS (9)	96.03†	93.06†	94.05†	94.04†	93.06†	93.06†	93.54†
	CON (9)	96.03†	93.03†	94.05†	94.04†	93.88†	94.32†	94.45†

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Zoo	CAR (6)	94.05†	93.92†	93.32†	94.02†	94.07†	94.05†	95.45†
	Relief-F (7)	95.03†	93.70†	93.01†	93.01†	94.12†	94.12†	95.03†
	SVD (8)	96.09†	94.67†	94.78†	94.67†	94.32†	94.67†	95.56†
	MOGA (6)	94.78†	94.23†	94.45†	95.21≈	94.32†	94.34†	94.54†
	GRG (8)	**97.01**	**95.04**	**95.05**	**95.48**	**95.89**	**95.12**	**97.04**
Dermatology	CFS (9)	**98.76**	97.42†	97.01†	98.06≈	98.07†	98.62≈	98.76≈
	CON (9)	98.52≈	98.25≈	95.56†	98.06≈	**98.86**	98.67≈	98.52≈
	CAR (11)	98.73≈	98.30≈	97.42†	98.31≈	98.06†	98.07≈	98.73≈
	Relief-(11)	98.72≈	98.45≈	95.56†	97.16†	98.76≈	98.46≈	98.72≈
	SVD (9)	97.78†	98.03≈	96.75†	97.09†	98.01†	97.89†	97.78†
	MOGA (9)	97.87†	97.89†	97.90†	97.05†	98.67≈	97.90†	97.87†
	GRG (9)	98.23≈	**98.57**	**98.45**	**98.51**	98.41≈	**98.93**	**98.97**
Mushroom	CFS (4)	97.52†	96.01†	96.52†	97.01†	97.01†	97.01†	97.52†
	CON (5)	98.52†	98.85†	98.52†	99.05≈	98.16†	**99.86**	98.52≈
	CAR (8)	98.02†	98.32†	99.02≈	**99.65**	**99.23**	99.01†	98.02†
	Relief-F (5)	97.04†	98.03†	98.03†	98.13†	98.10†	98.10†	97.04†
	SVD (5)	97.04†	97.23†	97.23†	97.83†	97.34†	87.64†	97.04†
	MOGA (5)	97.34†	95.45†	96.67†	96.34†	96.34†	96.45†	97.34†
	GRG (5)	**99.04**	**99.02**	**99.34**	98.78†	96.89†	99.08†	**98.65**
Coil20	CFS (194)	78.12†	79.24†	80.01†	80.10†	79.25†	79.85†	78.12†
	CON (194)	79.60†	79.20†	78.00†	79.35†	79.34†	78.79†	79.60†
	CAR (201)	77.12†	78.24†	77.01†	80.10†	75.25†	78.85†	77.12†
	Relief (198)	77.60†	78.20†	77.80†	78.35†	77.34†	78.49†	77.60†
	SVD (132)	76.60†	76.20†	77.80†	75.35†	76.64†	78.49†	76.60†
	MOGA (95)	82.20†	79.99†	**82.92**	**83.02**	83.02†	84.20≈	82.20†
	GRG (132)	**83.12**	**80.01**	81.87†	82.32†	**84.32**	**84.43**	**85.43**
Orl	CFS (201)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	55.12†
	CON (198)	56.60†	53.20†	52.00†	51.35†	50.70†	52.89†	56.60†
	CAR (204)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	55.12†
	Relief-F (213)	54.60†	54.20†	54.01†	54.25†	53.79†	53.89†	54.60†
	SVD (142)	52.12†	53.24†	53.01†	52.10†	53.80†	54.35†	52.12†
	MOGA (110)	59.60†	57.20†	59.00†	58.25†	59.70†	59.09†	59.60†
	GRG (142)	**60.01**	**59.03**	**60.07**	**61.02**	**61.02**	**60.21**	**61.98**
Allaml	CFS (221)	81.12†	81.32†	82.62†	83.82†	82.21†	83.01†	81.12†
	CON (230)	79.12†	80.32†	81.62†	82.82†	80.21†	82.01†	79.12†
	CAR (201)	80.02†	80.32†	81.52†	82.72†	81.21†	82.02†	80.02†

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
	Relief-F (210)	81.12†	81.32†	82.62†	83.82†	82.21†	83.11†	81.12†
	SVD (212)	80.22†	81.62†	81.27†	82.92†	82.81†	83.19†	80.22†
	MOGA (194)	81.12†	83.32≈	**84.72**	81.82†	82.01†	82.21†	81.12†
	GRG (212)	**83.43**	**83.67**	84.32≈	**84.21**	**83.70**	**84.76**	**84.50**
Leukemia	CFS (147)	83.23†	84.23†	82.67†	82.02†	82.05†	83.23†	83.23†
	CON (159)	84.05†	85.87†	84.23†	84.12†	84.21†	84.67†	84.05†
	CAR (147)	83.47†	85.23†	84.56†	84.53†	84.54†	84.56†	83.47†
	Relief-F (159)	84.23†	85.34†	84.12†	84.34†	84.34†	84.98†	84.23†
	SVD (124)	73.68†	76.32†	71.05†	71.02†	71.05†	73.68†	73.68†
	MOGA (97)	86.34†	85.90†	85.50†	85.78†	87.12†	86.23†	86.34†
	GRG (124)	**87.67**	**86.98**	**86.67**	**86.98**	**88.20**	**87.98**	**88.90**

To show the effectiveness of the classifiers based on the reduced features achieved from different existing feature selection method, some other statistical measurements given in Equation (2.25) to (2.28) are performed and the average results for all seven classifiers are listed in Table 3.9.

Table 3.9: Statistical measures for GRG and different competitive algorithm

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Wine	CFS (8)	0.96	0.04	0.95	0.96
	CON (8)	0.96	0.04	0.96	0.96
	CAR (8)	0.95	0.06	0.94	0.95
	Relief-F (9)	0.96	0.04	0.96	0.95
	SVD (9)	**0.97**	0.03	0.96	**0.97**
	MOGA (7)	**0.97**	0.03	0.97	**0.97**
	GRG (9)	**0.97**	**0.02**	**0.98**	**0.97**
Heart	CFS (8)	**0.98**	**0.02**	**0.98**	**0.97**
	CON (11)	0.82	0.18	0.83	0.82
	CAR (10)	0.83	0.16	0.82	0.83
	Relief-F (10)	0.83	0.17	0.83	0.83
	SVD (9)	0.82	0.17	0.82	0.83
	MOGA (8)	0.83	0.18	0.82	0.82
	GRG (9)	0.85	0.14	0.84	0.85
	CFS (6)	0.66	0.36	0.67	0.66

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Glass	CON (7)	0.65	0.36	0.65	0.64
	CAR (8)	0.69	0.31	0.68	0.69
	Relief-F (8)	0.68	0.32	0.69	0.68
	SVD (8)	0.69	0.31	0.69	0.68
	MOGA (7)	0.68	0.32	0.68	0.68
	GRG (8)	**0.74**	**0.25**	**0.73**	**0.74**
Zoo	CFS (9)	0.94	0.06	0.93	0.94
	CON (9)	0.94	0.06	0.94	0.93
	CAR (6)	0.94	0.06	0.92	0.94
	Relief-F (7)	0.94	0.05	0.94	0.93
	SVD (8)	0.95	0.05	0.95	0.95
	MOGA (6)	0.95	0.04	0.95	0.94
	GRG (8)	**0.96**	**0.03**	**0.96**	**0.95**
Dermatology	CFS (9)	0.98	**0.01**	**0.99**	0.98
	CON (9)	0.98	**0.01**	0.98	0.97
	CAR (11)	0.98	0.02	0.97	0.98
	Relief-F (11)	0.98	0.02	0.98	0.98
	SVD (9)	0.98	0.02	0.98	0.97
	MOGA (9)	0.98	0.02	0.97	0.98
	GRG (9)	**0.99**	**0.01**	**0.99**	**0.99**
Mushroom	CFS (4)	0.97	0.03	0.96	0.97
	CON (5)	**0.99**	**0.01**	0.98	**0.99**
	CAR (8)	**0.99**	**0.01**	**0.99**	0.98
	Relief-F (5)	0.98	0.02	0.97	0.98
	SVD (5)	0.96	0.04	0.96	0.95
	MOGA (5)	0.97	**0.01**	0.96	0.97
	GRG (5)	**0.99**	**0.01**	**0.99**	0.98
Coil20	CFS (194)	0.80	0.19	0.81	0.80
	CON (194)	0.79	0.21	0.79	0.78
	CAR (201)	0.78	0.21	0.78	0.79
	Relief-F (198)	0.78	0.22	0.77	0.78
	SVD (132)	0.77	0.23	0.77	0.78
	MOGA (95)	**0.83**	**0.16**	**0.83**	**0.83**
	GRG (132)	0.83	0.17	0.83	0.83
	CFS (201)	0.54	0.45	0.53	0.54
	CON (198)	0.53	0.45	0.53	0.52

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Orl	CAR (204)	0.53	0.46	0.52	0.53
	Relief-F (213)	0.54	0.46	0.54	0.53
	SVD (142)	0.54	0.46	0.54	0.54
	MOGA (110)	0.59	0.37	0.60	0.59
	GRG (142)	**0.60**	**0.39**	**0.60**	**0.60**
Allaml	CFS (221)	0.82	0.17	0.82	0.82
	CON (230)	0.83	0.18	0.83	0.82
	CAR (201)	0.81	0.19	0.80	0.81
	Relief-F (210)	0.81	0.19	0.81	0.82
	SVD (212)	0.82	0.18	0.81	0.82
	MOGA (194)	0.82	0.18	0.82	0.81
	GRG (212)	**0.84**	**0.15**	**0.83**	**0.84**
Leukemia	CFS (147)	0.83	0.16	0.81	0.83
	CON (159)	0.85	0.13	0.85	0.84
	CAR (147)	0.85	0.14	0.85	0.85
	Relief-F (159)	0.85	0.15	0.86	0.85
	SVD (124)	0.73	0.23	0.73	0.77
	MOGA (97)	0.88	0.12	0.87	0.88
	GRG (124)	**0.88**	**0.12**	**0.89**	**0.88**

From Table 3.8 and 3.9, it is proved that the proposed GRG method is better than most of the existing standard methods.

3.2.3 Comparative Analysis of SRG and GRG Methods:

Comparisons of SRG [43], and GRG [44] methods are made based on the results given by both methods on experimental datasets.

Here, both the methods generate single reduct. The method SRG selects single reduct based on RST [17-20]. On the other hand, GRG method selects reduct based on Minimal Spanning Tree and RST concepts.

In Table 3.10, the average classification accuracies are measured based on the reduced feature subset generated by SRG and GRG method by mentioned existing state of the art classifiers from Weka tool [218].

It is observed that the SRG method identifies lesser number of features in the reduct than GRG method but at the same time GRG method gives better classification accuracies for some experimental datasets such as Heart, Zoo, Mushroom and Allaml. Computational time required for execution of SRG is little less than GRG.

Table 3.10: Comparison of SRG and GRG methods

Dataset	#Selected features		Average Accuracy (%)		Computational time (sec)	
	SRG	**GRG**	**SRG**	**GRG**	**SRG**	**GRG**
Wine	**6**	9	**97.89**	97.29	**3.10**	5.23
Heart	**4**	9	84.05	**84.70**	**2.01**	3.12
Glass	**6**	8	**73.62**	73.61	**1.99**	2.98
Zoo	**8**	**8**	95.59	**95.80**	**0.98**	1.74
Dermatology	**9**	**9**	**99.13**	98.57	**22.43**	31.23
Mushroom	**4**	5	98.53	**98.68**	**167.23**	248.45
Coil20	**123**	132	**89.56**	83.07	**635.21**	987.34
Orl	**132**	142	**62.13**	60.47	**435.23**	735.23
Allaml	**201**	212	82.81	**84.08**	**1021.23**	1670.43
Leukemia	**102**	124	**88.97**	87.62	**1010.84**	1800.02

3.3 Multiple Feature Subset Selection:

In many applications, feature selection technique focuses on multiple feature subset selection as they calculate the group performance of features and helps to identify the best set of features in a data set. In this section, two multiple feature subset selection methods [47, 48] are proposed by which multiple reducts are generated from the datasets. First method [47] generates multiple reducts based on the concepts of RST only while the second method [48] generates multiple reducts based on the concepts of RST, clustering algorithm and graph theory.

3.3.1 Multiple Reducts Generation Using Forward Selection and Backward Removal Techniques (FSBR):

In this section, the multiple feature subset selection method FSBR [47] is described for selecting a set of compact feature subset from a dataset without losing any information. This novel and heuristic method tries to find out only a compact reduct set based on the concepts like discernibility relation and attribute dependency of the rough set theory [17-20].

The method tries to tradeoff between the two approaches popularly used by different researchers by forming only a few reducts without spending much time to compute all possible reducts. Before describing the method, basics of rough set theory is presented briefly below.

a. Related Concepts of Rough Set Theory:

The rough set theory is based on indiscernibility relations and approximations. Indiscernibility relation is usually assumed to be equivalence relation, interpreted so that two objects are equivalent if they are not distinguishable by their properties.

Given a decision system $DS = (U, A)$, where U is the universe of discourse and A is the total number of attributes consisting of two types of attributes namely conditional attributes (C) and decision attributes (D) so that $A = C \cup D$.

Let the universe $U = \{x_1, x_2, .., x_n\}$, then with any $P \subseteq A$, there is an associated P-indiscernibility relation $IND(P)$ defined by Equation (2.7) in the section 2.3.2.6.

The lower approximation of a target set X with respect to P is the set of all objects which certainly belongs to X, as defined by the Equation (2.8) in the section 2.3.2.6.

The upper approximation of the target set X with respect to P is the set of all objects which can possibly belong to X, as defined by the Equation (2.10) in the section 2.3.2.6.

The rough set is defined by the tuple $<\underline{P}X, \overline{P}X>$.

Based on the concepts of discernibility, a discernibility matrix is constructed to represent the family of discernibility relations.

Each cell in a discernibility matrix consists of all the attributes on which the two objects have the different values. Two objects are discernible with respect to a set of attributes if the set is a subset of the corresponding cell of the discernibility matrix.

Discernibility matrix $M = (m_i)$ is a $|U| \times |U|$ matrix, in which the element m_i for an object pair (x_i, x_j) is defined by Equation (3.6).

$m_{ij} = \{a \in C: a(x_i) \neq a(x_j) \wedge (d \in D, d(x_i) \neq d(x_j))\}$, where, $i, j = 1, 2, 3, .., n$ (3.6)

Example: To illustrate the concept of discernibility matrix, a decision system is considered in Table 3.11. According to Equation (3.6) discernibility matrix of the Table 3.11 is given in Table 3.12.

Table 3.11: **Sample Decision System**

Attributesm /Objects	***a*'**	***b*'**	***c*'**	***d*'**	***D***
O_1	0	1	3	0	1
O_2	0	1	1	2	1
O_3	1	1	1	0	1
O_4	1	4	3	0	2
O_5	0	4	2	0	2
O_6	1	4	3	1	3
O_7	2	4	3	1	3
O_8	2	5	3	1	3

Table 3.12: Discernibility Matrix of the Decision System Given in Table 3.11

			a'b'	*b'c'*	*a'b'd'*	*a'b'd'*	*a'b'd'*
			b'c'd'	*b'c'd'*	*a'b'c'd'*	*a'b'c'd'*	*a'b'c'd'*
			b'c'	*a'b'c'*	*b'c'd'*	*a'b'c'd'*	*a'b'c'd'*
					d'	*a'd'*	*a'b'd'*
					a'c'd'	*a'c'd'*	*a'b'c'd'*

- **Attribute Dependency:**

Attribute dependency of two disjoint attribute set *Q* on attribute set *P* is denoted by $\gamma_P(Q)$ in RST and is given in Equation (2.12). Where *P* and *Q* are disjoint to each other. That is, for each equivalence class Q_i in $[x]_Q$, the size of its lower approximation is added up by the attributes in *P*, i.e., $\underline{P}Q_i$.

This approximation is the number of objects which on attribute set *P* can be positively identified as belonging to target set Q_i. Added across all equivalence classes in $[x]_Q$, the numerator above represents the total number of objects which, based on attribute set *P*, can be positively categorized according to the classification induced by attributes *Q*. The dependency ratio therefore expresses the proportion of such classifiable objects.

- **Core and Noncore Attribute:**

Each entry (*i*, *j*) of matrix *M* contains the attributes by which the objects *i* and *j* are distinguishable. An entry contains minimum number of attributes implies that the attribute(s) is (are) sufficient to distinguish two associated objects and so it is considered as the most important attribute. In matrix *M*, the entries with the minimum number of attributes form *core* attribute set, say *CR*, defined by Equation (3.7) and remaining are treated as *noncore* attributes, say *NC*.

$$CR = \cup \{m_{ij} | m_{ij} \neq \emptyset \text{ and } |m_{ij}| = 1, \forall\, i, j = 1, 2, \ldots, n\} \qquad (3.7)$$

Example: The discernibility matrix in Table 3.12 show that the single entry (4, 6) contains *d'*, which is the minimum and so *CR* = {*d'*} is formed.

Hence the rest of the conditional attributes form a set of *noncore* set *NC* = {*a'*, *b'*, *c'*}.

Noncore attributes are further ranked based on their frequency in matrix *M* using equation (3.8).

Higher frequency indicates higher ranked attribute and vice versa.

$$RNK(a) = \left|\{x: x \in M \wedge a \in \{x\}\}\right| \quad (3.8)$$

From the discernibility matrix in Table 3.12, since the frequency of a'=15, b'= 17 and c'= 14, so according to the rank of decreasing order, the attribute set NC = $\{b', a', c'\}$.

- **Reduction of Attributes:**

A reduct can be thought of as a complete set of attributes to represent the category structure of the decision system. Projected on just these attributes, the decision system possesses the same equivalence class structure as that expressed by the full attribute set. A subset of attributes *R* is a reduct if the dependency of decision attribute *D* on *R* is exactly equal to that of *D* on whole conditional attribute set *C* and defined in Equation (3.9).

$$\gamma_R(D) = \gamma_C(D) \quad (3.9)$$

The reduct of an information system is not unique.

There may be many subsets of attributes which preserve the equivalence-class structure (i.e., the knowledge) expressed in the decision system.

The detail procedure of reduct generation by FSBR is discussed below.

b. Compact Reduct Set Formation:

Based on the discernibility matrix *M*, the attributes are divided into the core set *CR* and noncore set *NC*.

The FSBR method uses (i) a forward attribute selection method and (ii) a backward attribute removal method for the computation of final reduct set *RED*.

h. **Forward Attribute Selection:** Rank of all noncore attributes in *NC* is calculated based on their frequency in the discernibility matrix *M* using Equation (3.8). Obviously, attribute having higher frequency has higher rank and is more important than the other attributes. Next, highest ranked element of *NC* is added to the core *CR* in each iteration; provided the dependency of the decision attribute *D* on the resultant set increases; otherwise, it is ignored and next iteration with the remaining elements in *NC* is performed. The process terminates when the resultant set satisfies the Equation (3.9) or satisfy the criteria ($|\gamma_R(D) - \gamma_C(D)| < \delta$) where the value of δ is very small, set experimentally. After getting one reduct, the same process is repeated with core *CR* and remaining noncore attributes in *NC* and finally, multiple reducts are obtained.

ii. Backward Attribute Removal: The demerit of forward attribute selection is that it always selects the higher ranked attribute before the lower one. In some cases, one higher ranked attribute (say, in i-th iteration) together with another comparatively lower ranked attribute (say, in (i+2)-th iteration) may have higher attribute dependency compared to that in the case which arises in forward selection method by three consecutive, namely, i-th, (i+1)-th and (i+2)-th iterations. In such situations, the noncore attribute added in (i+1)-th iteration may be removed from the generated reduct. So, for each noncore attribute x in generated reduct R, it is checked whether Equation (3.9) or the criteria ($|\gamma_R(D) - \gamma_C(D)| < \delta$) is satisfied using $R - \{x\}$, instead of R. If it is satisfied, then x is redundant and must be removed. Thus, all redundant attributes are obtained and stored in set RM. Now, if the Equation (3.9) or ($|\gamma_R(D) - \gamma_C(D)| < \delta$) is satisfied using $R - RM$ instead of R, then $R - RM$ is a final reduct; otherwise, repeatedly compute all subsets of RM taking $|RM| - 1$ elements together and check the Equation (3.9) or ($|\gamma_R(D) - \gamma_C(D)| < \delta$) for all those subsets. For any subset S satisfying the Equation (3.9) or ($|\gamma_R(D) - \gamma_C(D)| < \delta$), removing S from R gives a reduct and further processing with the subsets of S is not required. This acts as the terminating condition for the process. Thus, for a single reduct obtained by the forward selection method, a set of reducts may be formed. Repeating the process for all reducts obtained from forward selection method gives a compact set of reducts.

Example: For the decision system DS in Table 3.11, the forward selection method gives two reducts $RED = \{\{ b', d' \}, \{ a', c', d' \}\}$. Here, both the reducts contains no redundant attributes, so backward removal method can't eliminate any attribute from the reducts in RED. So, $RED = \{\{b', d'\}, \{ a', c', d' \}\}$ is the final reducts of the sample decision system.

The algorithm of a compact set of reducts formation for a decision system $DS = (U, A)$ is described below, where forward selection method is described in the 'Reduct_Formation' algorithm and for each generated reduct, backward removal algorithm 'Back_Removal' is invoked to obtain the compact set of reducts.

Algorithm: Reduct_Formation (*DS, CR, NC*)
Input: *DS, CR, NC* /*Decision system with C conditional attributes and D decision attributes, the core and the non-core attributes */
Output: *RED* /*set of compact reducts */
Begin
  Repeat
    $R = CR$ /* core is considered as initial reduct*/
    $NC\_OLD = NC$ /* a copy of initial elements of NC*/
    Repeat
      x = the next highest ranked element of NC
      if $\gamma_{R\cup\{x\}}(D) > \gamma_R(D)$ then
          $R = R \cup \{x\}$
          $NC = NC - \{x\}$
      end-if
    Until (($\gamma_R(D) = \gamma_C(D)$) OR ($|\gamma_R(D) - \gamma_C(D)| < \delta$)).
    Call Back_Removal (*DS, RED, R, CR, NC*)
      $RED = RED \cup R$
  Until (*NC* is empty)
End
Algorithm: Back_Removal (*DS, RED, R, CR,NC*)

/**DS* = Decision system with *C* conditional attributes and *D* decision attributes, *CR* = the core, *NC* = the non-core attributes, *RED* = set of reducts, *R* = a single reduct */
Begin
  RD = *RM* = ϕ //*RM* contains all redundant attributes
  for each *x* in (*R* – *CR*) do
    if (($\gamma_{R-\{x\}}(D) = \gamma_C(D)$) OR ($|\gamma_R(D) - \gamma_C(D)| < \delta$)) then
      RM = *RM* ∪ {*x*}
  Insert *RM* into Queue *Q*
  While (*Q* is not empty)
    RM = Remove (*Q*)
    if (($\gamma_{R-RM}(D) = \gamma_C(D)$) OR ($|\gamma_{R-RM}(D) - \gamma_C(D)| < \delta$)) then
      if (*R* – *RM* ⊃ any reduct in *RED*) then
        Continue.
      else
        RED = *R* – *RM*
        NC = *NC* ∪ *RM*
      end-if
    else
      Compute all subsets of length |*RM*| - 1 of *RM*
      Insert all subsets into *Q*
    end-if
  end-while

End

c. Experimental Results of Method:

Extensive experiments are done to evaluate the FSBR method using experimental benchmark datasets [27, 28] described in the section 2.2.

Experimental results presented here provide an evidence of effectiveness of FSBR method to identify the set of significant feature subset as a multiple reduct. At first, all the attributes are discretized by ChiMerge [219] discretization algorithm.

Then proposed FSBR [47] and other well-known feature selection methods such as, CFS [95], CON [213], CAR [214], Relief-F [215], SVD [216] and MOGA [217] are applied on the dataset for selecting the important features by different method and the reduced datasets are classified on the considered base classifiers.10-fold cross validation is used for the classification performance evaluation.

The results of the best reduct of the multiple reducts generated by FSBR method are shown in Table 3.13.

Similar to the SRG method discussed in section 3.2.1.6, the statistical analysis is done using Wilcoxon's rank sum test [39] and results are listed in Table 3.13. To indicate the best performing algorithm bold faced font is used.

Table 3.13. Performance analysis of FSBR and existing methods

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†	95.70†
	CON (8)	96.19†	97.11†	96.63†	94.94†	94.94†	94.30†	97.65†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.56†
	Relief-(9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†	97.23†
	SVD (7)	96.65†	96.56†	96.76†	95.78†	95.45†	95.98†	97.80†
	MOGA (7)	97.17†	96.65†	95.56†	96.64	95.78	95.87†	97.67†
	MRG (7)	**98.96**	**98.64**	**97.98**	**97.87**	**96.90**	**96.83**	**98.67**
Heart	CFS (8)	84.36†	84.75†	81.67†	81.11†	81.11†	81.67†	82.78†
	CON (11)	84.50†	84.44†	82.07†	81.48†	82.89†	79.55†	82.72†
	CAR (10)	83.36†	84.75†	81.67†	83.11†	82.11†	80.67†	82.34†
	Relief-10)	83.50†	84.44†	82.07†	81.48†	83.89†	79.59†	82.30†
	SVD (7)	83.45†	83.67†	83.98†	82.21†	83.21†	82.08†	83.87†
	MOGA (8)	84.67†	83.32†	83.22†	83.56†	83.87†	83.26†	84.98†
	MRG (7)	**85.22**	**85.31**	**84.37**	**84.98**	**84.93**	**84.97**	**85.83**
Glass	CFS (6)	43.92†	57.94†	79.91†	73.83†	68.69†	70.09†	66.02†
	CON (7)	47.20†	57.48†	78.50†	71.50†	64.20†	68.60†	64.65†
	CAR (8)	56.92†	58.94†	**80.91**	75.83†	69.69†	71.09†	68.54†
	Relief-(8)	57.20†	57.48†	79.50†	70.50†	63.20†	72.60†	67.74†
	SVD (4)	56.67†	57.75†	77.39†	71.56†	67.50†	74.45†	75.89†
	MOGA (7)	56.54†	57.76†	76.49†	72.45†	64.76†	70.89†	76.23†
	MRG (4)	**70.42**	**72.36**	72.98†	**78.13**	**75.19**	**77.42**	**80.56**
Zoo	CFS (9)	96.03†	93.06†	94.05†	94.04†	93.06†	93.06†	93.54†
	CON (9)	96.03†	93.03†	94.05†	94.04†	93.88†	94.32†	94.45†
	CAR (6)	94.05†	93.92†	93.32†	94.02†	94.07†	94.05†	95.45†
	Relief-(7)	95.03†	93.70†	93.01†	93.01†	94.12†	94.12†	95.03†
	SVD (6)	96.09†	94.67†	94.78†	94.67†	94.32†	94.67†	95.56†
	MOGA (6)	94.78†	94.23†	94.45†	95.21†	94.32†	94.34†	94.54†
	MRG (6)	**98.02**	**97.19**	**97.24**	**97.04**	**98.27**	**96.21**	**97.04**
Dermatology	CFS (9)	98.76†	97.42†	97.01†	98.06†	98.07†	98.62†	**99.09**
	CON (9)	98.52†	98.25†	95.56†	98.06†	98.86†	98.67†	98.45†
	CAR (11)	98.73†	98.30†	97.42†	98.31†	98.06†	98.07†	98.54†
	Relief-11)	98.72†	98.45†	95.56†	97.16†	98.76†	98.46†	98.45†
	SVD (8)	97.78†	98.03†	96.75†	97.09†	98.01†	97.89†	97.65†
	MOGA (9)	97.87†	97.89†	97.90†	97.05†	98.67†	97.90†	97.50†
	MRG (8)	**99.33**	**99.27**	**98.06**	**99.17**	**99.16**	**99.27**	99.06≈
	CFS (4)	97.52†	96.01†	96.52†	97.01†	97.01†	97.01†	97.04†

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†	95.70†
	CON (8)	96.19†	97.11†	96.63†	94.94†	94.94†	94.30†	97.65†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.56†
	Relief-(9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†	97.23†
	SVD (7)	96.65†	96.56†	96.76†	95.78†	95.45†	95.98†	97.80†
	MOGA (7)	97.17†	96.65†	95.56†	96.64	95.78	95.87†	97.67†
	MRG (7)	**98.96**	**98.64**	**97.98**	**97.87**	**96.90**	**96.83**	**98.67**
Mushroom	CON (5)	**98.52**	**98.85**	98.52≈	99.05†	98.16†	**99.86**	98.54≈
	CAR (8)	98.02≈	98.32≈	**99.02**	**99.65**	**99.23**	99.01†	98.45≈
	Relief-(5)	97.04†	98.03†	98.03†	98.13†	98.10†	98.10†	98.23≈
	SVD (4)	97.04†	97.23†	97.23†	97.83†	97.34†	87.64†	97.45†
	MOGA (5)	97.34†	95.45†	96.67†	96.34†	96.34†	96.45†	97.64†
	MRG (4)	**98.52**	98.55≈	98.39†	97.44†	98.54†	98.72†	**98.68**
Coil20	CFS (194)	78.12†	79.24†	80.01†	80.10†	79.25†	79.85†	80.98†
	CON (194)	79.60†	79.20†	78.00†	79.35†	79.34†	78.79†	80.21†
	CAR (201)	77.12†	78.24†	77.01†	80.10†	75.25†	78.85†	78.98†
	Relief-198)	77.60†	78.20†	77.80†	78.35†	77.34†	78.49†	80.21†
	SVD (119)	76.60†	76.20†	77.80†	75.35†	76.64†	78.49†	79.21†
	MOGA (95)	82.20†	79.99†	82.92†	83.02≈	83.02†	84.20†	84.71†
	MRG (119)	**84.60**	**82.90**	**84.80**	**83.48**	**84.98**	**86.76**	**85.56**
Orl	CFS (201)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	57.65†
	CON (198)	56.60†	53.20†	52.00†	51.35†	50.70†	52.89†	54.87†
	CAR (204)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	57.65†
	Relief-(213)	54.60†	54.20†	54.01†	54.25†	53.79†	53.89†	54.87†
	SVD (137)	52.12†	53.24†	53.01†	52.10†	53.80†	54.35†	57.65†
	MOGA (110)	59.60†	57.20†	59.00†	58.25†	**59.70**	59.09	**59.87**
	MRG (137)	**61.20**	**63.40**	**60.30**	**60.21**	59.34	**59.34**	59.32
Allaml	CFS (221)	81.12†	81.32†	82.62†	83.82†	82.21†	83.01†	83.80†
	CON (230)	79.12†	80.32†	81.62†	82.82†	80.21†	82.01†	83.80†
	CAR (201)	80.02†	80.32†	81.52†	82.72†	81.21†	82.02†	82.80†
	Relief-(210)	81.12≈	81.32†	82.62†	83.82†	82.21†	83.11†	83.80†
	SVD (173)	80.22†	81.62†	81.27†	82.92†	82.81†	83.19†	82.80†
	MOGA (194)	81.12≈	83.32†	84.72†	81.82†	82.01†	82.21†	84.50†
	MRG (173)	**81.46**	**84.32**	**85.98**	**85.36**	**85.98**	**84.23**	**86.32**
	CFS (147)	83.23†	84.23†	82.67†	82.02†	82.05†	83.23†	84.34†
	CON (159)	84.05†	85.87†	84.23†	84.12†	84.21†	84.67†	85.01†

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†	95.70†
	CON (8)	96.19†	97.11†	96.63†	94.94†	94.94†	94.30†	97.65†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.56†
	Relief-(9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†	97.23†
	SVD (7)	96.65†	96.56†	96.76†	95.78†	95.45†	95.98†	97.80†
	MOGA (7)	97.17†	96.65†	95.56†	96.64	95.78	95.87†	97.67†
	MRG (7)	**98.96**	**98.64**	**97.98**	**97.87**	**96.90**	**96.83**	**98.67**
Leukemia	CAR (147)	83.47†	85.23†	84.56†	84.53†	84.54†	84.56†	89.10†
	Relief-F (159)	84.23†	85.34†	84.12†	84.34†	84.34†	84.98†	87.50†
	SVD (99)	73.68†	76.32†	71.05†	71.02†	71.05†	73.68†	75.87†
	MOGA (97)	86.34†	85.90†	85.50†	85.78≈	87.12≈	86.23†	**89.80**
	MRG (99)	**87.90**	**86.42**	**87.32**	**85.98**	**87.45**	**87.34**	89.34≈

To show the effectiveness of the classifiers based on the reduced features achieved from different existing feature selection method, some other statistical measurements given in Equation (2.25) to (2.28) in the section 2.6.2 are performed and the average results for all seven classifiers are listed in Table 3.14.

Table 3.14: Statistical measures for FSBR and different competitive algorithm

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Wine	CFS (8)	0.96	0.04	0.95	0.96
	CON (8)	0.96	0.04	0.96	0.96
	CAR (8)	0.95	0.06	0.94	0.95
	Relief-F (9)	0.96	0.04	0.96	0.95
	SVD (6)	**0.97**	0.03	0.96	0.97
	MOGA (7)	**0.97**	0.03	**0.97**	0.97
	FSBR (6)	**0.97**	**0.02**	**0.97**	**0.98**
Heart	CFS (8)	**0.98**	**0.02**	**0.98**	**0.97**
	CON (11)	0.82	0.18	0.83	0.82
	CAR (10)	0.83	0.16	0.82	0.83
	Relief-F (10)	0.83	0.17	0.83	0.83
	SVD (9)	0.82	0.17	0.82	0.83
	MOGA (8)	0.83	0.18	0.82	0.82
	FSBR (9)	0.85	0.15	0.84	0.85
	CFS (6)	0.66	0.36	0.67	0.66
	CON (7)	0.65	0.36	0.65	0.64

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Glass	CAR (8)	0.69	0.31	0.68	0.69
	Relief-F (8)	0.68	0.32	0.69	0.68
	SVD (6)	0.69	0.31	0.69	0.68
	MOGA (7)	0.68	0.32	0.68	0.68
	FSBR (6)	**0.74**	**0.25**	**0.75**	**0.74**
Zoo	CFS (9)	0.94	0.06	0.93	0.94
	CON (9)	0.94	0.06	0.94	0.93
	CAR (6)	0.94	0.06	0.92	0.94
	Relief-F (7)	0.94	0.05	0.94	0.93
	SVD (8)	**0.95**	0.05	**0.95**	**0.95**
	MOGA (6)	**0.95**	0.04	**0.95**	0.94
	FSBR (8)	**0.95**	**0.03**	0.94	**0.95**
Dermatology	CFS (9)	0.98	**0.01**	**0.99**	0.98
	CON (9)	0.98	**0.01**	0.98	0.97
	CAR (11)	0.98	0.02	0.97	0.98
	Relief-F (11)	0.98	0.02	0.98	0.98
	SVD (11)	0.98	0.02	0.98	0.97
	MOGA (9)	0.98	0.02	0.97	0.98
	FSBR (11)	**0.99**	**0.01**	0.98	**0.99**
Mushroom	CFS (4)	0.97	0.03	0.96	0.97
	CON (5)	**0.99**	**0.01**	0.98	**0.99**
	CAR (8)	**0.99**	**0.01**	**0.99**	0.98
	Relief-F (5)	0.98	0.02	0.97	0.98
	SVD (5)	0.96	0.04	0.96	0.95
	MOGA (5)	0.97	**0.01**	0.96	0.97
	FSBR (5)	**0.99**	**0.01**	**0.99**	0.98
Coil20	CFS (194)	0.80	0.19	0.81	0.80
	CON (194)	0.79	0.21	0.79	0.78
	CAR (201)	0.78	0.21	0.78	0.79
	Relief-F (198)	0.78	0.22	0.77	0.78
	SVD (156)	0.77	0.23	0.77	0.78
	MOGA (95)	0.83	0.16	0.83	0.83
	FSBR (156)	**0.84**	**0.16**	**0.83**	**0.84**
	CFS (201)	0.54	0.45	0.53	0.54
	CON (198)	0.53	0.45	0.53	0.52
	CAR (204)	0.53	0.46	0.52	0.53

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Orl	Relief-F (213)	0.54	0.46	0.54	0.53
	SVD (157)	0.54	0.46	0.54	0.54
	MOGA (110)	0.59	0.37	0.60	0.59
	FSBR (157)	**0.61**	**0.37**	**0.61**	**0.61**
Allaml	CFS (221)	0.82	0.17	0.82	0.82
	CON (230)	0.83	0.18	**0.83**	0.82
	CAR (201)	0.81	0.19	0.80	0.81
	Relief-F (210)	0.81	0.19	0.81	0.82
	SVD (167)	0.82	0.18	0.81	0.82
	MOGA (194)	0.82	0.18	0.82	0.81
	FSBR (167)	**0.84**	**0.15**	**0.83**	**0.84**
Leukemia	CFS (147)	0.83	0.16	0.81	0.83
	CON (159)	0.85	0.13	0.85	0.84
	CAR (147)	0.85	0.14	0.85	0.85
	Relief-F (159)	0.85	0.15	0.86	0.85
	SVD (127)	0.73	0.23	0.73	0.77
	MOGA (97)	**0.88**	**0.12**	0.87	**0.88**
	FSBR (127)	0.87	0.13	**0.88**	**0.88**

From Table 3.13 and 3.14, it has been observed that the FSBR method is superior than most of the other methods in terms of both number of features, average classification accuracy and other statistical measures. It is also proved that the proposed FSBR method is statistically significant.

3.3.2 Multiple Reducts Generation using Clustering Algorithm and Rough Set Theory(MRG):

In this section, the multiple feature subset selection method MRG [48] is described for selecting multiple feature subsets from dataset with losing any information. The method can handle real valued data.

The novel and heuristic method tries to find out multiple reducts based on the concepts of indiscernibility relation of rough set theory [17-20], graph theory [21] and clustering algorithm [22]. Here, the data mining problem is converted to graph theoretic problem and then multiple feature subset called multiple reduct in RST is generated. Many feature selection techniques use heuristics which may degrade the performance, but the proposed MRG method has a strong mathematical foundation and hence, produces better results.

In this method, the concept of Rough Set Theory, graph theory and clustering algorithm is used to generate the multiple reducts from the dataset. The concept of Indiscernibility

relation [17] and Clustering approach [22] is used to make partitions of objects into equivalence classes.

Indiscernibility relation of RST defined in Equation (2.7) is used for partitioning of objects of the decision system based on decision attribute only. Now, from the dataset by taking two conditional attributes at a time, partitioning of objects is done using *K*-means [127] or *K*-prototype [130] clustering algorithm based on the nature of the dataset. Simple K-means algorithm is used for the continuous valued dataset whereas *K*-protoype clustering algorithm is used for categorical dataset for the clustering purpose. Based on two sets of partition, connecting factor between two conditional attributes is computed and an attribute connecting set (*ACS*) containing all pair-wise connection of the attributes with respect to Decision attribute is obtained. Then attribute connection of *ACS* having connecting factor less than average connecting value are removed and an undirected weighted graph called Attribute Connecting Graph (*ACG*) is constructed based on the reduced set *ACS*. The *ACG*, therefore, represents the total connecting structure of the connecting set *ACS*. The connecting factor between two attributes C_i and C_j is denoted by *k* which signifies that both C_i and C_j together partitioned the objects which is (*k**100) % similar to that obtained only by the decision attribute *D*. Vertex with the sum of weights associated with the edges incident on a vertex is considered as the weighted degree of the vertex. Then average weighted degree or degree of connection of each vertex is calculated. Now, vertex with maximum degree of connection is removed with the adjustment of weighted degree of the vertices adjacent to it and stored the vertex in the reduct set. The process is repeated for modified graph until all the edges are removed from the graph, forming a compact set of attributes, called reduct. The method provides multiple reducts which is explained in detail in section 3.3.2.4.

The detail procedure of reduct generation is discussed below.

a. Partitioning the Objects of Decision System:

The objects are partitioned by two different ways:

- Partitioning of objects based on decision attribute using indiscernibility relation

Let *DS* = (*U*, *A*) be a decision system where *U* is the finite, non-empty set of objects and $A = C \cup D$ such that *C* and *D* are set of condition and decision attributes respectively. For any $P \subseteq A$, there exists a binary relation *IND(P)*, called indiscernibility relation and is defined in Equation (2.7). *IND*(*P*) is an equivalence relation which induces equivalence classes. The family of all equivalence classes of *IND* (*P*), i.e., partition determined by *P*, is denoted by *U*/*IND* (*P*) or simply *U*/*P*.Thus for decision attribute *D*, the equivalence classes are *U*/*D* obtained by *IND (D)* using the Equation (2.7). Let $U/D = CL^D = \{CL_1^D, CL_2^D, \ldots\ldots\ldots, CL_k^D\}$.

- Partitioning of objects by applying clustering algorithm on the projections of dataset

Let $C = \{C_1, C_2, \ldots\ldots, C_n\}$ be the set of conditional attributes. Now projection on the dataset *DS* for two attributes C_i and C_j is performed using the Equation (3.10) to obtain the projected dataset (*PDS*).

$$PDS = \prod_{C_i, C_j}(DS) \tag{3.10}$$

So *PDS* contains same number of objects as *DS*. Now the dataset *PDS* is clustered using *K*-means or *K*-prototype algorithm with *K* as the number of distinct values of decision attribute *D*.

Let the clusters obtain by C_i and C_j are

$CL^{ij} = \{ CL_1^{ij}, CL_2^{ij}, \ldots\ldots CL_k^{ij} \}$ for all $i, j = 1, 2, \ldots\ldots, n; i < j$.

b. Computation of Attribute Connecting Strength:

Here, computation of connecting strength between attributes C_i and C_j are made based on those two partitions obtained in section 3.3.2.1. Attributes C_i and C_j are totally connected with respect to *D* if there are one to one correspondence among the elements of CL^D and CL^{ij}.

But in real situation, it rarely occurs and so connecting power of attributes C_i and C_j is measured by introducing the connecting factor $\delta_f^{i,j}$ using the Equation (3.11), which measures the degree of connectivity of attributes between each other with respect to decision attribute.

$$\delta_f^{i,j} = \frac{1}{K} \sum_{CL_t^{ij} \in CL^{ij}} \frac{1}{CL_t^{ij}} \max_{\forall\, CL_P^D \in CL^D} \{CL_t^{ij} \cap CL_P^D\} \tag{3.11}$$

So $\delta_f^{i,j} = 1$; if C_i and C_j are totally connected with respect to *D*

< 1; otherwise.

Thus, for *n* conditional attributes, there are $\frac{n(n-1)}{2}$ pair wise connection of attributes with respect to *D* for the decision system *DS* in the form $\{C_i C_j \xrightarrow{\delta_f^{i,j}} D\}$. Let the attribute connecting set *ACS* = $\{C_i C_j \xrightarrow{\delta_f^{i,j}} D \;\forall\, i, j\}$ which consists of all possible pair wise connection of attributes. Now the average connecting factor δ_f is computed and the elements $C_i C_j \xrightarrow{\delta_f^{i,j}} D$ with $\delta_f^{i,j} < \delta_f$ are discarded, and the rest is considered as the modified attribute connecting set *MCS*.

c. Construction of Attribute Connecting Graph:

Now from the *MCS*, a weighted undirected graph *ACG* = (*V*, *E*) is constructed as follows:

- For each element $C_i C_j \xrightarrow{\delta_f^{i,j}} D \in MCS$

i. C_i and C_j are considered as vertices of the graph G i.e., $V = V \cup \{C_i\} \cup \{C_j\}$ Where $V = \{\emptyset\}$ initially.

ii. An edge (C_i, C_j) is drawn with weight $\delta_f^{i,j}$ i.e., $E = E \cup \{(C_i, C_j)\}$ where $E = \{\emptyset\}$ initially. Thus, E is a proper subset of $V \times V$.

This graph is called the attribute connecting graph *ACG* which represents how the attributes are connected to represent a decision system.

The graph formation algorithm "Weighted_Undirected_Graph_Formation" is described below:

Algorithm: Weighted_Undirected_Graph_Formation (*DS, ACG*)
Input: $DS = (U, A)$ where $C = \{C_1, C_2, \ldots Cn\}$
Output: Attribute Connecting graph $ACG = (V, E)$
Begin
$CL^D = \{CL_1^D, CL_2^D, \ldots\ldots\ldots, CL_k^D\}$ using (2.7), where $k = |D|$
$\delta_f = 0$ /*Average connection factor*/
for $i = 1$ ton do
for j = i+1 toto*n* do
PDS = $\prod_{Ci,Cj}(DS)$
$CL^{ij} = \{CL_1^{ij}, CL_2^{ij}, \ldots\ldots, CL_k^{ij}\}$ by clustering algorithm on *PDS*
Compute $\delta_f^{i,j}$ using equation (3.11)
$\delta_f = \delta_f + \delta_f^{i,j}$
end-for
end-for
$\delta_f = 2\delta_f / n(n\text{-}1)$ /*Average Connecting Factor*/
$V = \{\emptyset\}$, $E = \{\emptyset\}$
for $i = 1$ ton *do*
for j = i+1 toto*n* do
if $\delta_f^{i,j} > \delta_f$ then
$V = V \cup \{C_i\} \cup \{C_j\}$.
$E = E \cup \{(C_i, C_j)$ with weight $\delta_f^{i,j}\}$
end-if
end-for
end-for
Return $ACG = (V, E)$
End

c. Generation of Reduct:

The undirected weighted graph *ACG = (V, E)* has the weighted edges. The weight of an edge indicates the classification power of the attributes corresponding to the terminal nodes of the edge.

Higher the weight of an edge indicates better the classification power of the combined attributes (nodes). Now a term degree of connection of a node is defined as follows:

Definition: Degree of Connection of a Node:

Let *ACG = (V, E)* be an undirected weighted graph and $v_i \in V$ be a node. Then the degree of connection of a node v_i denoted by $dc(v_i)$ is defined as

$$dc(v_i) = \frac{1}{deg\ (vi)} \sum w_{ij} \,/\, (v_i, v_j) \in E \text{ and } w_{ij} \text{ is the weight of } (v_i, v_j) \qquad (3.12)$$

where $deg\ (v_i)$ is the degree [21] of the vertex v_i.

Here, higher the degree of connection implies the corresponding attribute is more important. So, the reduct is formed using following steps:

- Initially the attribute associated with the node *v* with the highest degree of connection is considered as reduct.
- Then the vertex *v* is removed from the attribute connecting graph (*ACG*). As a vertex is removed, so the 'degree of connection' of the vertices incident on the removed vertex are reduced by the weight associated with the corresponding edge.
- Thus, the graph *ACG* is modified, and the new attribute associated with the current highest degree of connection is added to the reduct and repeat the same process until all the edges are removed or the graph becomes empty.

Here, multiple reducts will be generated if more than one vertex has the highest degree of connection at some iteration. For example if after certain iteration the reduct set $R = \{C'_1, C'_2, \ldots, C'_r\}$ and for next iteration *j*-vertices v'_{r+1}, v'_{r+2},... v'_{r+j} has the highest degree of connection then after this iteration the reduct is $R = \{(C'_1, C'_2, \ldots C'_r, C'_{r+1}), (C'_1, C'_2, \ldots C'_r, C'_{r+2}), \ldots (C'_1, C'_2, \ldots C'_r, C'_{r+j})\}$. Thus, for a single reduct in previous iteration and j-vertices of highest degree of connection in *ACG*, *j*-number of reducts is obtained in the next iteration. This process provides us multiple numbers of reducts at the end of the iteration.

The detail algorithm for multiple reduct generation is given below:

Algorithm: Multiple_Reduct_Gen (*ACG*, *RED*)
Input: *ACG* = (*V*, *E*) with weight of edge $(v_i, v_j) \in E$ as *wt* (v_i, v_j)
Output: multiple reduct set *RED*
Begin
  RED=∅
  Repeat
    for each node $v_i \in V$ do
      Compute degree of v_i as deg (v_i)
      Compute degree of connection of v_i using Equation (3.12)
    end-for
  Let $V' = \{v'_{r+1}, v'_{r+2}, \ldots . v'_{r+j}\}$, the vertex set of highest degree of connection

RED = *RED* × {{C'_{r+1}},{ C'_{r+2}}, ….{C'_{r+j}}} /*here C'_{r+i} is the attribute corresponding to vertex v'_{r+i}∀*i*=1, 2, *j*/*
/*Modify *ACG*/*
for each vertex v'_{r+i}∀i=1, 2, *j* do
Let incident vertices are {v_1^i, v_2^i, ….. v_p^i}
for each *k* = 1 to *p* do
deg_con (v_k^i) = deg_con (v_k^i) - wt (v'_{r+i}, v_k^i)
Remove v'_{r+i} from *ACG* = (*V*, *E*)
/*Thus associated edges are removed*/
end-for
Until (*E* = = ∅)
Return (*RED*)
End

e. Illustrative Example of MRG Method:

Let a decision system *DS* consists of 8 objects {O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8} and 4 conditional attributes {C_1, C_2, C_3, C_4} and one decision attribute *D* with 2 decision classes.

i. At first *DS* is partitioned based on *D* using indiscernibility relation defined in Equation (2.7) and following equivalence classes are obtained :

CL_1^D = {O_2, O_3, O_4, O_5} and CL_2^D = {O_1, O_6, O_7, O_8}

ii. Projection on dataset *DS* for two attributes C_1 and C_2 using Equation (3.10) is taken and k-means clustering algorithm is applied on it with *k* = 2 that produce following two clusters:

CL_1^{12} = {O_2, O_3, O_4, O_8} and CL_2^{12} = {O_1, O_5, O_6, O_7}

iii. The connecting factor $\delta_f^{1,2}$ for two attributes C_1 and C_2 is calculated using the Equation (3.11)

$\delta_f^{1,2} = \frac{1}{2}\left\{\frac{1}{4} \times 3 + \frac{1}{4} \times 3\right\} = 0.75$

(iv) In this way, after applying the clustering algorithm on each pair wise attributes in {C_1, C_2, C_3, C_4}, an attribute connecting set (*ACS*) representing connection of every pair of conditional attributes to decision attribute is constructed.

$ACS = \{C_1C_2 \xrightarrow{0.75} D, C_1C_3 \xrightarrow{0.82} D, C_1C_4 \xrightarrow{0.85} D, C_2C_3 \xrightarrow{0.88} D, C_2C_4 \xrightarrow{0.90} D, C_3C_4 \xrightarrow{0.73} D\}$

v. The elements of *ACS* having connecting factor less than the average value are removed and modified *ACS* is formed.

Here, average connecting factor (δ_f) = 0.83. So, the modified *ACS* is

$ACS = \{\ C_1C_4 \xrightarrow{0.85} D, C_2C_3 \xrightarrow{0.88} D, C_2C_4 \xrightarrow{0.90} D\ \}$

vi. Then using 'Weighted_Undirected_Graph_Formation' algorithm discussed in section 3.3.2.3 an attribute connecting graph (*ACG*) is formed from modified *ACS*. Figure 3.8 represents the attribute connecting graph for the example data.

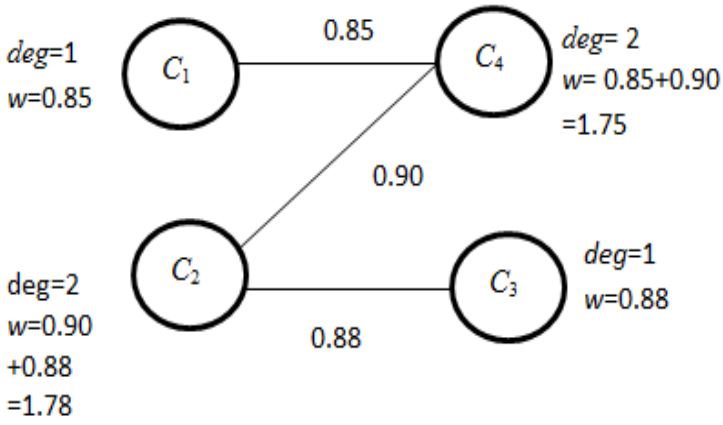


Figure 3.8: *ACG* achieved from modified *ACS*

vii. Now from *ACG* the *degree of connection* of each vertex *v* (here it is C_1, C_2, C_3 and C_4) are calculated using the Equation (3.12).

viii. From *ACG*, the vertex C_2 has the highest degree of connection and according to the 'Multiple_Reduct_Gen' algorithm, C_2 has been considered as the reduct *R* and removed it from *ACG* with the adjustment of degree of connection of the vertices adjacent to it. Figure 3.9 represents the modified *ACG*.

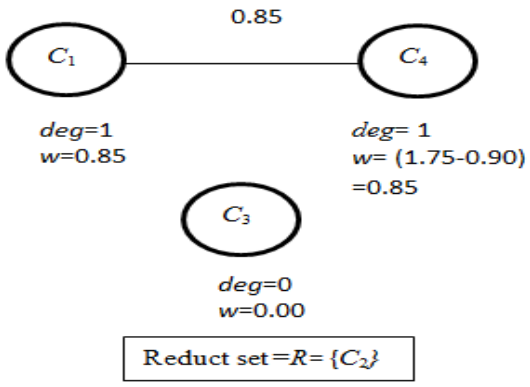


Figure 3.9: Modified *ACG* after first iteration

ix. Now in the next iteration, two vertices C_1 and C_4 have same *degree of connection* and so according to the 'Multiple_Reduct_Gen' algorithm, for a single reduct in previous iteration

and 2-vertices of highest degree of connection in *ACG*, 2 reducts are obtained. This process provides two reducts at the end of this iteration with $R = \{C_2C_1$ and $C_2C_4\}$ and *ACG* becomes empty as shown in Figure 3.10, which indicates the termination of the iteration.

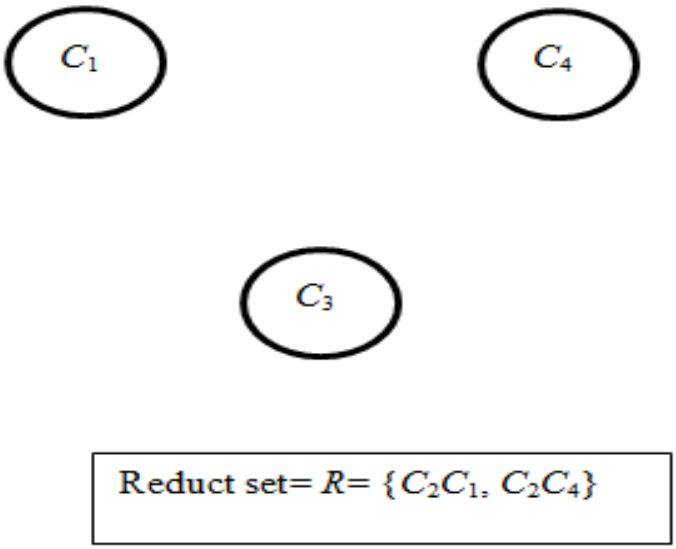


Figure 3.10: Modified ACG after second iteration

f. Experimental Results of MRG Method:

Experimental results presented here provide an evidence of effectiveness of MRG method to identify the set of significant feature subset as a multiple reduct for experimental dataset [27, 28] summarized in the section 2.2.

Then MRG method [48] and well-known feature selection methods such as, CFS [95], CON [213], CAR [214], Relief-F [215], SVD [216] and MOGA [217] are applied on the dataset for selecting the important features and the reduced datasets are classified based on considered base classifiers. 10-fold cross validation is used for the classification performance evaluation.

Number of attributes after applying mentioned feature selection methods and the accuracies (%) of the datasets by mentioned classifiers are computed and listed in Table 3.15, which shows the efficiency of the proposed MRG method.

The results of the best reduct of the multiple reducts generated by MRG method are shown in Table 3.15. Similar to the SRG method discussed in section 3.2.1.6, the statistical analysis is done using Wilcoxon's rank sum test [39] and results are listed in Table 3.15. To indicate the best performing algorithm a bold-faced font is used.

Table 3.15: Performance analysis of MRG and existing feature selection methods

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†	95.70†
	CON (8)	96.19†	97.11†	96.63†	94.94†	94.94†	94.30†	97.65†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.56†
	Relief-(9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†	97.23†
	SVD (7)	96.65†	96.56†	96.76†	95.78†	95.45†	95.98†	97.80†
	MOGA (7)	97.17†	96.65†	95.56†	96.64	95.78	95.87†	97.67†
	MRG (7)	**98.96**	**98.64**	**97.98**	**97.87**	**96.90**	**96.83**	**98.67**
Heart	CFS (8)	84.36†	84.75†	81.67†	81.11†	81.11†	81.67†	82.78†
	CON (11)	84.50†	84.44†	82.07†	81.48†	82.89†	79.55†	82.72†
	CAR (10)	83.36†	84.75†	81.67†	83.11†	82.11†	80.67†	82.34†
	Relief-10)	83.50†	84.44†	82.07†	81.48†	83.89†	79.59†	82.30†
	SVD (7)	83.45†	83.67†	83.98†	82.21†	83.21†	82.08†	83.87†
	MOGA (8)	84.67†	83.32†	83.22†	83.56†	83.87†	83.26†	84.98†
	MRG (7)	**85.22**	**85.31**	**84.37**	**84.98**	**84.93**	**84.97**	**85.83**
Glass	CFS (6)	43.92†	57.94†	79.91†	73.83†	68.69†	70.09†	66.02†
	CON (7)	47.20†	57.48†	78.50†	71.50†	64.20†	68.60†	64.65†
	CAR (8)	56.92†	58.94†	**80.91**	75.83†	69.69†	71.09†	68.54†
	Relief-(8)	57.20†	57.48†	79.50†	70.50†	63.20†	72.60†	67.74†
	SVD (4)	56.67†	57.75†	77.39†	71.56†	67.50†	74.45†	75.89†
	MOGA (7)	56.54†	57.76†	76.49†	72.45†	64.76†	70.89†	76.23†
	MRG (4)	**70.42**	**72.36**	72.98†	**78.13**	**75.19**	**77.42**	**80.56**
Zoo	CFS (9)	96.03†	93.06†	94.05†	94.04†	93.06†	93.06†	93.54†
	CON (9)	96.03†	93.03†	94.05†	94.04†	93.88†	94.32†	94.45†
	CAR (6)	94.05†	93.92†	93.32†	94.02†	94.07†	94.05†	95.45†
	Relief-(7)	95.03†	93.70†	93.01†	93.01†	94.12†	94.12†	95.03†
	SVD (6)	96.09†	94.67†	94.78†	94.67†	94.32†	94.67†	95.56†
	MOGA (6)	94.78†	94.23†	94.45†	95.21†	94.32†	94.34†	94.54†
	MRG (6)	**98.02**	**97.19**	**97.24**	**97.04**	**98.27**	**96.21**	**97.04**
Dermatology	CFS (9)	98.76†	97.42†	97.01†	98.06†	98.07†	98.62†	**99.09**
	CON (9)	98.52†	98.25†	95.56†	98.06†	98.86†	98.67†	98.45†
	CAR (11)	98.73†	98.30†	97.42†	98.31†	98.06†	98.07†	98.54†
	Relief-11)	98.72†	98.45†	95.56†	97.16†	98.76†	98.46†	98.45†
	SVD (8)	97.78†	98.03†	96.75†	97.09†	98.01†	97.89†	97.65†
	MOGA (9)	97.87†	97.89†	97.90†	97.05†	98.67†	97.90†	97.50†
	MRG (8)	**99.33**	**99.27**	**98.06**	**99.17**	**99.16**	**99.27**	99.06≈
	CFS (4)	97.52†	96.01†	96.52†	97.01†	97.01†	97.01†	97.04†

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†	95.70†
	CON (8)	96.19†	97.11†	96.63†	94.94†	94.94†	94.30†	97.65†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.56†
	Relief-(9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†	97.23†
	SVD (7)	96.65†	96.56†	96.76†	95.78†	95.45†	95.98†	97.80†
	MOGA (7)	97.17†	96.65†	95.56†	96.64	95.78	95.87†	97.67†
	MRG (7)	**98.96**	**98.64**	**97.98**	**97.87**	**96.90**	**96.83**	**98.67**
Mushroom	CON (5)	**98.52**	**98.85**	98.52≈	99.05†	98.16†	**99.86**	98.54≈
	CAR (8)	98.02≈	98.32≈	**99.02**	**99.65**	**99.23**	99.01†	98.45≈
	Relief-(5)	97.04†	98.03†	98.03†	98.13†	98.10†	98.10†	98.23≈
	SVD (4)	97.04†	97.23†	97.23†	97.83†	97.34†	87.64†	97.45†
	MOGA (5)	97.34†	95.45†	96.67†	96.34†	96.34†	96.45†	97.64†
	MRG (4)	**98.52**	98.55≈	98.39†	97.44†	98.54†	98.72†	**98.68**
Coil20	CFS (194)	78.12†	79.24†	80.01†	80.10†	79.25†	79.85†	80.98†
	CON (194)	79.60†	79.20†	78.00†	79.35†	79.34†	78.79†	80.21†
	CAR (201)	77.12†	78.24†	77.01†	80.10†	75.25†	78.85†	78.98†
	Relief-198)	77.60†	78.20†	77.80†	78.35†	77.34†	78.49†	80.21†
	SVD (119)	76.60†	76.20†	77.80†	75.35†	76.64†	78.49†	79.21†
	MOGA (95)	82.20†	79.99†	82.92†	83.02≈	83.02†	84.20†	84.71†
	MRG (119)	**84.60**	**82.90**	**84.80**	**83.48**	**84.98**	**86.76**	**85.56**
Orl	CFS (201)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	57.65†
	CON (198)	56.60†	53.20†	52.00†	51.35†	50.70†	52.89†	54.87†
	CAR (204)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	57.65†
	Relief-(213)	54.60†	54.20†	54.01†	54.25†	53.79†	53.89†	54.87†
	SVD (137)	52.12†	53.24†	53.01†	52.10†	53.80†	54.35†	57.65†
	MOGA (110)	59.60†	57.20†	59.00†	58.25†	**59.70**	59.09	**59.87**
	MRG (137)	**61.20**	**63.40**	**60.30**	**60.21**	59.34	**59.34**	59.32
Allaml	CFS (221)	81.12†	81.32†	82.62†	83.82†	82.21†	83.01†	83.80†
	CON (230)	79.12†	80.32†	81.62†	82.82†	80.21†	82.01†	83.80†
	CAR (201)	80.02†	80.32†	81.52†	82.72†	81.21†	82.02†	82.80†
	Relief-(210)	81.12≈	81.32†	82.62†	83.82†	82.21†	83.11†	83.80†
	SVD (173)	80.22†	81.62†	81.27†	82.92†	82.81†	83.19†	82.80†
	MOGA (194)	81.12≈	83.32†	84.72†	81.82†	82.01†	82.21†	84.50†
	MRG (173)	**81.46**	**84.32**	**85.98**	**85.36**	**85.98**	**84.23**	**86.32**
	CFS (147)	83.23†	84.23†	82.67†	82.02†	82.05†	83.23†	84.34†
	CON (159)	84.05†	85.87†	84.23†	84.12†	84.21†	84.67†	85.01†

Dataset	Methods (#features)	Classifiers (%)						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†	95.70†
	CON (8)	96.19†	97.11†	96.63†	94.94†	94.94†	94.30†	97.65†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.56†
	Relief-(9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†	97.23†
	SVD (7)	96.65†	96.56†	96.76†	95.78†	95.45†	95.98†	97.80†
	MOGA (7)	97.17†	96.65†	95.56†	96.64	95.78	95.87†	97.67†
	MRG (7)	**98.96**	**98.64**	**97.98**	**97.87**	**96.90**	**96.83**	**98.67**
Leukemia	CAR (147)	83.47†	85.23†	84.56†	84.53†	84.54†	84.56†	89.10†
	Relief-F (159)	84.23†	85.34†	84.12†	84.34†	84.34†	84.98†	87.50†
	SVD (99)	73.68†	76.32†	71.05†	71.02†	71.05†	73.68†	75.87†
	MOGA (97)	86.34†	85.90†	85.50†	85.78≈	87.12≈	86.23†	**89.80**
	MRG (99)	**87.90**	**86.42**	**87.32**	**85.98**	**87.45**	**87.34**	89.34≈

To show the effectiveness of the classifiers based on the reduced features, some statistical measurements given in Equation (2.25) to (2.28) in the section 2.6.2 are also performed and the average results for all seven classifiers are listed in Table 3.16.

Table 3.16: Statistical measures for MRG and different competitive algorithm

Dataset	Methods	Recall	Fallout	Specificity	F1Score
Wine	CFS (8)	0.96	0.04	0.95	0.96
	CON (8)	0.96	0.04	0.96	0.96
	CAR (8)	0.95	0.06	0.94	0.95
	Relief-F (9)	0.96	0.04	0.96	0.95
	SVD (7)	0.97	0.03	0.96	0.97
	MOGA (7)	0.97	0.03	0.97	0.97
	MRG (7)	**0.98**	**0.01**	**0.98**	**0.99**
Heart	CFS (8)	**0.98**	**0.02**	**0.98**	**0.97**
	CON (11)	0.82	0.18	0.83	0.82
	CAR (10)	0.83	0.16	0.82	0.83
	Relief-F (10)	0.83	0.17	0.83	0.83
	SVD (7)	0.82	0.17	0.82	0.83
	MOGA (8)	0.83	0.18	0.82	0.82
	MRG **(7)**	0.85	0.15	0.85	0.85
	CFS (6)	0.66	0.36	0.67	0.66
	CON (7)	0.65	0.36	0.65	0.64
	CAR (8)	0.69	0.31	0.68	0.69

Dataset	Methods	Recall	Fallout	Specificity	F1Score
Glass	Relief-F (8)	0.68	0.32	0.69	0.68
	SVD (4)	0.69	0.31	0.69	0.68
	MOGA (7)	0.68	0.32	0.68	0.68
	MRG (4)	**0.75**	**0.24**	**0.74**	**0.75**
Zoo	CFS (9)	0.94	0.06	0.93	0.94
	CON (9)	0.94	0.06	0.94	0.93
	CAR (6)	0.94	0.06	0.92	0.94
	Relief-F (7)	0.94	0.05	0.94	0.93
	SVD (6)	0.95	0.05	0.95	0.95
	MOGA (6)	0.95	0.04	0.95	0.94
	MRG (6)	**0.97**	**0.02**	**0.98**	**0.97**
Dermatology	CFS (9)	0.98	**0.01**	**0.99**	0.98
	CON (9)	0.98	**0.01**	0.98	0.97
	CAR (11)	0.98	0.02	0.97	0.98
	Relief-F (11)	0.98	0.02	0.98	0.98
	SVD (8)	0.98	0.02	0.98	0.97
	MOGA (9)	0.98	0.02	0.97	0.98
	MRG (8)	**0.99**	**0.01**	**0.99**	**0.99**
Mushroom	CFS (4)	0.97	0.03	0.96	0.97
	CON (5)	**0.99**	**0.01**	0.98	**0.99**
	CAR (8)	**0.99**	**0.01**	**0.99**	0.98
	Relief-F (5)	0.98	0.02	0.97	0.98
	SVD (4)	0.96	0.04	0.96	0.95
	MOGA (5)	0.97	0.01	0.96	0.97
	MRG (4)	0.98	**0.01**	0.97	0.98
Coil20	CFS (194)	0.80	0.19	0.81	0.80
	CON (194)	0.79	0.21	0.79	0.78
	CAR (201)	0.78	0.21	0.78	0.79
	Relief-F (198)	0.78	0.22	0.77	0.78
	SVD (119)	0.77	0.23	0.77	0.78
	MOGA (95)	0.83	0.16	0.83	0.83
	MRG (119)	**0.85**	**0.15**	**0.85**	**0.85**
	CFS (201)	0.54	0.45	0.53	0.54
	CON (198)	0.53	0.45	0.53	0.52
	CAR (204)	0.53	0.46	0.52	0.53
	Relief-F (213)	0.54	0.46	0.54	0.53

Dataset	Methods	Recall	Fallout	Specificity	F1Score
Orl	SVD (137)	0.54	0.46	0.54	0.54
	MOGA (110)	0.59	0.37	0.60	0.59
	MRG (137)	**0.60**	**0.34**	**0.60**	**0.60**
Allaml	CFS (221)	0.82	0.17	0.82	0.82
	CON (230)	0.83	0.18	0.83	0.82
	CAR (201)	0.81	0.19	0.80	0.81
	Relief-F (210)	0.81	0.19	0.81	0.82
	SVD (173)	0.82	0.18	0.81	0.82
	MOGA (194)	0.82	0.18	0.82	0.81
	MRG (173)	0.85	0.12	0.84	0.85
Leukemia	CFS (147)	0.83	0.16	0.81	0.83
	CON (159)	0.85	0.13	0.85	0.84
	CAR (147)	0.85	0.14	0.85	0.85
	Relief-F (159)	0.85	0.15	0.86	0.85
	SVD (99)	0.73	0.23	0.73	0.77
	MOGA (97)	**0.88**	0.12	**0.87**	**0.88**
	MRG (99)	0.87	**0.12**	0.86	0.87

3.3.3 Comparison of the FSBR and MRG Methods:

The comparison of FSBR [47] and MRG [48] methods, based on experimental datasets, is shown in Table 3.17. Both the methods generate multiple feature subsets or reducts. The method FSBR selects feature subset based on RST concepts. On the other hand, in MRG method the feature subset is selected based on the integrating concept of RST, graph theory and clustering algorithm. It is observed that the cardinality of the feature subset of MRG method is less and gives the highest average classification accuracy for the considered classifiers in most of the cases. FSBR method also gives better classification accuracies for Coil20 and Orl dataset and comparable classification accuracies for other datasets. Less computational time is required for the execution of MRG method than FSBR method.

Table 3.17: Comparison of FSBR and MRG methods

Dataset	#Selected features		Average Accuracy (%)		Computational time(sec)	
	FSBR	MRG	FSBR	MRG	FSBR	MRG
Wine	**6**	7	97.18	**97.97**	7.01	**6.23**
Heart	9	**7**	84.67	**85.08**	5.24	**3.89**
Glass	6	**4**	73.97	**75.29**	3.79	**2.95**
Zoo	8	**6**	95.34	**97.28**	2.01	**1.92**

Dataset	#Selected features		Average Accuracy (%)		Computational time(sec)	
	FSBR	**MRG**	**FSBR**	**MRG**	**FSBR**	**MRG**
Dermatology	11	**8**	99.00	**99.01**	35.39	**32.32**
Mushroom	5	**4**	**98.87**	98.44	300.01	**287.69**
Coil20	156	**119**	83.78	**84.76**	1200.02	**1000.32**
Orl	157	**137**	**60.90**	60.44	930.05	**887.65**
Allaml	**167**	173	84.15	**84.82**	2013.87	**1897.76**
Leukemia	127	**99**	86.64	**87.37**	3467.98	**2887.98**

3.4 Summary:

The concept of rough set offers a sound theoretical foundation for constructing the reduct sets, which is very effective for dimensionality reduction and feature selection from the static data. Feature selection through reduct generation is the main issue of this chapter. Since, the method of reduct generation is NP-hard; heuristic methods are developed to create single and multiple reduct. In the chapter four simple but efficient feature selection methods are proposed. The main objective is to select informative features that are highly correlated with the class and sufficient to discriminate the objects of different class. Among four feature selection methods, two are novel single reduct generation methods.

First method is based only on RST (SRG) for selecting important single feature subset to classify datasets accurately. Second method is based on RST and graph theory (GRG) for selecting important single feature subset to classify datasets.

Both the single feature subset selection methods, SRG and GRG are compared among themselves as well as with several standard existing feature selection methods in terms of number of selected features, classification accuracy and other statistical measures applying classifiers on reduced datasets to show their effectiveness.

SRG and GRG method outperforms in most of the cases and gives higher classification accuracies with respect to other standard feature selection methods. Between SRG and GRG method, SRG method is better in terms of selected feature, computational time, classification accuracy and other statistical measures. The novelty of these two approaches is the absence of search process in comparison with other algorithms which requires long computational time.

The chapter also presents two multiple feature subset selection methods. In many times, it is noticed that a set of feature subset rather than single feature subset is more important for classification purpose. So, the work describes a multiple feature subset (multiple reduct) selection method (FSBR) using the concept of RST.

Method uses the concept of discernibility matrix, attribute dependency of the rough set theory. FSBR method generates a compact set of reduct with less number of features in a feature subset with good classification accuracy for experimental datasets. Another method

(MRG) has also been proposed to select multiple informative feature subsets, by using RST, graph theory and clustering algorithm.

Both the multiple feature subset selection methods, MRG and FSBR are compared among themselves as well as with several existing feature selection methods to show their effectiveness in terms of selected feature, classification accuracies and other statistical measures on reduced data by some classification methods.

FSBR and MRG outperform in most of the cases and gives higher classification accuracies with respect to other standard feature selection methods.

Between FSBR and MRG, it is observed that the cardinality of the feature subset obtained by MRG method is less, requires lesser execution time and gives higher average classification accuracy than FSBR method for most of the datasets due to strong mathematical support of the MRG algorithm. But FSBR algorithm is also performed better for Mushroom and Orl datasets with higher classification accuracies and close contender to MRG method for other datasets.

ISBN: 978-93-90847-37-2

Chapter 4

Feature Selection in Dynamic Environment

4.1 Introduction:

In recent years, dimension of datasets are growing rapidly in many applications which brings great difficulty to data mining and pattern recognition. As datasets changes with time, it is very time consuming or even infeasible to run repeatedly a knowledge acquisition algorithm. Incremental learning [50-56] is a technique where the learning process occurs whenever new data comes and added with the existing data. A learning algorithm [225] is considered as an incremental learning algorithm, if the training sets such as $t_1 \ldots t_n$, generates a series of hypotheses as $H_0, H_1, \ldots, H_n$, and $H \quad +_1$ depends only on H_i and the current training example t_i. But the resultant hypothesis is applicable for all the training data seen so far. So, it reduces both the space and time complexity with respect to data storing and processing.

The difference between incremental learning and traditional machine learning is that the former does not consider the availability of a sufficient training dataset before the learning process, but the training data comes with the varied time. For example, human learning is also incremental. People gather knowledge, learned from facts, and incrementally update the knowledge base when new observations become presented. As we know, due to sequential flow of information, limited memory space and processing power humans must learn incrementally as biological systems are able to always learn through their lifetime and gather knowledge over time. A key objective of machine learning research is the dimension reduction of the dataset for relevant feature selection applied prior to extract interesting rules and patterns from the large repository of data in dynamic environment. Same dimension reduction method used in old dataset may be applied on incremental dataset, but it unnecessarily analyzes the previous one which is already reduced and ready for mining process. In dynamic environment, newly generated data together with the information extracted from the previous data are analyzed to select the important features of whole dataset. As a result, efficiency and acceptability of the system increases. Incremental learning is applicable in both supervised and unsupervised domain.

Rough Set Theory (RST) [17-20, 51], a soft computing tool to imperfect knowledge, helps to select the important features in terms of the static as well as dynamic reduct. Dynamic reducts can put up better performance in very large datasets as well as effectively enhance the ability to accommodate noisy data. The reduct generation method based on standard Rough Set Theory [17-20] are effective to some extent but there are some problems that has to be solved in practice especially for incremental dataset which are time variant [50-56]. The problem of attribute reduction for incremental data falls under the class of Online Algorithms and hence demands a dynamic solution to reduce re-computation of the reducts. To handle the dynamic data, several incremental feature selection algorithms [50-56] have been proposed.

A common characteristic of these algorithms is that they are appropriate for the new data that is being generated one by one. When many objects are produced at a time, those algorithms may not be efficient enough, as repetitive execution is needed to handle the new group of objects. Guan (2009) [52] developed an incremental updating algorithm to find an attribute reduction set in decision tables based on the discernibility matrix, where the added number of groups of objects in the decision tables changes the discernibility matrix and updates the attribute reduction set accordingly. Hu et al. (2005) [53] has developed an incremental attribute reduction algorithm, based on the elementary sets, which can determine the attribute reduction set from a dynamic information system. Wang et al. (2013) [226] has developed an attribute reduction algorithm for datasets with dynamic data values using the concept of information entropy. Deng (2010) [227] has presented a method of attribute reduction by generating a parallel reduct using the concept of positive region and the attribute significance. Bazan et al. (1996) [51] introduces the concept of dynamic reducts to handle large amounts of data or incremental data, in which the quality of the dynamic reduct is measured using the stability coefficients. Jun Xie et al. (2013) [228] has developed an improved incremental attribute reduction algorithm by exploring the concept of relative positive region, which can handle both the incremental attributes and incremental samples. Liang et al. (2014) [214] proposed a group incremental method for feature selection in the framework of rough set theory. The method uses information entropy as a parameter for measuring the feature significance. Dun Liu et al. (2014) [229] proposed a matrix based incremental approach in dynamic incomplete information systems for knowledge discovery. In this method, three types of matrices, namely support matrix, accuracy matrix and coverage matrix under four different extended relations such as tolerance relation, similarity relation, limited tolerance relation and characteristics relation are introduced to incomplete information systems for inducing knowledge dynamically. Though the method is helpful to deal with the missing and incomplete data, but it is time consuming for learning knowledge for datasets with high volumes, as addition and deletion of individual objects take place for knowledge discovery in this type of incremental model. Xu et al. (2011) [230] proposed an incremental attribute reduction method based on 0-1 integer programming when multiple objects enter into an information system incrementally. The method updates the old reduct based on the newly entered objects in the system. Though the method is helpful for attribute reduction in incremental environment, but the performance of the method is not very significant compared to other incremental algorithms. Shu et al. (2014) [231] proposed a method for incremental feature selection which is very important for dynamic incomplete data. This method employed a rough set theory based incremental approach to compute the new positive region when objects with varied feature values are added dynamically. Based on the calculated positive region value, features are selected incrementally. In this paper, two efficient incremental feature selection algorithms are proposed, one considering single new object at a time and the other considering multiple objects or group of new objects entering into the system. Single new object based incremental feature selection technique is more time consuming and also provide poor feature selection performance compared to multiple objects based incremental feature selection algorithm.

In this chapter, two different approaches to incremental feature selection methods [49, 57] have been proposed, each of which has novelty in feature selection. The first method called DRED generates multiple feature subset as dynamic reducts using the property of RST. The second method, called IFS, a group incremental feature selection using RST and Genetic Algorithm is proposed for generation of single optimum feature subset.

Results of the proposed incremental methods are evaluated and compared with standard existing static attribute reduction techniques such as, CFS [95], CON [213], CAR [214], Relief-F [215], SVD [216] and MOGA [217] and some popular incremental attribute reduction techniques such as IUAARI [50], IUAARS [52], GIARC [214], Xu et al.[230] and Shu et al.[231] to explain the effectiveness of the proposed methods for experimental benchmark datasets [27, 28]. Important features are selected by the proposed incremental methods, existing static and incremental methods and then the reduced datasets are classified on various well known classifiers [37] such as Naïve Bayes (NB) [7], Support vector machine (SVM) [6], K-nearest neighbors K-NN [37], Bagging [191], Tree based classifier (J48) [5], Multilayer Perceptron (MLP) [3] available at "Weka" tool [218] and an incremental classifier IPSO [63], implemented by me. SVM is used with RBF kernel, K value of K-NN is set to the square root of sample size of data. Statistical analysis is also performed to show the efficiency of the proposed methods. The main focus of the experiments is on the three issues: number of features, classification accuracy and execution efficiency.

The remaining part of the chapter is organized as follows: The incremental feature subset selection methods based on RST concepts and genetic algorithm are described and their performances are compared in section 4.2. The chapter is summarized in section .

4.2 Incremental Feature Subset Selection:

For the incremental data, running a learning algorithm in a repetitive manner is a difficult as well as extremely time-consuming task. There are a number of methods [17-20] that have discussed different approaches to generating reduct for static data or time invariant data. However, the methods are developed for datasets in batch mode and are not capable of considering the newly added data subsets. Thus, if a new dataset arrives, the algorithm has to be re-run entirely to consider the newly added dataset in the computation, which is impractical for larger datasets. The important and relevant feature selection [17-20] is necessary from these dynamic data in a lesser time to reduce the complexity of the subsequent data mining tasks. As finding important features by exhaustive search of all possible combination of features is an NP-complete problem [156], so efficient heuristics are proposed [50-56] for important feature subset selection in the dynamic environment. In this section, two incremental feature subset selection methods [49, 57] are proposed by which important features in terms of reduct are selected from the dataset.

First method [49] provides multiple reduct based on the concepts of RST only while the other method [57] generates single reduct based on the concepts of RST and genetic algorithm (GA) in an incremental way.

4.2.1 Dynamic Reduct Generation using Rough Set Theory (DRED):

Feature selection methodology in dynamic environment is necessary as it reduces both space and time complexity to determine features responsible for classifying the objects, which be included in learning network and provide information about class related features. The incremental feature selection technique is used in dynamic environment where newly generated group of data, together with the knowledge extracted from the previous data are analyzed to select the most relevant features of the entire dataset.

Here, an incremental feature selection method (DRED) [49] has been proposed for selecting important feature subsets as multiple reducts from the incremental dataset for classifying objects and the generated reducts preserve the property of the whole decision system.

The method (DRED) [49] can compute the dynamic reduct from the incremental dataset using the concept of Rough Set Theory [17-20]. The concepts of discernibility relation and attribute dependency of Rough Set Theory are used for generation of dynamic reduct set which are discussed in section 3.3.1. In DRED [49], FSBR algorithm [47] discussed in section 3.3.1 has been used to generate dynamic reduct from the incremental data. The main objective of the method is to run FSBR algorithm [47] in incremental way to reduce the computational time of FSBR algorithm [47] to generate feature subset without compromising the classification accuracy. In DRED, to apply the concept of dynamic data the original decision system $DS = (U, A, D)$ where A = set of conditional attributes and D = Decision attribute and U is the set of objects, is divided into two sub systems namely $DS_{old} = (U_1, A, D)$ and $DS_{new} = (U_2, A, D)$ as old and new subsystems respectively. When the DRED algorithm is first run for the initial subsystem DS_{old}, no previous reduct information is available; so, application of FSBR algorithm [47] is applied on DS_{old} generates a reduct set *RED* of reducts from the old subsystem. Subsequently, when newly arrived decision subsystem $DS_{new} = (U_2, A, D)$ is become available then the previous reduct set *RED* with the new subsystems determines a set *DR* of dynamic reducts of the whole system $DS = DS_{old} \cup$ DSnew using DRED algorithm.

The DRED algorithm is given below.

a. DRED Algorithm:

Algorithm: DRED (DS_{old}, DS_{new}, DR)

Input: Reduct set *RED* of decision subsystem $DS_{old} = (U_1, A, D)$ and newly arrived decision subsystem $DS_{new} = (U_2, A, D)$

Output: Dynamic Reduct set *DR* of $DS = DS_{old} \cup DS_{new}$

$DR = \phi$

for each reduct R in RED do

if $\gamma_R(D) = \gamma_A(D)$ with respect to DS_{new} then

$DR = DR \cup R$

else

Apply FSBR algorithm on DS_{new} considering R as the core set CR

if FSBR algorithm generates a reduct R' then

$DR = DR \cup R'$

end-if

end-for

b. Experimental Results of DRED Method:

The proposed method computes multiple feature subsets in terms of reduct for experimental benchmark datasets summarized in section 2.2 in an incremental way. At first, all the attributes are discretized by ChiMerge [219] discretization algorithm. The proposed DRED method is compared with standard static attribution reduction techniques and some popular incremental attribute reduction techniques mentioned in section 4.1. The main focus of the experiments was on the three issues: number of features, classification accuracy and execution efficiency. Here, 80% of each dataset is considered as old/existing data and rest 20% of data is considered as incremental data. Wilcoxon's rank sum test [39] is also performed on the result to show the statistical significance of the method. As DRED is a technique for multiple feature subset selection, so here all the results are given based on the best feature subset selected by DRED method.

- **Comparison with Static Attribute Reduction Techniques:**

To judge the effectiveness and the efficiency of the proposed DRED method, it is compared with common standard static or non-incremental attribute reduction methods mentioned in section 4.1. Original number of attributes, number of attributes after applying proposed and existing static feature selection methods and the accuracies (%) of the reduced datasets by mentioned classifiers are computed and listed in Table 4.1. To test whether the DRED method is statistically significant or not wilcoxon's rank sum test [39] is carried out with *p* value as 0.05 (or a significance level of 5%) to validate if the result obtained by the best performing algorithm differs from the others in a statistically significant way. The test confirms if the final accuracy obtained by an algorithm is statistically and significantly different from that of the best performing algorithm on some classification problem. Thus, if the performance of an algorithm is differing from the best result with a p value ≤ 0.05 then the mean error of the first one is marked with a '†' symbol, otherwise the two performances are considered as equivalent and the difference is not statistically significant, and the mean error is marked with a '≈' symbol, as shown in Table 4.1. To indicate the best performing algorithm a bold-faced font is used.

Table 4.1: Performance Comparison of DRED and Static Feature Selection Methods

Dataset (#Original features)	**Methods (#features)**	**Classifiers (%)**						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†	95.70†
	CON (8)	96.19†	97.11≈	96.63†	94.94†	94.94†	94.30†	97.65†

Dataset (#Original features)	**Methods (#features)**	**Classifiers (%)**						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine (13)	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†	96.56†
	Relief-F (9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†	97.23†
	SVD (7)	96.45†	96.46†	96.56†	95.78†	95.35†	95.78†	97.30†
	MOGA (7)	97.17†	96.65≈	95.56†	**96.64**	95.78†	95.87†	97.67†
	FSBR (6)	**98.65**	97.45≈	97.01≈	96.36≈	96.61≈	96.50†	98.04≈
	DRED (7)	98.31†	**97.75**	**97.75**	96.46≈	**97.19**	**97.19**	**98.10**
Heart (13)	CFS (8)	84.36†	84.75†	81.67†	81.11†	81.11†	81.67†	82.78†
	CON (11)	84.50†	84.44†	82.07†	81.48†	82.89†	79.55†	82.72†
	CAR (10)	83.36†	84.75†	81.67†	83.11†	82.11†	80.67†	82.34†
	Relief-F (10)	83.50†	84.44†	82.07†	81.48†	83.89†	79.59†	82.30†
	SVD (10)	83.55†	83.77†	**83.99**	82.31†	83.61†	82.28†	83.97†
	MOGA (8)	84.67†	83.32†	83.22≈	83.56†	83.87†	83.26≈	**84.98**
	FSBR (9)	**85.72**	**85.12**	83.94≈	**84.58**	**84.90**	**83.49**	**84.98**
	DRED (10)	82.96†	84.44†	80.37†	82.59†	82.22†	78.51†	84.90≈
Glass (9)	CFS (6)	43.92†	57.94†	79.91†	73.83†	68.69†	70.09†	66.02†
	CON (7)	47.20†	57.48†	78.50†	71.50†	64.20†	68.60†	64.65†
	CAR (8)	56.92†	58.94†	80.91†	75.83†	69.69†	71.09†	68.54†
	Relief-F (8)	57.20†	57.48†	79.50†	70.50†	63.20†	72.60†	67.74†
	SVD (8)	57.79†	58.75†	76.39†	72.56†	67.70†	75.45†	76.89†
	MOGA (7)	56.54†	57.76†	76.49†	72.45†	64.76†	70.89†	76.23†
	FSBR (6)	65.73†	63.44†	**83.57**	77.53†	72.30†	78.00†	77.23†
	DRED (8)	**68.73**	**65.34**	44.34†	**78.93**	**73.30**	**78.32**	**78.30**
Zoo (16)	CFS (9)	96.03†	93.06†	94.05†	94.04†	93.06†	93.06†	93.54†
	CON (9)	96.03†	93.03†	94.05†	94.04†	93.88†	94.32†	94.45†
	CAR (6)	94.05†	93.92†	93.32†	94.02†	94.07†	94.05†	95.45†
	Relief-F (7)	95.03†	93.70†	93.01†	93.01†	94.12†	94.12†	95.03†
	SVD (5)	96.19†	94.77†	94.88†	94.67†	94.42†	94.77†	95.76†
	MOGA (6)	94.78†	94.23†	94.45†	**95.21**	94.32†	94.34†	94.54†
	FSBR (8)	**97.04**	**95.05**	**95.05**	94.06†	96.03†	94.07†	**96.09**
	DRED (5)	96.03†	87.12†	94.05†	93.06†	**97.02**	**98.01**	96.02≈
Dermatology (33)	CFS (9)	96.03†	93.06†	94.05†	94.04†	93.06†	93.06†	93.54†
	CON (9)	96.03†	93.03†	94.05†	94.04†	93.88†	94.32†	94.45†
	CAR (6)	94.05†	93.92†	93.32†	94.02†	94.07†	94.05†	95.45†
	Relief-F (7)	95.03†	93.70†	93.01†	93.01†	94.12†	94.12†	95.03†
	SVD (5)	96.19†	94.77†	94.88†	94.67†	94.42†	94.77†	95.76†
	MOGA (6)	94.78†	94.23†	94.45†	**95.21**	94.32†	94.34†	94.54†
	FSBR (8)	**97.04**	**95.05**	**95.05**	94.06†	96.03†	94.07†	**96.09**

Dataset (#Original features)	**Methods (#features)**	**Classifiers (%)**						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
	DRED (5)	96.03†	87.12†	94.05†	93.06†	**97.02**	**98.01**	96.02≈
Mushroom (21)	CFS (4)	97.52†	96.01†	96.52†	97.01†	97.01†	97.01†	97.04†
	CON (5)	98.52†	98.85†	98.52†	99.05†	98.16†	**99.86**	98.54≈
	CAR (8)	98.02†	98.32†	99.02≈	99.65≈	**99.23**	99.01	98.45≈
	Relief-F (5)	97.04†	98.03†	98.03†	98.13†	98.10†	98.10†	98.23≈
	SVD (4)	97.14†	97.33†	97.13†	97.93†	97.24†	87.74†	97.65†
	MOGA (5)	97.34†	95.45†	96.67†	96.34†	96.34†	96.45†	97.64†
	FSBR (5)	**99.30**	**99.02**	**99.34**	**99.78**	98.25†	98.08†	98.34≈
	DRED (4)	99.02≈	98.54†	96.76†	97.66†	97.78†	96.46†	**98.64**
Coil20(1024)	CFS (194)	78.12†	79.24†	80.01†	80.10†	79.25†	79.85†	80.98†
	CON (194)	79.60†	79.20†	78.00†	79.35†	79.34†	78.79†	80.21†
	CAR (201)	77.12†	78.24†	77.01†	80.10†	75.25†	78.85†	78.98†
	Relief-F (198)	77.60†	78.20†	77.80†	78.35†	77.34†	78.49†	80.21†
	SVD (166)	78.54†	78.31†	78.90†	76.95†	76.54†	78.79†	79.91†
	MOGA (95)	82.20†	79.99†	82.92†	83.02†	83.02≈	84.20≈	84.71
	FSBR (156)	**84.87**	82.76≈	83.45≈	**84.76**	83.23≈	84.34†	**83.09**
	DRED (166)	**84.87**	**82.96**	**83.55**	84.56≈	**83.52**	**85.34**	**83.09**
Orl (1024)	CFS (201)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	57.65†
	CON (198)	56.60†	53.20†	52.00†	51.35†	50.70†	52.89†	54.87†
	CAR (204)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†	57.65†
	Relief-F (213)	54.60†	54.20†	54.01†	54.25†	53.79†	53.89†	54.87†
	SVD (212)	54.12†	53.94†	57.01†	54.10†	53.80†	54.95†	58.65†
	MOGA (110)	59.60†	57.20†	59.00†	58.25†	59.70†	59.09†	59.87†
	FSBR (210)	**61.65**	60.45≈	61.23≈	**60.12**	60.23†	60.23†	62.45†
	DRED (212)	**61.65**	**60.95**	**61.33**	60.02≈	**61.32**	**60.53**	**63.45**
Allaml (7129)	CFS (221)	81.12†	81.32†	82.62†	83.82†	82.21†	83.01†	83.80†
	CON (230)	79.12†	80.32†	81.62†	82.82†	80.21†	82.01†	83.80†
	CAR (201)	80.02†	80.32†	81.52†	82.72†	81.21†	82.02†	82.80†
	Relief-F (210)	81.12†	81.32†	82.62†	83.82†	82.21†	83.11†	83.80†
	SVD (169)	80.22†	81.62†	81.27†	82.92†	82.81†	83.19†	82.80†
	MOGA (194)	81.12†	83.32†	84.72†	81.82†	82.01†	82.21†	**84.50**
	FSBR (167)	**82.67**	**84.65**	85.89†	84.54†	83.45≈	84.54≈	83.34†
	DRED (169)	**82.87**	**84.65**	**86.89**	**85.54**	**83.65**	**84.64**	83.24†
Leukemia	CFS (147)	83.23†	84.23†	82.67†	82.02†	82.05†	83.23†	84.34†
	CON (159)	84.05†	85.87†	84.23†	84.12†	84.21†	84.67†	85.01†
	CAR (147)	83.47†	85.23†	84.56†	84.53†	84.54†	84.56†	89.10
	Relief-F (159)	84.23†	85.34†	84.12†	84.34†	84.34†	84.98†	87.50†

Dataset (#Original features)	**Methods (#features)**	**Classifiers (%)**						
		NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**IPSO**
(7070)	SVD (130)	73.68†	76.32†	71.05†	71.02†	71.05†	73.68†	75.87†
	MOGA (97)	86.34†	85.90†	85.50†	85.78†	87.12≈	**86.23**	**89.80**≈
	FSBR (127)	87.01≈	86.98†	86.32†	86.05≈	87.34≈	**86.23**	86.57†
	DRED (130)	**87.51**	**87.32**	**87.02**	**86.55**	**87.44**	**86.23**	86.57†

As accuracy is not only the measurement of effectiveness of the classifiers, some statistical measurements given in Equation (2.25) to Equation (2.28) are also performed and the average results for all seven classifiers are listed in Table 4.2.

Table 4.2: Statistical Measure for DRED and Static Feature Selection Methods

Dataset	**Methods(#features)**	**Recall**	**Fall_out**	**Specificity**	**F1_Score**
Wine	CFS (8)	0.96	0.04	0.95	0.96
	CON (8)	0.96	0.04	0.96	0.96
	CAR (8)	0.95	0.06	0.94	0.95
	Relief-F (9)	0.96	0.04	0.96	0.95
	SVD (7)	0.97	0.03	0.96	0.97
	MOGA (7)	0.97	0.03	0.97	0.97
	FSBR (6)	0.97	0.02	0.97	0.98
	DRED (7)	**0.98**	**0.02**	**0.98**	**0.98**
Heart	CFS (8)	**0.98**	**0.02**	**0.98**	**0.97**
	CON (11)	0.82	0.18	0.83	0.82
	CAR (10)	0.83	0.16	0.82	0.83
	Relief-F (10)	0.83	0.17	0.83	0.83
	SVD (10)	0.82	0.17	0.82	0.83
	MOGA (8)	0.83	0.18	0.82	0.82
	FSBR (9)	0.85	0.15	0.84	0.85
	DRED (10)	0.82	0.16	0.83	0.82
Glass	CFS (6)	0.66	0.36	0.67	0.66
	CON (7)	0.65	0.36	0.65	0.64
	CAR (8)	0.69	0.31	0.68	0.69
	Relief-F (8)	0.68	0.32	0.69	0.68
	SVD (8)	0.69	0.31	0.69	0.68
	MOGA (7)	0.68	0.32	0.68	0.68
	FSBR (6)	**0.74**	**0.25**	**0.75**	**0.74**

Dataset	Methods(#features)	Recall	Fall_out	Specificity	F1_Score
	DRED (8)	0.70	0.28	0.71	0.70
Zoo	CFS (9)	0.94	0.06	0.93	0.94
	CON (9)	0.94	0.06	0.94	0.93
	CAR (6)	0.94	0.06	0.92	0.94
	Relief-F (7)	0.94	0.05	0.94	0.93
	SVD (5)	0.95	0.05	0.95	0.95
	MOGA (6)	0.95	0.04	0.95	0.94
	FSBR (8)	**0.95**	**0.03**	**0.94**	**0.95**
	DRED (5)	0.94	0.06	0.95	0.94
Dermatology	CFS (9)	0.98	0.01	0.99	0.98
	CON (9)	0.98	0.01	0.98	0.97
	CAR (11)	0.98	0.02	0.97	0.98
	Relief-F (11)	0.98	0.02	0.98	0.98
	SVD (11)	0.98	0.02	0.98	0.97
	MOGA (9)	0.98	0.02	0.97	0.98
	FSBR (11)	**0.99**	**0.01**	**0.98**	**0.99**
	DRED (8)	0.98	0.02	0.97	0.98
Mushroom	CFS (4)	0.97	0.03	0.96	0.97
	CON (5)	0.99	0.01	0.98	0.99
	CAR (8)	**0.99**	**0.01**	**0.99**	**0.98**
	Relief-F (5)	0.98	0.02	0.97	0.98
	SVD (4)	0.96	0.04	0.96	0.95
	MOGA (5)	0.97	0.01	0.96	0.97
	FSBR (5)	**0.99**	**0.01**	**0.99**	**0.98**
	DRED (4)	0.98	0.02	0.98	0.98
Coil20	CFS (194)	0.80	0.19	0.81	0.80
	CON (194)	0.79	0.21	0.79	0.78
	CAR (201)	0.78	0.21	0.78	0.79
	Relief-F (198)	0.78	0.22	0.77	0.78
	SVD (156)	0.77	0.23	0.77	0.78
	MOGA (95)	0.83	0.16	0.83	0.83
	FSBR (156)	**0.84**	**0.16**	**0.83**	**0.84**
	DRED (166)	**0.84**	**0.16**	**0.83**	**0.84**
	CFS (201)	0.54	0.45	0.53	0.54

Dataset	Methods(#features)	Recall	Fall_out	Specificity	F1_Score
Orl	CON (198)	0.53	0.45	0.53	0.52
	CAR (204)	0.53	0.46	0.52	0.53
	Relief-F (213)	0.54	0.46	0.54	0.53
	SVD (157)	0.54	0.46	0.54	0.54
	MOGA (110)	0.59	0.37	0.60	0.59
	FSBR (157)	**0.61**	**0.37**	**0.61**	**0.61**
	DRED (212)	**0.61**	**0.37**	**0.61**	**0.61**
Allaml	CFS (221)	0.82	0.17	0.82	0.82
	CON (230)	0.83	0.18	0.83	0.82
	CAR (201)	0.81	0.19	0.80	0.81
	Relief-F (210)	0.81	0.19	0.81	0.82
	SVD (167)	0.82	0.18	0.81	0.82
	MOGA (194)	0.82	0.18	0.82	0.81
	FSBR (167)	0.84	0.15	0.83	0.84
	DRED (169)	**0.85**	**0.14**	**0.85**	**0.85**
Leukemia	CFS (147)	0.83	0.16	0.81	0.83
	CON (159)	0.85	0.13	0.85	0.84
	CAR (147)	0.85	0.14	0.85	0.85
	Relief-F (159)	0.85	0.15	0.86	0.85
	SVD (127)	0.73	0.23	0.73	0.77
	MOGA (97)	0.88	0.12	0.87	0.88
	FSBR (127)	0.87	0.13	0.88	0.88
	DRED (130)	**0.87**	**0.12**	**0.88**	**0.88**

From the table 4.1 and 4.2, it is seen that the performance of DRED is better than the other static attribute reduction techniques in most of the cases and also the method is statistically significant.

- **Comparison of DRED Algorithm with Popular Incremental Algorithms:**

Proposed DRED algorithm is compared with incremental algorithms mentioned in section 4.1.

Table 4.3 shows the performance comparison between proposed DRED and considered incremental feature selection methods with respect to the computational time and number of selected features where R represents number of reduct, and T represents execution time for different algorithms in seconds.

Table 4.3: Performance analysis of DRED and incremental feature selection methods

Dataset/ Attributes	IUAARI [50]		IUAARS [52]		Xu et al. [230]		GIARC-L [214]		Shu et al. [231]		DRED	
	R	**T**	**R**	**T**	**R**	**T**	**R**	**T**	**R**		**R**	**T**
Wine/13	7	0.23	**6**	**0.02**	10	0.31	**6**	0.08	7	0.04	7	0.09
Heart/13	**8**	0.30	**8**	0.09	9	0.35	**8**	**0.02**	9	0.09	10	0.03
Glass/9	8	0.11	8	**0.01**	9	20.87	**7**	**0.01**	8	5.43	8	0.10
Zoo/16	6	0.06	6	**0.04**	6	0.41	**5**	0.06	**5**	0.12	**5**	0.06
Dermatology/33	11	0.50	9	0.25	11	7.34	10	0.20	11	9.03	8	0.19
Mushroom/21	5	92.78	**4**	**34.56**	**4**	120.98	5	35.78	5	99.03	**4**	45.38
Coil20/1024	201	603.12	201	**587.76**	265	1520.0	201	588.76	199	1289.67	**166**	1140.40
Orl/1024	222	708.95	232	597.98	278	1673.21	222	679.89	**205**	1230.98	212	828.98
Allaml/7129	245	**1759.98**	187	1604.32	295	2345.21	173	1346.87	198	2321.23	**169**	1797.78
Leukemia/7070	197	1529.67	187	1323.87	243	2456.78	167	**1220.34**	**123**	2301.65	130	1887.65

From Table 4.3, it is seen that the computation time needed for the proposed method is less for many cases and greater for few datasets but at the same time the amount of reduction is much more compared to the other algorithms. Also, the proposed incremental DRED algorithm is compared with the static FSBR algorithm [47] which is in essence the static version of DRED algorithm and the results are given in the Table 4.4. Table 4.4 shows that the computational time needed for the DRED method is less than the FSBR method [47] where the objective of the method is met. DRED method also provides greater classification accuracies more than 50% cases.

Table 4.4: Performance comparison of DRED and FSBR methods

Dataset	#Selected features		Average Accuracy (%)		Execution time(sec)	
	FSBR	**DRED**	**FSBR**	**DRED**	**FSBR**	**DRED**
Wine	**6**	7	97.18	**97.53**	7.01	**0.20**
Heart	**9**	10	**84.67**	82.28	5.24	**0.12**
Glass	**6**	8	**73.97**	69.60	3.79	**0.10**
Zoo	8	**5**	**95.34**	94.47	2.01	**0.07**
Dermatology	11	**8**	**99.00**	98.24	35.39	**0.52**
Mushroom	5	**4**	**98.87**	97.83	300.01	**50.38**

Dataset	#Selected features		Average Accuracy (%)		Execution time(sec)	
	FSBR	**DRED**	**FSBR**	**DRED**	**FSBR**	**DRED**
Coil20	**156**	166	83.78	**83.98**	1200.02	**1140.40**
Orl	**157**	212	60.90	**61.32**	930.05	**828.98**
Allam	**167**	169	84.15	**84.49**	2013.87	**1797.78**
Leukemia	**127**	130	86.64	**86.94**	3467.98	**1887.65**

c Incremental Feature Selection Using Rough Set Theory and Genetic Algorithm (IFS):

This section describes a new method [57] of incremental feature selection using the concepts of Rough Set Theory [17-20] and Genetic Algorithm [23, 102, 103] (IFS). Here the incremental feature selection technique is used in dynamic environment where newly generated group of data, together with the knowledge extracted from the previous data are analyzed to select the most relevant features of the entire dataset. As a result, efficiency and acceptability of the system increases. Proposed IFS method selects the optimized and relevant features called reduct. The novelty of the proposed algorithm is that it can select features both in static and dynamic environment and no prior statistical information of the data is required. Algorithm is dependent only on few controlling parameters and it has an ability of lifelong learning to deal with the new data and update the organization of the system incrementally. The IFS method proposes a single objective genetic algorithm by combining multiple criteria for obtaining single optimal solution which effectively reduces dimensionality of the dataset without sacrificing classification accuracy. The overall flow diagram of the IFS method [57] is shown in Figure 4.1.

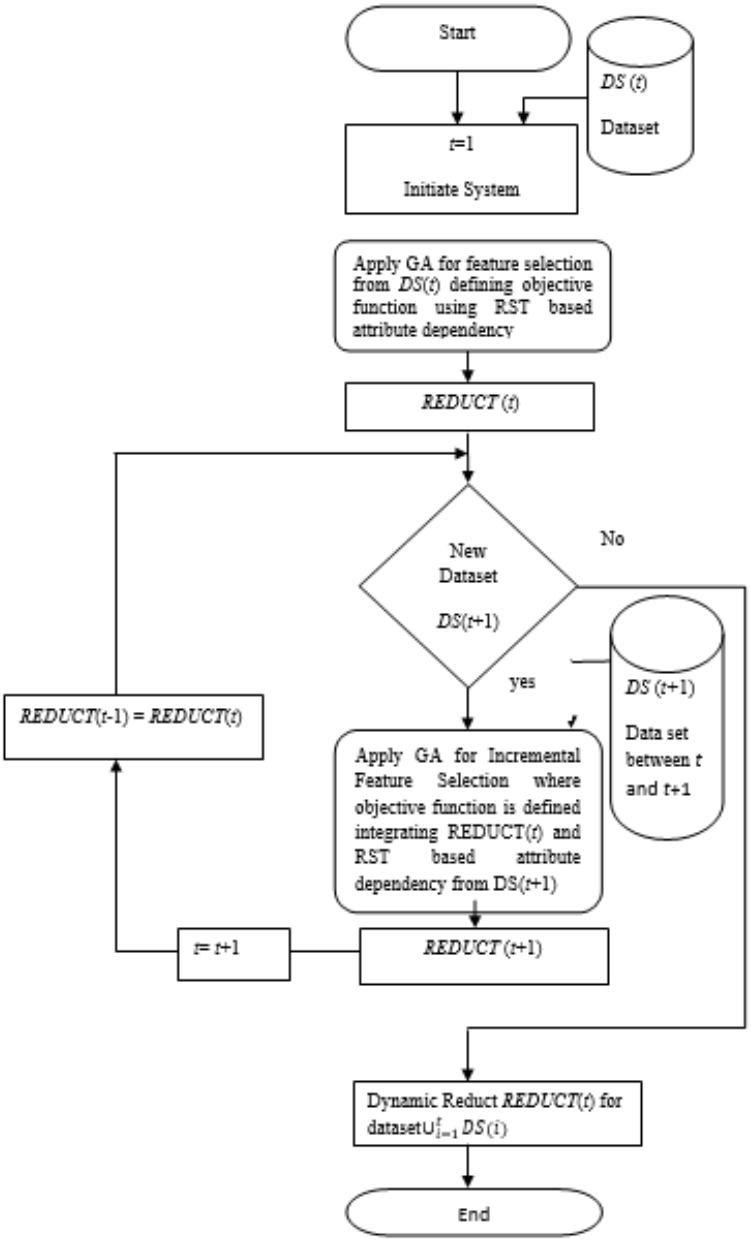


Figure 4.1: Flow diagram of IFS

Figure 4.1 shows the sequence of steps in which the dynamic reduct is developed and evaluated in the time interval *(t, t+1)*.

The dataset exists at time t is considered as old data and the GA based reduct generation method is applied on this dataset which provides initial reduct *REDUCT*(t) for the dataset at time t. Now after *t+1*-time new data called incremental data are stored as new dataset arrives in time interval *(t, t+1)*.

Now in the proposed IFS technique, initial reduct *REDUCT(t)* and the new dataset are feed in the training process and the modified reduct is generated for the whole dataset available at time *t+1*. At next instant of time, the new reduct *REDUCT(t+1) at t+1* time is considered as old reduct and new group of data is taken care for constructing further modified incremental reduct. This process is continued after every interval of time when a group of data enters into the system. Thus, a dynamic reduct is generated for incremental data set efficiently.

- **Incremental Feature Selection:**

In the work, a novel incremental feature selection technique is proposed for incremental datasets to select the reduct dynamically using the concept of Rough Set Theory (RST) and Genetic Algorithm (GA). Here, GA is applied not on the whole dataset but only on new chunk of objects currently enters into the system. Thus, the major issue of time complexity for running GA on high voluminous dataset is tackled in the proposed method as it is invoked in regular basis after small to moderate size of data subset is generated. Genetic algorithm-based reduct generation, using a single criterion, may not always yield better results due to the varied characteristics of the datasets used. If multiple criteria are combined for attribute selection, an algorithm generally provides more important attributes compared to the algorithm relying on a single criterion.

In the method, to apply the concept of dynamic data, the original decision system *DS = (U, A, D)* is divided into two subsystems namely *S1 = (U1, A, D)* and *S2 = (U2, A, D)* as old and new subsystems respectively. When the IFS algorithm is first run for the initial subsystem *S1*, no previous reduct information is available; thus, the fitness function is defined based only on the attribute dependency and the reduct *R1* is formed by the algorithm. Subsequently, when newly arrived decision subsystem *S2 = (U2, A, D)* is become available then the previous reduct *R1* with the new subsystems determines the reduct R of the whole system $S = S1 \cup S2$.

I. Initial Population:

As GA is a population based stochastic search algorithm, so the initial population is created firstly at random from the new decision subsystem *S2*. Binary string representation of chromosome is opted in this work.

Each chromosome consists of only two values, '1' and 0 which implies that the index feature is present and absent respectively in current subset. The method generates a population of size *M* randomly, where the length of each binary chromosome is $|A| = N$, *where N =* number of conditional attributes present in *S2*. *Let A = {B1, B2, B3, …, BN},* and the *i*-th bit of a chromosome ch corresponds to attribute *Bi*. All '1's' in chromosome ch correspond to an attribute subset, i.e., $A1 \cup A$.

II. Fitness Function:

The fitness function determines the quality of a solution in the population; thus, a strong fitness function is imperative for obtaining good results.

Here, a heuristic is applied, which has two parts, to define the fitness function. The first part is the positive region overlapping value of the attribute subset A_1 with the whole attribute set A in S_2, and the second part is the extent to which A_1 matches with the pre-computed reduct R_1 of already available data in S_1. Thus, the fitness function of GA [23, 102, 103] is defined using positive region overlap computed by the Equation (4.2) for new group of added data and the reduct obtained from the old existing data discussed below.

i. Positive Region Overlap:

Before going in detail discussion on positive region overlap, first of all concepts of positive region is described. Let, *DS = (U, A, D)*, where *A* = set of conditional attributes, *D* = decision attribute and *U* is the set of objects.

To Compute the positive region of an attribute subset *P*, *U* is partitioned into equivalence classes $[x]_P$ using the indiscernibility relation defined in Equation (2.7).

Equivalence classes $[x]_D$ are also formed using the Equation (2.7). Two different partitions U/P and U/D of equivalence classes $[x]_P$ and $[x]_D$ are formed. Now each class $[x]_D$ in U/D is considered as the target set *X*. Computation of lower approximation $\underline{P}X$ of target set *X* under *P* can be done using the Equation (2.8) for all $X \in U/D$. Positive region $POS_P(D)$ is obtained by taking the union of the lower approximations $\underline{P}X$ under *P* for all *X* in U/D using Equation (4.1).

$$POS_P(D) = \bigcup_{X \in U/D} \underline{P}X \qquad (4.1)$$

Now positive region overlap is achieved by taking the intersection of the Positive region $POS_A(D)$ of the decision system *DS* with the attribute set *A* and the Positive Region $POS_{A'}(D)$ of *DS* with the attribute subset *A'*, where $A' \subseteq A$. Then it is called the Positive Region Overlap for *A'* and referred as $P_{A'}(D)$ and defined by the Equation (4.2).

$$P_{A'}(D) = POS_A(D) \cap POS_{A'}(D) \qquad (4.2)$$

$P_{A'}(D)$ is always less than or equal to $POS_A(D)$, $P_{A'}(D) \leq POS_A(D)$, because the objects cannot be made more discernible by reducing the attributes. Reduction of attributes only renders the objects more indiscernible.

The value of Positive Region Overlap, $P_{A'}(D)$, signifies to what extent the objects of the universe are discernible based on the attribute subset *A*'.

Hence, a reduct should have the highest possible value of *Positive region overlap* with the minimum number of attributes.

Now in the IFS method, the positive regions of S_2 with the attribute set *A* and attribute subset A_1 are $POS_A(D)$ and $POS_{A_1}(D)$, respectively, computed using Equation (4.1). Overlapping positive region $P_A(D) = POS_A(D) \cap POS_{A_1}(D)$, computed using Equation (4.2) indicates that the number of objects in S_2 is correctly classified by both *A* and A_1.

ii. Similarity Measure:

For the second part of the fitness function, attributes common to subset A_1 and precomputed reduct R_1 determined. Because the reduct R_1 is already formed on the subsystem S_1, it is desirable that more attributes in R_1 will be in the subsequent reduct when new subsystem S_2 is added to the system. Thus, the rate at which the number of attributes of A_1 is common to R_1, is also considered as other criteria in the fitness function.

Therefore, the first part of the fitness function is derived from the newly arrived subsystem S_2, and the second part indicates how much the new subsystem agrees with the reduct of the old existing subsystem. Thus, without the direct involvement of the subsystem S_1, only the already generated reduct R_1 is used, together with subsystem S_2 for the formation of the subsequent reduct in the incremental data set. Here, the relative importance of two parts of the fitness function is considered by introducing a weight factor 'w' because both these parts need to be maximized to obtain the reduct. Thus, the fitness function $f(ch)$ for a chromosome ch is defined in equation (4.3), which helps us to find the optimal feature subset of the system S.

$$f(ch) = w\left(\left(\frac{P_A(D)}{|U_2|}\right)^k \Big/ |A_1|^z\right) + (1-w)\frac{|R_1 \cap A_1|}{|A|} \tag{4.3}$$

In equation (4.3), the power 'k' is used to maximize the contribution of the positive region value in the fitness function. The power 'z' is kept very low, to give the reducts having lesser attributes, a higher score. The algorithm has been tested for different values of these controlling parameters on the test data sets, and based on experimental results, the values of the parameters have been fixed as follows $k = 2$, $z = 0.05$, and $w = 0.55$.

iii. Genetic Operations:

As the convergence of genetic algorithm depends on the proper selection of parameters, so selection of parameters value is an important task here. Selection is the first genetic operator applied on the population. Here, ranks based roulette-wheel selection method [23] is used and it is continued until the mating pool is filled up.

The population may lose the best chromosome by crossover and mutation. So, elitism operation includes 5% of chromosomes with the optimum fitness values into the mating pool. Crossover operator is applied to the mating pool hoping that it would create a better string.

Crossover operator used is uniform crossover with probability 0.9. New chromosomes generated by crossover are taken into the next generation population only if their fitness is better than that of their parent chromosomes. Mutation is also used to maintain diversity in the population. Mutation operation involves flipping of a bit in a chromosome, changing 0 to 1 and 1 to 0.

It is done in a random bit of each chromosome with probability 0.001. All parameter values are determined experimentally, and the algorithm terminates when the best fitness value of the chromosome has no improvement over a specified number of generations. The detail algorithm is given below.

Algorithm: IFS (S_1, S_2, R)

Input: Reduct R_1 of decision subsystem S_1 = (U_1, A, D) and newly arrived decision subsystem S_2 = (U_2, A, D)

Output: Reduct R of $S = S_1 \cup S_2$

Step I: Compute the positive region $POS_A(D)$ of S_2 with respect to attribute set A.

Step II: Initialize the population P of GA with size = M.

Step III: Set chromosome length as N, number of attributes in A.

Step IV: Calculate fitness value for each chromosome *ch* in P using equation (4.3).

Step V: Use Roulette Wheel Selection to select the best chromosomes into the mating pool based on their fitness values.

Step VI: Apply uniform crossover and mutation operations on chromosomes in the mating pool with crossover probability of 0.9 and mutation rate of 0.001.

Step VII: Choose the chromosomes for the next generation with 50% replacement of the parent population.

Step VIII: Repeat Step IV to step VII until the GA converges.

Step IX: The best chromosome of the final population forms the reduct R of the entire system S.

- **Space and Time Complexity of the IFS Algorithm:**

This algorithm requires storing only the newly arrived decision subsystem S_2 = (U_2, A, D). Hence, it has a space complexity of O ($|U_2|*(|A| + 1)$), where $|U_2|$ is the number of objects in S_2 and $|A|$ represents the number of conditional attributes while 1 is added for the decision class.

The time complexity is calculated according to the following steps:

i. Calculation of the Equivalent Set has a time complexity of O ($|U_2|^2$).
ii. The complexity of the Fitness Function, f (*ch*), also requires calculation of the equivalent set. Thus, its time complexity is O ($|U_2|^2$).

iii. Therefore, the running time of the GA is $O(|M| * t * |U_2|^2)$, where $|M|$ is the population size of the GA, and t represents the number of generations requires to GA converge.
iv. So, the total time complexity is $O(|U_2|^2 + |M|* t * |U_2|^2)$, which is $O(|M| * t * |U_2|^2)$.

As $|M|$ and t are constants, time complexity is $O(|U_2|^2)$, a polynomial of degree two. Since, the algorithm runs in regular interval of time, so the size $|U_2|$ of added group of data is not very large.

As a consequence, the algorithm selects the optimal feature subset in incremental dataset efficiently.

- **Experimental Results of the IFS method:**

The algorithm has been extensively tested on experimental datasets summarized in section 2.2. Experiments are done on a computer with the specification as Computer Model: ACER emachines D725; CPU:

Pentium(R) Dual-Core CPU T4400 @ 2.20GHz × 2; Memory: 4GB; OS: Ubuntu 12.04 LTS - 32 bit. Java programming language is used for the implementation of the work.

To judge the performance of the IFS algorithm, a series of experiments is conducted, and comparative analyses are made. At first, all the attributes are discretized by ChiMerge [219] discretization algorithm. Here, randomly selected 80% data of each dataset is considered as the existing data, and the rest 20% data is used as the new group of data. The proposed IFS method is compared with standard static attribute reduction techniques and some popular incremental attribute reduction techniques. The main focus of the experiments is on the three issues: number of features, classification accuracy and execution efficiency.

- **Comparison With Static Attribute Reduction Techniques:**

To judge the effectiveness and the efficiency of the proposed IFS method, the method is compared with common standard static or non-incremental feature selection methods. The static feature selection methods CFS [95], CON [213] and Relief-F [215] are available at "Weka" tool [218] while CAR [214], SVD [216] and MOGA [217] are implemented.

Results of the existing static feature selection methods and the proposed IFS method in static (Static Approach) and incremental mode are evaluated and compared on the basis of classification accuracies on whole dataset and reduced datasets of the state-of-the-art classifiers available in weka tool [218].

In the work, considered classifiers are Naïve Bayes (NB) [7], Support vector machine (SVM) [6], K-nearest neighbors K-NN [37], Bagging [191], Tree based classifier (J48) [5], and Multilayer Perceptron (MLP) [3]. SVM is used with RBF Kernel; *K* value of K-NN is set to the square root of sample size of data. Original number of attributes, number of attributes after applying proposed IFS and existing static feature selection methods and the accuracies (%) of the reduced datasets by the mentioned classifiers are computed and listed in Table 4.5.

Table 4.5: Performance comparison of IFS and static feature selection methods

Dataset (#Original features)	Methods (#features)	Claifiers					
		NB	**SVM**	**Bagging**	**J48**	**MLP**	**IPSO**
Wine(13)	CFS (8)	96.19	96.96	96.45	94.94	93.82	93.10
	CON (8)	96.19	97.11	96.63	94.94	94.94	94.30
	CAR (8)	96.19	96.21	96.45	94.74	93.82	93.10
	Relief-F (9)	96.69	96.61	96.63	94.94	94.97	94.40
	SVD (6)	96.45	96.46	96.56	95.78	95.35	95.78
	MOGA (7)	97.17	96.65	95.56	96.64	95.78	95.87
	Static Approach (7)	98.65	97.45	97.01	96.36	96.61	96.50
	IFS (6)	**98.91**	**100**	**97.32**	**96.76**	**96.62**	**97.19**
Heart (13)	CFS (8)	84.36	84.75	81.67	81.11	81.11	81.67
	CON (11)	84.50	84.44	82.07	81.48	82.89	79.55
	CAR (10)	83.36	84.75	81.67	83.11	82.11	80.67
	Relief-F (10)	83.50	84.44	82.07	81.48	83.89	79.59
	SVD (9)	83.55	83.77	**83.99**	82.31	83.61	82.28
	MOGA (8)	84.67	83.32	83.22	83.56	83.87	83.26
	Static Approach (9)	85.72	85.12	83.94	**84.58**	**84.90**	83.49
	IFS (9)	**85.89**	**85.86**	83.78	84.43	84.87	**84.32**
Glass (9)	CFS (6)	43.92	57.94	79.91	73.83	68.69	70.09
	CON (7)	47.20	57.48	78.50	71.50	64.20	68.60
	CAR (8)	56.92	58.94	80.91	75.83	69.69	71.09
	Relief-F (8)	57.20	57.48	79.50	70.50	63.20	72.60
	SVD (8)	57.79	58.75	76.39	72.56	67.70	75.45
	MOGA (7)	56.54	57.76	76.49	72.45	64.76	70.89
	Static Approach (6)	65.73	63.94	83.97	78.53	72.60	79.00
	IFS (8)	**69.73**	**66.34**	**84.34**	**79.93**	**76.30**	**79.32**
Zoo (16)	CFS (9)	96.03	93.06	94.05	94.04	93.06	93.06
	CON (9)	96.03	93.03	94.05	94.04	93.88	94.32
	CAR (6)	94.05	93.92	93.32	94.02	94.07	94.05
	Relief-F (7)	95.03	93.70	93.01	93.01	94.12	94.12
	SVD (5)	96.19	94.77	94.88	94.67	94.42	94.77
	MOGA (6)	94.78	94.23	94.45	95.21	94.32	94.34
	Static Approach (8)	97.14	95.55	95.75	94.06	96.03	94.07
	IFS (5)	**97.43**	**97.12**	**97.05**	**95.76**	**97.02**	**98.01**

Dataset (#Original features)	**Methods (#features)**	**Clasifiers**					
		NB	**SVM**	**Bagging**	**J48**	**MLP**	**IPSO**
Dermatology (33)	CFS (9)	98.76	97.42	97.01	98.06	98.07	98.62
	CON (9)	98.52	98.25	95.56	98.06	98.86	98.67
	CAR (11)	98.73	98.30	97.42	98.31	98.06	98.07
	Relief-F (11)	98.72	98.45	95.56	97.16	98.76	98.46
	SVD (9)	97.88	98.13	96.85	97.29	98.31	98.89
	MOGA (9)	97.87	97.89	97.90	97.05	98.67	97.90
	Static Approach (9)	99.01	99.01	98.77	**99.02**	**99.32**	99.30
	IFS (9)	**99.83**	**99.02**	**98.95**	98.98	98.02	**99.41**
Mushroom (21)	CFS (4)	97.52	96.01	96.52	97.01	97.01	97.01
	CON (5)	98.52	98.85	98.52	99.05	98.16	**99.86**
	CAR (8)	98.02	98.32	99.02	99.65	**99.23**	99.01
	Relief-F (5)	97.04	98.03	98.03	98.13	98.10	98.10
	SVD (3)	97.14	97.33	97.13	97.93	97.24	87.74
	MOGA (5)	97.34	95.45	96.67	96.34	96.34	96.45
	Static Approach (5)	**99.25**	99.12	**99.24**	**99.88**	98.35	98.28
	IFS (3)	**99.76**	**99.54**	**99.76**	98.86	**98.78**	**98.46**
Coil20 (1024)	CFS (194)	78.12	79.24	80.01	80.10	79.25	79.85
	CON (194)	79.60	79.20	78.00	79.35	79.34	78.79
	CAR (201)	77.12	78.24	77.01	80.10	75.25	78.85
	Relief-F (198)	77.60	78.20	77.80	78.35	77.34	78.49
	SVD (157)	78.64	78.41	78.96	76.98	76.84	78.99
	MOGA (95)	82.20	79.99	82.92	83.02	83.02	84.20
	Static Approach (172)	**84.97**	83.76	84.45	**84.76**	83.43	84.94
	IFS (157)	**84.97**	**83.96**	**84.55**	**84.76**	**83.52**	**85.34**
Orl (1024)	CFS (201)	55.12	55.24	53.01	51.10	53.80	51.35
	CON (198)	56.60	53.20	52.00	51.35	50.70	52.89
	CAR (204)	55.12	55.24	53.01	51.10	53.80	51.35
	Relief-F (213)	54.60	54.20	54.01	54.25	53.79	53.89
	SVD (109)	54.12	53.94	57.01	54.10	53.80	54.95
	MOGA (110)	59.60	57.20	59.00	58.25	59.70	59.09
	Static Approach (134)	61.55	60.95	61.83	**60.12**	60.33	60.03
	IFS (109)	**62.95**	**61.95**	**62.33**	**60.12**	**61.32**	**61.53**
	CFS (221)	81.12	81.32	82.62	83.82	82.21	83.01
	CON (230)	79.12	80.32	81.62	82.82	80.21	82.01

Dataset (#Original features)	Methods (#features)	Claifiers					
		NB	**SVM**	**Bagging**	**J48**	**MLP**	**IPSO**
Allaml (7129)	CAR (201)	80.02	80.32	81.52	82.72	81.21	82.02
	Relief-F (210)	81.12	81.32	82.62	83.82	82.21	83.11
	SVD (189)	80.22	81.62	81.27	82.92	82.81	83.19
	MOGA (194)	81.12	83.32	84.72	81.82	82.01	82.21
	Static Approach (157)	**83.07**	84.05	85.99	84.94	83.15	84.14
	IFS (139)	82.87	**84.65**	**86.89**	**85.54**	**83.65**	**85.24**
Leukemia (7070)	CFS (147)	83.23	84.23	82.67	82.02	82.05	83.23
	CON (159)	84.05	85.87	84.23	84.12	84.21	84.67
	CAR (147)	83.47	85.23	84.56	84.53	84.54	84.56
	Relief-F (159)	84.23	85.34	84.12	84.34	84.34	84.98
	SVD (88)	73.68	76.32	71.05	71.02	71.05	73.68
	MOGA (97)	86.34	85.90	85.50	85.78	87.12	86.23
	Static Approach (147)	**87.61**	86.18	87.02	86.05	86.34	86.33
	IFS (88)	87.51	**87.32**	**87.32**	**87.55**	**87.44**	**87.03**

From Table 4.5, it is observed that the proposed IFS method selects fewer attributes with higher classification accuracy compared to the other methods in most of the cases. It is worth noting to mention that all the static methods are run on the whole reduced dataset only once, whereas IFS is run dynamically on the dataset in an incremental way. This implies that, the proposed incremental feature selection method selects the most important attributes without losing too much information.

- **Comparison Of IFS Algorithm with Popular Incremental Algorithms:**

To evaluate the performance of IFS method, the method is compared with popular incremental algorithms such as IUAARI [50], IUAARS [52], Xu et al. [230], GIARC-L based on complementary entropy [214] and Shu et al. [231] mainly based on the criteria such as, size of selected feature subset and the computational time. These incremental methods used for the comparison purpose are discussed in the section 4.1. Here, randomly selected 80% data of each dataset is considered as the original decision table, and the rest 20% data is used as the new group of data. The whole dataset is used as the new decision table. First, the method RIDAS [232] is used to generate the reduced attribute set for each original decision table. Secondly, based on previous results, we respectively use IUAARI [50], IUAARS [52], Xu et al. [230], GIARC-L based on complementary entropy [214] and Shu et al. [231] are used to generate the reduced attribute set for each of the new decision table. Finally, to demonstrate the practicability and strength of the proposed method IFS (proposed algorithm) is applied to find out the reduced attribute set for each of the new decision tables, and the results are compared with the existing algorithms. The experimental results are shown in Table 4.6. Here, T is the running time of an algorithm measured in second and R is the number of attributes selected by the algorithms.

Table 4.6: Performance comparison of IFS and incremental feature selection methods Tebl Tyaping

Dataset/Features	RIDAS [232]		IUAARI [50]		IUAARS [52]		Xu et al. [230]		GIARC-L [214]		Shu et al. [231]		IFS	
	R	**T**	**R**	**T**	**R**	**T**	**R**	**T**	**R**	**T**	**R**	**T**	**R**	**T**
Wine/13	7	0.34	7	0.23	**6**	0.02	10	0.31	**6**	0.08	7	0.04	**6**	**0.01**
Breast Cancer/9	6	0.56	6	0.50	6	0.01	9	29.15	4	0.06	5	7.01	**3**	1.03
Glass/9	**7**	0.19	8	0.11	8	0.01	9	20.87	**7**	0.01	8	5.43	8	**0.01**
Car/6	**5**	2.67	6	0.06	6	0.04	6	12.34	**5**	0.06	6	20.01	**5**	**0.01**
Dermatology/33	10	4.32	11	0.50	**9**	0.25	11	7.34	10	0.20	11	9.03	**9**	**0.19**
Mushroom/21	4	96.45	5	92.78	4	**84.56**	4	120.98	5	90.78	5	99.03	**3**	92.38
Coil20/1024	194	1003.67	201	603.12	201	**587.76**	265	1520.0	201	588.76	199	1289.67	**157**	1220.40
Orl/1024	176	1087.32	222	708.95	232	**597.98**	278	1673.21	222	679.89	205	1230.98	**109**	1054.78
Allaml/7129	205	2010.23	245	1759.98	187	1604.32	295	2345.21	173	**1346.87**	198	2321.23	**139**	2197.78
Leukemia/7070	158	2308.97	197	1529.67	187	1323.87	243	2456.78	167	**1220.34**	123	2301.65	**88**	2134.65

From Table 4.6, it is seen that although the computation time needed is bit more for IFS method for few datasets, the amount of reduction is much more compared to the other algorithms.

The objective of this method is to produce maximum reduction at the cost of computation time (this situation will rarely occur, as the method will run in a regular interval of time while the added group of data is small to moderate in size), such that the time required by the classifiers is greatly reduced.

From the Table 4.5 and Table 4.6 it is seen that IFS method selects the feature subset with lesser cardinality and gives the highest classification accuracies compare to the standard static and incremental feature selection algorithm for most of the cases.

Computational time to execute IFS is also comparable with other algorithms.

- **Statistical assessment of IFS:**

Not only the feature reduction and classification accuracies are in favor of the effectiveness of IFS method, but the statistical analysis is also performed on the data set using Wilcoxon's rank sum test [39] to demonstrate that the method is also statistically significant with respect to the existing static and incremental feature selection methods. In the experiment, this test is made, as it is valid for data of any distribution and much less sensitive to outliers compare to other testing methods. Here similar to the DRED method the statistical analysis is done using Wilcoxon's rank sum test [39] and the result is shown in Table 4.7. A bold-faced font is used to write its mean. From Table 4.7, considering all given datasets and all classifiers, experimentally it is observed that proposed IFS method is statistically significant which express the effectiveness of the method. Thus, the method is very effective with all respects of dimension reduction, classification, efficiency, and statistical analysis, which demonstrate the importance of the method.

Table 4.7. Performance comparison of IFS with related existing methods

Dataset (#Original features)	**Methods (#features)**	**Classifiers Accuracy (%)**					
		NB	**SVM**	**Bagging**	**NB**	**MLP**	**IPSO**
Wine(13)	CFS (8)	96.19†	96.96†	96.45†	94.94†	93.82†	93.10†
	CON (8)	96.19†	97.11†	96.63†	94.94†	94.94†	94.30†
	CAR (8)	96.19†	96.21†	96.45†	94.74†	93.82†	93.10†
	Relief-F (9)	96.69†	96.61†	96.63†	94.94†	94.97†	94.40†
	RIDAS (7)	95.94†	96.17†	96.63†	95.44†	95.97†	95.40†
	IUAARI (7)	96.89†	97.01†	96.83†	96.34†	96.77†	96.60†
	IUAARS (6)	96.99†	97.31†	96.73†	95.64†	**97.77**†	96.65†
	Xu et al. (10)	97.99†	97.61†	96.93†	96.54†	97.70≈	97.40≈
	GIARC-L (6)	98.78≈	98.89†	98.45≈	**98.45**	97.67≈	**97.65**
	Shu et al. (7)	98.12≈	98.91†	98.23≈	97.60†	96.77†	97.60≈
	IFS (6)	**98.91**	**100**	**98.93**	96.76†	96.62†	97.19≈
Breast Cancer (9)	CFS (4)	95.71≈	94.42†	94.56†	94.85†	94.56†	94.13†
	CON (5)	**95.99**	95.27≈	94.56†	94.70†	93.56†	94.56†
	CAR (5)	**95.99**	94.34†	94.56†	94.12†	93.45†	94.21†
	Relief-F (5)	95.93≈	94.34†	94.56†	94.82†	93.75†	94.27†
	RIDAS (6)	95.73≈	95.34≈	94.96†	95.02≈	94.75†	95.07≈
	IUAARI (6)	95.80≈	95.82≈	95.06≈	95.32≈	95.25≈	95.17≈
	IUAARS (6)	95.77≈	95.54≈	95.26≈	95.12≈	**95.73**	95.15≈
	Xu et al. (9)	95.70≈	95.84≈	95.22≈	95.52≈	95.37≈	95.38≈
	GIARC-L (4)	95.98≈	**95.94**	95.15≈	95.42≈	95.47≈	95.28≈
	Shu et al. (5)	95.76≈	95.64≈	95.06≈	**95.82**	95.27≈	95.08≈

Dataset (#Original features)	**Methods (#features)**	**Classifiers Accuracy (%)**					
		NB	**SVM**	**Bagging**	**NB**	**MLP**	**IPSO**
	IFS (3)	95.71≈	94.70†	**95.27**	94.42†	93.99†	**95.42**
Glass (9)	CFS (6)	66.82†	81.77†	80.86†	**76.63**	**77.57**	75.70≈
	CON (7)	66.82†	83.18≈	82.86†	74.76†	76.17†	70.09†
	CAR (8)	66.86†	83.24≈	83.21†	74.87†	77.20≈	70.23†
	Relief-F (8)	66.80†	82.34†	83.21†	74.87†	77.20≈	70.23†
	RIDAS (7)	66.22†	83.08≈	82.76†	74.46†	76.27†	70.29†
	IUAARI (8)	66.12†	82.24†	83.13†	74.87†	77.10≈	73.13†
	IUAARS (8)	66.08†	82.43†	83.19†	74.27†	76.20†	73.30†
	Xu et al. (9)	66.52†	83.58≈	82.96†	74.48†	76.59†	73.55†
	GIARC-L (7)	66.47†	82.91†	83.80†	74.73†	76.57†	74.93†
	Shu et al. (8)	66.57†	81.97†	83.96†	74.93†	75.57†	74.90†
	IFS (8)	**67.57**	**83.77**	**84.86**	75.23	**77.57**	**75.93**
Car (6)	CFS (1)	55.55†	70.02†	**70.02**	70.02†	70.02†	70.02†
	CON (5)	82.52≈	93.57≈	**70.02**	95.13≈	95.25†	94.50†
	CAR (5)	82.12≈	93.32≈	69.12†	95.14≈	94.45†	94.35†
	Relief-F (5)	82.46≈	93.57≈	70.00≈	**95.42**	**96.52**	**96.35**
	RIDAS (5)	**82.56**	93.21≈	70.00≈	95.17≈	96.21≈	96.05≈
	IUAARI (6)	82.02≈	92.57†	70.01≈	95.15≈	95.15†	94.01†
	IUAARS (6)	82.11≈	93.38≈	**70.02**	95.31≈	95.15†	94.32†
	Xu et al. (6)	82.32≈	93.37≈	**70.02**	95.01≈	95.05†	94.21†
	GIARC-L (5)	82.22≈	93.42≈	70.01≈	95.13≈	95.15†	94.13†
	Shu et al. (6)	82.02≈	93.30≈	69.99†	95.10≈	95.17†	94.20†
	IFS (5)	82.52≈	**93.57**	**70.02**	95.13≈	95.19†	94.50†
Dermatology (33)	CFS (9)	98.76†	97.42†	97.01†	98.06≈	98.07≈	98.62†
	CON (9)	98.52†	98.25†	95.56†	98.06≈	98.86≈	98.67†
	CAR (11)	98.73†	98.30†	97.42†	98.31≈	98.06≈	98.07†
	Relief-F (11)	98.72†	98.45†	95.56†	97.16†	**98.76**	98.46†
	RIDAS (10)	98.32†	98.24†	97.36†	98.16≈	98.06≈	98.27†
	IUAARI (11)	98.23†	98.50†	97.92†	98.01≈	97.86†	97.97†
	IUAARS (9)	98.21†	98.40†	97.29†	98.01≈	97.26†	97.88†
	Xu et al. (11)	97.33†	97.10†	97.45†	98.11≈	97.86†	98.12†
	GIARC-L (10)	97.73†	97.30†	97.72†	98.01≈	97.06†	97.27†
	Shu et al. (11)	98.73†	98.30†	98.42≈	98.31≈	98.06≈	98.87†
	IFS (9)	**99.83**	**99.02**	**98.95**	**98.98**	98.02≈	**99.41**

Dataset (#Original features)	**Methods (#features)**	**Classifiers Accuracy (%)**					
		NB	**SVM**	**Bagging**	**NB**	**MLP**	**IPSO**
Mushroom (21)	CFS (4)	97.52†	96.01†	96.52†	97.01†	97.01†	97.01†
	CON (5)	98.52†	98.85†	98.52†	99.05≈	98.16†	**99.86**
	CAR (8)	98.02†	98.32†	99.02≈	**99.65**	**99.23**	99.01≈
	Relief-F (5)	97.04†	98.03†	98.03†	98.13†	98.10†	98.10†
	RIDAS (4)	98.29†	98.36†	98.56†	98.79†	98.32†	98.10†
	IUAARI (5)	98.89†	98.76†	98.86†	98.89†	98.88†	98.16†
	IUAARS (4)	98.79†	98.55†	98.66†	98.49†	98.18†	98.26†
	Xu et al. (4)	97.79†	97.55†	97.66†	97.89†	97.88†	97.16†
	GIARC-L (5)	98.76†	98.54†	98.76†	98.06†	98.08†	98.36†
	Shu et al. (5)	99.17†	99.21≈	98.33†	98.24†	98.37†	98.61†
	IFS (3)	**99.76**	**99.54**	**99.76**	98.86†	98.78†	98.46†
Coil20 (1024)	CFS (194)	78.12†	79.24†	80.01†	80.10†	79.25†	79.85†
	CON (194)	79.60†	79.20†	78.00†	79.35†	79.34†	78.79†
	CAR (201)	77.12†	78.24†	77.01†	80.10†	75.25†	78.85†
	Relief-F (198)	77.60†	78.20†	77.80†	78.35†	77.34†	78.49†
	RIDAS (194)	77.40†	78.10†	78.23†	78.37†	78.49†	78.89†
	IUAARI (201)	79.24†	79.42†	78.98†	81.10†	81.25†	81.75†
	IUAARS (201)	78.12†	78.24†	80.01†	79.10†	79.25†	80.85†
	Xu et al. (265)	82.16†	82.09†	81.90†	82.05†	81.39†	82.41†
	GIARC-L (201)	80.12†	81.24†	80.32†	81.30†	81.95†	82.45†
	Shu et al. (199)	81.60†	81.90†	81.70†	82.35†	82.30†	81.42†
	IFS (157)	**84.97**	**83.96**	**84.55**	**84.76**	**83.52**	**85.34**
Orl (1024)	CFS (201)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†
	CON (198)	56.60†	53.20†	52.00†	51.35†	50.70†	52.89†
	CAR (204)	55.12†	55.24†	53.01†	51.10†	53.80†	51.35†
	Relief-F (213)	54.60†	54.20†	54.01†	54.25†	53.79†	53.89†
	RIDAS (176)	58.01†	58.78†	57.09†	57.98†	58.78†	58.81†
	IUAARI (222)	59.09†	59.70†	59.23†	59.92	59.23†	59.41†
	IUAARS (232)	58.97†	58.78†	58.89†	59.54†	59.78†	59.01†
	Xu et al. (278)	59.09†	59.70†	59.23†	59.98≈	59.23†	59.41†
	GIARC-L (222)	60.76†	60.90†	59.56†	**60.94**	60.90≈	59.91†

Dataset (#Original features)	**Methods (#features)**	**Classifiers Accuracy (%)**					
		NB	**SVM**	**Bagging**	**NB**	**MLP**	**IPSO**
	Shu et al. (205)	60.60†	60.78†	60.09†	60.76≈	60.98≈	60.82†
	IFS (109)	**62.95**	**61.95**	**62.33**	60.12≈	**61.32**	**61.53**
Allaml (7129)	CFS (221)	81.12†	81.32†	82.62†	83.82†	82.21†	83.01†
	CON (230)	79.12†	80.32†	81.62†	82.82†	80.21†	82.01†
	CAR (201)	80.02†	80.32†	81.52†	82.72†	81.21†	82.02†
	Relief-F (210)	81.12†	81.32†	82.62†	83.82†	82.21†	83.11†
	RIDAS (205)	81.23†	81.43†	83.54†	83.34†	82.91†	84.10†
	IUAARI (245)	80.97†	80.38†	81.56†	82.34†	82.91†	83.90†
	IUAARS(187)	81.23†	81.43†	82.54†	82.76†	82.91†	84.00†
	Xu et al. (295)	80.08†	81.32†	82.76†	82.22†	82.08†	82.39†
	GIARC-L (173)	80.32†	81.24†	82.62†	81.82†	81.90†	82.33†
	Shu et al. (198)	81.12†	82.32†	82.09†	82.34†	82.21†	83.11†
	IFS (139)	**82.87**	**84.65**	**86.89**	**85.54**	**83.65**	**85.24**
Leukemia (7070)	CFS (147)	83.23†	84.23†	82.67†	82.02†	82.05†	83.23†
	CON (159)	84.05†	85.87†	84.23†	84.12†	84.21†	84.67†
	CAR (147)	83.47†	85.23†	84.56†	84.53†	84.54†	84.56†
	Relief-F (159)	84.23†	85.34†	84.12†	84.34†	84.34†	84.98†
	RIDAS (158)	85.01†	85.23†	84.30†	84.20†	83.20†	84.96†
	IUAARI (197)	84.23†	85.76†	85.10†	85.04†	84.34†	84.98†
	IUAARS (187)	85.20†	85.14†	84.12†	84.97†	85.30†	85.03†
	Xu et al. (243)	85.29†	85.90†	84.94†	85.59†	84.76†	84.93†
	GIARC-L (167)	85.91†	85.49†	84.91†	84.87†	83.60†	84.06†
	Shu et al. (123)	86.09†	85.93†	85.30†	84.50†	83.40†	85.36†
	IFS (88)	**87.51**	**87.32**	**87.32**	**87.55**	**87.44**	**87.03**

4.2.2 Comparative Analysis of DRED and IFS Method:

Comparisons of DRED [49], and IFS [57] methods are made based on the results given by the both the methods on experimental datasets. The method DRED selects multiple informative feature subset based on RST [17-20]. On the other hand, IFS method selects feature subset using RST and genetic algorithm. In Table 4.8, the average classification accuracies are measured based on the reduced feature subset generated by DRED and IFS method by using state of the art classifiers from Weka tool [218] and one incremental classifier IPSO [63] developed by me.

As DRED produces multiple features subsets so the results are given here based on the best feature subset found by DRED method for the comparison with IFS method. It is observed that the IFS method identifies lesser no of feature in a feature subset than DRED method with better classification accuracies. Computational time required for execution of DRED is less than IFS for few datasets.

Table 4.8: Comparison of DRED and IFS methods

Dataset	#Selected features		Average Accuracy (%)		Computational time(sec)	
	DRED	IFS	DRED	IFS	DRED	IFS
Wine	7	**6**	97.53	**97.84**	0.09	**0.01**
Heart	10	**9**	82.28	**84.86**	0.03	**0.01**
Glass	8	**8**	69.60	**77.18**	0.10	**0.01**
Zoo	**5**	5	94.47	**96.91**	0.06	**0.01**
Dermatology	**8**	9	98.24	**99.03**	**0.19**	**0.19**
Mushroom	4	**3**	97.83	**99.15**	45.38	**30.38**
Coil20	166	**157**	83.98	**84.55**	**1140.40**	1220.40
Orl	212	**109**	**61.92**	61.75	**828.98**	1054.78
Allaml	169	**139**	84.49	**84.86**	**1797.78**	2197.78
Leukemia	130	**88**	86.94	**87.71**	**1887.65**	2134.65

4.3 Summary:

The concept of rough set offers a sound theoretical foundation for constructing the reduct sets, which is very effective for dimensionality reduction and feature selection from the incremental data.

Feature selection enables the learning techniques to work more effectively, improving the rate of classification by reducing the influence of unwanted information. Feature selection through reduct generation in dynamic environment is the main issue of this chapter.

Since, the method of reduct generation is NP-hard; heuristic methods are developed to create single and multiple reduct in dynamic environment. In the chapter two simple but efficient feature selection methods are proposed.

The main objective is to select good features that are highly correlated with the class. The proposed algorithms can select features both in static and dynamic environment, where data arrives gradually with respect to the time.

First method (DRED) is based only on the RST for selecting important multiple feature subsets to classify datasets. Second method (IFS) is based on the RST and genetic algorithm for selecting important single feature subset to classify datasets.

Both the feature subset selection methods, DRED and IFS are compared among themselves as well as with several standard existing static feature selection methods and incremental feature selection method in terms of number of selected features, computational time and the classification accuracies using some state-of-the-art classifiers on reduced data to show their effectiveness. It is observed that the cardinality of the feature subset of IFS method is less and gives the highest average classification accuracies than almost all the considered methods in most of the cases due to suitable fitness function of GA in the IFS algorithm.

IFS method reduces a great deal of the time complexity of the overall system as the GA is applied only on newly added small group of objects on regular basis so the great difficulty of using it for its larger complexity does not affect much in most of the applications. This rough set theory-based incremental feature selection approach is equally applicable in the fields of social networking, bioinformatics, and big data analytics for finding the important feature subset in dynamic environment. In spite of the above advantages, some further investigations are required for full utilization of the proposed methods. With the changes of datasets, though the feature selection is done in incremental manner, but the method assumes that the new group of data has same set of features with the existing one. But in many applications, if the new objects with some other features are added then more investigations are required to select the minimal set of features.

ISBN: 978-93-90847-37-2

Chapter 5

Classification Analysis

5.1 Introduction:

Classification is a major research area in the field of data mining [1, 2] for the static as well as dynamic environment. Accurate classification analysis leads to better understanding of the underlying data. As now a day's structure of the data is difficult to understand directly, many machine-learning approaches have been used for classification of dataset [14, 15, 80]. These methods include the k-nearest neighbors [37], Bayesian approaches [7], Support Vector Machines [6, 166], Artificial Neural Network [3, 175], and decision trees [5, 152]. But a single classifier may not always give good results as it depends on the training capability of the classifier on the data itself. The solution is to apply multiple classifiers whose combined decision often gives better result compared to a single one. The effectiveness of combined classification system is strongly dependent on proper selection of base classifiers. Various architecture of classifier combination [179, 185-187] are developed to achieve this goal. The techniques that combine multiple classifiers have received much attention, and this is now a standard approach to improving classification performance in machine learning [32].

Researchers have already developed lots of ensemble methods including two widely used popular methods such as Bagging [191] and Boosting [192]. Many studies and research proposals discuss the way of developing a multiple classifier system (MCS) [16, 179, 185-187]. Ho (2000) [233] discussed coverage-based optimization and decision based optimization techniques for combination of multiple classifier system. Garbs et.al [234] described a classifier fusion method using Genetic Algorithm where a multidimensional selection of MCS is done. Zhou, Wu, and Tang (2002) [235] demonstrated that ensemble of selected classifiers give better result than all the classifiers used in an MCS. A genetic algorithm-based ensemble classifier [236] is proposed for bankruptcy prediction where multi co-linearity problem of classifiers are resolved using the Variance influence factor (VIF).

The paper [237] proposed an ensemble approach that attempts to obtain highly accurate classification system. In the paper [191], a bagging (bootstrap aggregating) method is introduced. Bagging improves generalization error by reducing the variance of the base classifiers. The performance of the bagging depends on the stability of the base classifiers. If a base classifier is unstable, bagging aids to reduce the errors related with random fluctuations in the training dataset. If a base classifier is stable, and robust to minor perturbations in the training set, then the error of the ensemble is primarily caused by bias in the base classifier. In [192], a boosting method is introduced that produces a series of base classifiers. Here, a set of samples is chosen based on the outcome of prior classifiers in the series. Samples that are incorrectly classified by previous classifiers are giving further chances for classification. Ada-Boost [193] is currently used as a promising boosting technique [192]. Many methods for constructing ensembles of classifiers have been developed, several are universal, and some are definite to particular algorithms.

For example, [238] uses 25 classifiers; the paper [239] uses 100 classifiers while it is extended up to 1000 in the paper [240]. To overcome such limitation, the paper [241] proposed a classification algorithm based on several decision tree classifiers using the concept of probability theory and graph theory (EOCDPG), where minimum number of rules is obtained to build an efficient ensemble classifier. The paper [242] integrated a multi-objective GA based feature selection scheme with an ensemble of classifiers (EOCASD) consisting of three basic classifiers: Artificial Neural Network (ANN), Support Vector Machine (SVM), and Decision Tree (DT).

Now a day's, increase of huge amount of data in every application demands an incremental learning technique for data analysis. One of such data analysis tasks in dynamic environment is to design an incremental classifier for decision making and consequently updating the knowledge base of the overall system. Classifier construction depicts extraction of interesting patterns from the large repository of data and predicts the future trends based on the existing patterns. The time complexity of the classification system increases gradually, and the system becomes inefficient while it is learned repeatedly for adding new group of data with the existing one in a certain interval of time. Without learning the same classifier for the whole data, if the knowledge of old data extracted by the classifier from the old data is used together with the new group of data to design the updated classifier, called incremental classifier, then time complexity reduces drastically.

The paper (Ziarko and Shan, 1993) [243] proposes a deterministic method for incremental modification of classification rules where the concept of decision matrix, based on rough set Theory, is used. Ulas, A et.al. (2009) [244] developed an incremental classifier construction strategy and discriminant ensembles of the classifiers for some classes only. Ozawa, S. et.al (2008) [245] proposes an incremental learning technique on chunk data for online pattern classification problem and Chen, Z. et.al (2007) [246] have proposed the incremental learning methodology for text data classification. A hybrid intelligent system (Seera and Lim, 2014) [247] is developed for medical data classification using Fuzzy Min–Max neural network, Classification and Regression Tree and Random Forest for the incremental data. An incremental learning technique of hierarchical appearance model is proposed in (Wenzel and Forstner, 2008) [248] to detect objects occurring in the images in hierarchy where the concept of incremental detection and classifying the new images with existing instances are used. A research paper (Ozawa et al., 2005) [249] on online face recognition using incremental learning technique is used for adaptive learning of new features and classifiers. G. Bakırlı et.al. (2011) [250] have designed an incremental Genetic Algorithm (GA) approach to development of a dynamic classifier for the incremental datasets.

The chapter discusses two different types of classification analysis in the static environment based on the reduced dataset obtained by feature selection methods, described in Chapter 3. The chapter also describes classification analysis in the dynamic environment for the incremental data.

The remaining part of the chapter is organized as follows: the classification analysis in static environment is discussed in section 5.2. In section 5.3, construction of a novel incremental classifier using PSO technique is described and finally, the chapter is summarized in Section 5.4.

5.2 Classification Analysis in Static Environment:

This section describes two different types of classification analysis techniques for the static environment based on the reduced dataset obtained by the best single reduct generation method (GRG) and multiple reduct generation method (MRG), described in Chapter 3. The classification rule generation method (CGRG) using the most informative feature subset obtained by GRG method [44] is described in Section 5.2.1. In Section 5.2.2, design of an ensemble classifier system (ECS) using the informative feature subsets obtained by MRG method [48] and Genetic Algorithm is described.

5.2.1 Classification Using the Most Informative Feature Subset (CGRG):

The classifier is an essential tool for finding the hidden nature of the dataset for categorization of different objects in the dataset. In the work, a classification rule generation technique (CGRG) [58] is proposed using the most informative feature subset selected by GRG method [44], discussed in Section 3.3.2, to predict the actual class of the objects present in dataset.

Classification technique [14, 15, 80] is a form of data analysis that can be used to extract models describing important data classes or to predict future data trends. For any learning system rule generation is a very important task for prediction of the output for a given input that it has not encountered before.

Here, the classification rules are considered of the structure "$x \rightarrow$ y" indicating that "if *x*, then y", where x is the description of condition attributes value and y is the description of decision attributes. As all attributes are not equally important, selecting only the significant attributes is the most important task in feature selection techniques. The reduced feature subset not only reduces the complexity of the overall system but also takes an important role to increase the accuracy of the system.

In CGRG, the concept of decision matrix [59] based on Rough Set Theory is used for generation of important classification rule set from the reduced dataset. GRG [44], the rough set based single reduct generation method and the decision matrix [59] based rule generation technique is applied together to find a compact set of classification rules for all decision classes of the dataset.

a. Classification Rule Generation using Decision Matrix Approach:

In this section, the rule generation technique using single reduct of the dataset is discussed. The concept of decision matrix [59] is used here. The method of finding the minimal set of consistent rules (i.e., which can be expanded into the Disjunctive Normal Form) that actually characterizing the decision system is illustrated here. For a set of condition attributes $C = \{C_1, C_2, C_3 \ldots, C_n\}$ and a decision attribute D, $D \notin C$, the rules should have the form, $C_i^a \; C_j^b \ldots C_k^c \rightarrow D^d$

Or $(C_i = a)\text{^}(C_j = b)\ldots\text{^}(C_k = c) \rightarrow (D = d)$

Where {*a*, *b*, *c*…} are the values from the domains of their respective attributes and *d* is the decision class value. The method for extracting rules is to form a decision matrix corresponding to each individual value *d* of decision attribute *D*. Actually, the decision matrix for value *d* of decision attribute *D* lists all attribute-value pairs that differ between objects having $D = d$ and $D \neq d$.

Example 5.1

To illustrate the decision matrix method, the sample decision system i.e., Table 3.11 of chapter 3 is considered. At first, applying the method GRG [44] on the sample decision system, the most informative reduct {*b′, d′*} is generated.

Now considering the obtained reduct, classification rules are generated for the sample dataset. For the reduct {*b′, d′*}, the sample decision system is reduced with two conditional attributes {*b′, d′*}, and the same decision attribute *D* with three distinct decision classes {1, 2, 3}, set of eight objects say {O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8}. Thus, the sample decision system is reduced to Table 5.1.

Table 5.1: Decision System for {*b′, d′*}

Attributes/Objects	***b′***	***d′***	***D***
O_1	1	0	1
O_2	1	2	1
O_3	1	0	1
O_4	4	0	2
O_5	4	0	2
O_6	4	1	3
O_7	4	1	3
O_8	5	1	3

According to decision matrix approach [59] to generate all the classification rules for each decision classes, the decision matrix for all decision values is generated.

At first the value of Decision class $D = 1$ is considered for generation of its decision matrix given in Table 5.2. In this case, the objects having $D = 1$ is {O_1, O_2, O_3} while the objects which have $D \neq 1$ is {O_4, O_5, O_6, O_7, O_8}.

The decision matrix for $D = 1$ lists all the differences between the objects having $D = 1$ and those having $D \neq 1$; that is, the decision matrix lists all the differences between {O_1, O_2, O_3} and {O_4, O_5, O_6, O_7, O_8}.

So, in the decision table for $D = 1$, the "positive" objects ($D = 1$) are in the rows, and the negative objects $D \neq 1$ are in the columns.

Table 5.2: Decision matrix for $D = 1$

Objects	O_4	O_5	O_6	O_7	O_8
O_1	b'^1	b'^1	$b'^1 d'^0$	$b'^1 d'^0$	$b'^1 d'^0$
O_2	$b'^1 d'^2$	$b'^1 d'^2$	$b'^1 d'^2$	$b'^1 d'^2$	$b'^1 d'^2$
O_3	b'^1	b'^1	$b'^1 d'^0$	$b'^1 d'^0$	$b'^1 d'^0$

Next, from the decision table for $D = 1$ given in Table 5.2, a set of boolean expressions are formed, one for each row of the table.

The items within each cell are aggregated disjunctively, and the individual cells are then aggregated conjunctively.

Thus, for the above table following three boolean expressions are generated:

- $(b'^1) \wedge (b'^1) \wedge (b'^1 \vee d'^0) \wedge (b'^1 \vee d'^0) \wedge (b'^1 \vee d'^0)$
- $(b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2)$
- $(b'^1) \wedge (b'^1) \wedge (b'^1 \vee d'^0) \wedge (b'^1 \vee d'^0) \wedge (b'^1 \vee d'^0)$

Expression (i) and (iii) are identical so the distinct expressions are:

- $(b'^1) \wedge (b'^1) \wedge (b'^1 \vee d'^0) \wedge (b'^1 \vee d'^0) \wedge (b'^1 \vee d'^0)$ and
- $(b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2)$

As redundancy is associated with the expressions, it is simplified using traditional boolean algebra. So, the statements

The first statement simplifies to b'^1, which gives the implication

$(b' = 1) \rightarrow D = 1$,

Likewise, the second statement $(b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2) \wedge (b'^1 \vee d'^2)$ simplifies to $(b'^1 \vee d'^2)$. This gives the implication

$(b' = 1) \vee (d' = 2) \rightarrow D = 1$,

So, the above implications are the following rule set for the decision class $D = 1$:

- $(b' = 1) \rightarrow D = 1$
- b) $(d' = 2) \rightarrow D = 1$

In this way all the decision matrices for the decision class $D = 2$ and $D = 3$ are formed and given in Table 5.3 and Table 5.4.

Table 5.3: Decision matrix for $D = 2$

Objects	O_1	O_2	O_3	O_6	O_7	O_8
O_4	b'^4	$b'^4d'^0$	b'^4	d'^0	d'^0	$b'^4d'^0$
O_5	b'^4	$b'^4d'^0$	b'^4	d'^0	d'^0	$b'^4d'^0$

After simplification of the boolean expressions for decision matrix $D = 2$ given in Table 5.3, following rule is generated:

- $(b' = 4)$ ^ $(d' = 0) \rightarrow D=2$

Table 5.4: Decision table for $D = 3$

Objects	O_1	O_2	O_3	O_4	O_5
O_6	$b'^4d'^1$	$b'^4d'^1$	$b'^4d'^1$	d'^1	d'^1
O_7	$b'^4d'^1$	$b'^4d'^1$	$b'^4d'^1$	d'^1	d'^1
O_8	$b'^5d'^1$	$b'^5d'^1$	$b'^5d'^1$	$b'^5d'^1$	$b'^5d'^1$

After simplification of the boolean expressions for the above decision table $D = 3$ given in Table 5.4, following rules are generated:

- $(d' = 1) \rightarrow D=3$
- $(b' = 5) \vee (d' = 1) \rightarrow D=3$

Thus, the final classification rules for all the decision classes i.e. $D = 1$, 2, and 3 for the reduct $\{b', d'\}$ are:

- $(b' = 1) \rightarrow D=1$
- $(d' = 2) \rightarrow D=1$
- $(b' = 4)$ ^ $(d'=0) \rightarrow D=2$
- $(d' = 1) \rightarrow D=3$
- $(b' = 5) \rightarrow D=3$

In this way, most important and compact classification rules are generated from the dataset by integrating the GRG method [44] and the decision matrix approach [59] to classify the test objects.

b. Experimental Results of CGRG Method:

Experimental studies presented here provide evidence of effectiveness of CGRG method [58] on experimental datasets [27, 28], summarized in the section 2.2. In all the dataset, the important feature subset is selected using GRG method, described in Section 3.3.2. The classification rules are generated using proposed CGRG method from each important feature subset and the accuracies are measured and compared with state-of-the-art classification methods like NB, SVM, KNN, Bagging J48, and MLP, as listed in Table 5.5.

It is observed from Table 5.5 that the accuracy of the proposed CGRG is not better than all the methods, but comparable to other methods.

Table 5.5: Comparison based on classification accuracy (%)

Dataset	**Classification methods**						
	NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**CGRG**
Wine	97.70	**97.91**	97.48	97.09	96.65	96.49	96.35
Heart	85.27	85.42	84.81	84.52	**84.89**	83.43	84.70
Glass	67.28	64.48	**83.64**	76.63	70.09	75.23	74.42
Zoo	97.01	95.04	95.05	95.48	95.89	95.12	**97.51**
Dermatology	98.23	**98.57**	98.45	98.51	98.41	98.93	97.50
Mushroom	99.04	99.02	**99.34**	98.78	96.89	99.08	97.02
Coil20	83.12	80.01	81.87	82.32	84.32	84.43	84.23
Orl	60.01	59.03	60.07	61.02	61.02	60.21	60.97
Allaml	83.43	83.67	84.32	84.21	83.70	84.76	83.52
Leukemia	87.67	86.98	86.67	86.98	88.20	87.98	86.97

As accuracy is not only the measurement of effectiveness of the classifiers, some statistical measurements are performed like, Recall (Sensitivity), Fall_out, Specificity and F1_score are calculated using Equation (2.25), Equation (2.26), Equation (2.27) and Equation (2.28) and the results for all the existing and the proposed CGRG classifiers are listed in Table 5.6.

It is observed that, the CGRG method gives the satisfactory results for all of the experimental dataset in comparison with other algorithms.

Table 5.6: Statistical measure of CGRG and other competitive algorithms

Dataset	**Methods**	**Recall**	**Fall_out**	**Specificity**	**F1_Score**
Wine	NB	**0.98**	0.03	0.97	0.97
	SVM	0.97	0.03	0.97	0.97
	KNN	0.97	**0.02**	0.97	0.96
	Bagging	0.97	0.03	0.96	0.97
	J48	0.97	0.03	0.96	0.97
	MLP	0.96	0.04	0.97	0.96
	CGRG	0.96	**0.02**	0.96	0.96
Heart	NB	**0.85**	0.15	**0.86**	**0.85**
	SVM	**0.85**	0.14	0.84	**0.85**
	KNN	**0.85**	0.16	0.85	0.84

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
	Bagging	**0.85**	0.14	0.84	**0.85**
	J48	0.82	0.17	0.82	0.83
	MLP	0.83	0.18	0.82	0.82
	CGRG	**0.85**	**0.13**	0.84	**0.85**
Glass	NB	0.67	0.31	0.67	0.66
	SVM	0.64	0.36	0.65	0.64
	KNN	**0.83**	**0.16**	**0.82**	**0.82**
	Bagging	0.77	0.23	0.76	0.77
	J48	0.70	0.27	0.69	0.70
	MLP	0.75	0.25	0.74	0.75
	CGRG	0.74	0.25	0.74	0.74
Zoo	NB	0.94	0.06	0.93	0.94
	SVM	0.94	0.06	0.94	0.93
	KNN	0.94	0.06	0.92	0.94
	Bagging	0.94	0.05	0.94	0.93
	J48	0.95	0.05	0.95	0.95
	MLP	0.95	**0.04**	0.95	0.94
	CGRG	**0.98**	**0.04**	**0.98**	**0.98**
Dermatology	NB	0.98	**0.01**	**0.98**	0.98
	SVM	**0.99**	**0.01**	**0.98**	0.98
	KNN	0.98	0.02	0.97	0.98
	Bagging	0.99	0.02	**0.98**	0.98
	J48	0.98	0.02	**0.98**	0.97
	MLP	**0.99**	**0.01**	0.97	0.98
	CGRG	0.98	0.02	**0.98**	**0.99**
Mushroom	NB	**0.99**	**0.01**	0.98	0.98
	SVM	**0.99**	**0.01**	0.98	**0.99**
	KNN	**0.99**	**0.01**	**0.99**	0.98
	Bagging	**0.99**	**0.01**	0.98	0.98
	J48	0.97	0.03	0.96	0.97
	MLP	**0.99**	**0.01**	0.96	0.97
	CGRG	0.97	**0.01**	0.97	0.97
	NB	0.83	0.16	0.83	0.83
	SVM	0.80	0.20	0.79	0.80

Dataset	Methods	Recall	Fall_out	Specificity	F1_Score
Coil20	KNN	0.82	0.17	0.81	0.81
	Bagging	0.82	0.18	0.82	0.82
	J48	**0.84**	0.16	0.83	**0.84**
	MLP	**0.84**	0.16	0.83	**0.84**
	CGRG	**0.84**	**0.15**	**0.84**	**0.84**
Orl	NB	0.60	0.40	0.59	0.59
	SVM	0.59	0.39	0.59	0.59
	KNN	0.60	0.39	0.59	0.60
	Bagging	**0.61**	0.38	**0.60**	0.59
	J48	**0.61**	0.38	**0.60**	**0.61**
	MLP	0.60	**0.37**	**0.60**	0.59
	CGRG	**0.61**	0.38	**0.60**	**0.61**
Allaml	NB	0.83	0.17	0.83	0.82
	SVM	0.84	0.16	0.83	0.83
	KNN	0.84	0.16	0.83	0.83
	Bagging	0.84	0.16	0.83	**0.84**
	J48	0.84	0.16	0.83	**0.84**
	MLP	**0.85**	0.16	**0.84**	**0.84**
	CGRG	0.84	**0.14**	**0.84**	**0.84**
Leukemia	NB	**0.88**	0.12	**0.88**	0.87
	SVM	0.87	0.13	0.87	0.86
	KNN	0.87	0.13	0.87	0.86
	Bagging	0.87	0.13	0.86	0.87
	J48	**0.88**	0.12	0.87	0.87
	MLP	**0.88**	0.12	0.87	**0.88**
	CGRG	0.87	**0.11**	**0.88**	**0.88**

5.2.2 Ensemble Classifier Design using Multiple Feature Subsets (ECS):

Major motivation behind combining multiple classifiers is to achieve more classification accuracy compared to a single one.

The algorithm has been designed for construction of an optimal ensemble classifier system (ECS) using the multiple feature subsets generated by MRG algorithm [48] described in Section 3.3.2, and Genetic Algorithm (GA) [23].

In the method, suppose N number of feature subsets from a decision system is selected after applying MRG algorithm [48], so N number of base classifiers is constructed each from one of N number of feature subsets. It is noted that some base classifiers may perform well individually on the training dataset but others may show poor performances.

In the proposed ECS method [60] the best combination of classifiers from N different base classifiers is determined using genetic algorithm.

The method searches a particular combination of classifiers that produces the maximum classification accuracy. In the first phase of ECS, in the first phase, MRG algorithm is used to select the important feature subsets from the dataset.

Thus, a dataset is considered as a combination of multiple sub-datasets, each corresponding to a feature subset called reduct. Now, from each reduct, rule-based classifier is constructed using the concept of association rule mining [61, 62].

In this way, base classifier models, one for each reduct are generated. In the second phase, base classifiers are fused, and an optimal ensemble classifier system (ECS) is developed using GA and performance of the classifier is measured to express its effectiveness.

Here, combination of the best performing classifiers performs better compared to a single one, as objects which are not classified by one classifier may be classified by another classifier.

For a particular dataset, the ECS combines the classifiers with the objectives to maximize the classification accuracy of the ensemble classification system. The overall flow diagram of the ECS method is briefly given in Figure 5.1.

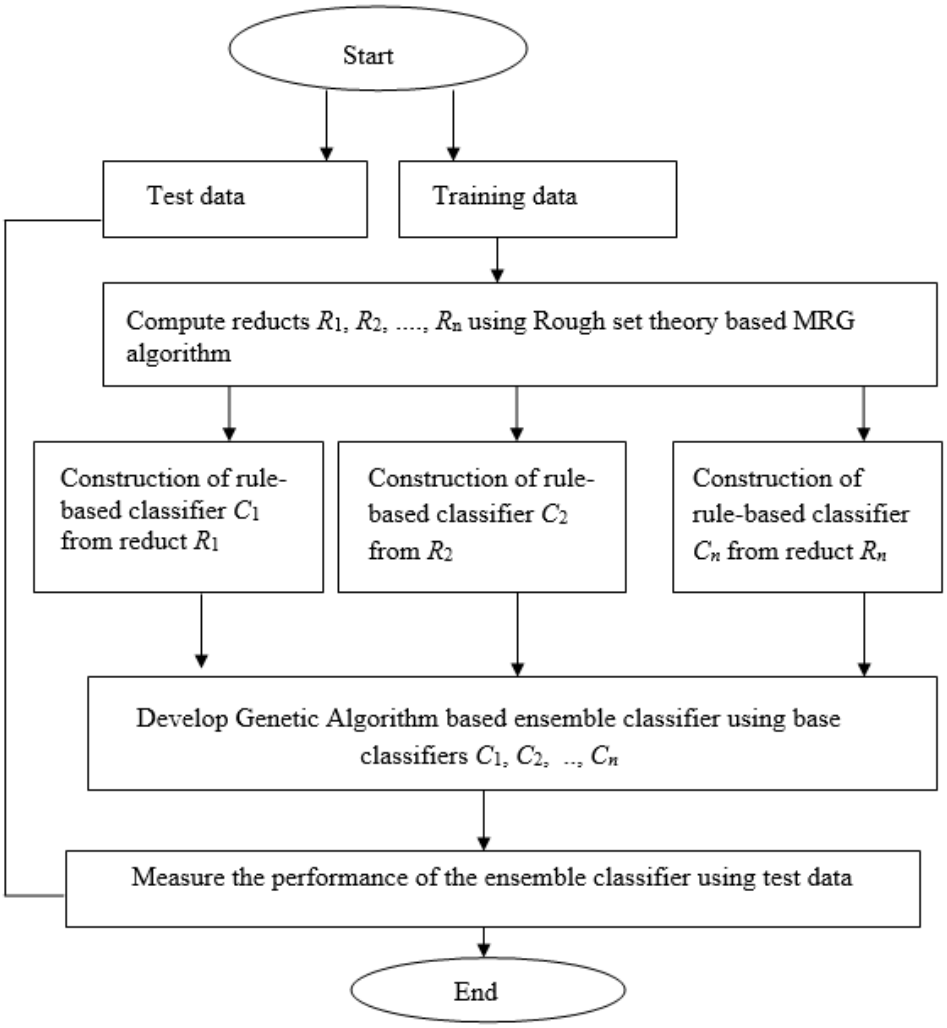


Figure 5.1: ECS workflow diagram

a. Construction of Base Classifiers (RBCM):

This section discusses the construction of rule base classification model RBCM which is the base classifier of the proposed ECS system. Here, the base classifiers are designed based on the set of important feature subsets called reducts selected using MRG algorithm [48]. The rules of the base classifiers are defined using two interesting measures namely support and confidence, the terms used for association rule mining. The whole process of development of the rule-based classifier model RBCM is divided into two phases such as (A) feature subsets selection or reducts generation and (B) Classification rule set Generation.

- **Reducts Generation:**

Feature subset selection or reducts generation using MRG algorithm is discussed in section 3.3.2.

- **Classification Rule Set Generation:**

The method develops rule-based classifiers based on the generated reducts. Initially, many possible rule items are generated based on the core and noncore attributes values for each reduct. The core attribute values are independently considered first and set as rule items and if a rule item is not an actual rule in the rule set (which is decided by two interesting measures namely, support and confidence of association rule mining concept), then combined it with a noncore attribute (in reduct) values and a new rule item is formed, which is checked to determine if it is a true rule in the rule set or not. The process is continued until all the attribute values in the reduct are exhausted. Thus, the method gives us a rule based classifier with a set of rules for a reduct. The same process is performed for all generated reducts, and a set of rule-based classifiers is designed for the decision system. To determine if a rule item is a true rule in the rule set, a weighted value is calculated for each rule item *r* using the support and confidence measures, concept of association rule mining [61, 62]. The formula of calculating rule weight is given in Equation (5.1).

Weight of rule r (W_r) = w**Confidence*$_r$ + (1-w) **Support*$_r$ (5.1)

where the value of w is set experimentally.
An association rule r is of the form ($C_1 = p1 \wedge C_2 = p2$) => ($D = d$), where C_1,C_2,, are the conditional attributes, $p1$, $p2$, …, are the values of $C1$,$C2$, …, respectively, and $D = d$ is a class with label d. So, a rule is a mapping from $C \rightarrow D$ i.e. r: $C \rightarrow D$.
Then the support (*Support*$_r$) and confidence (*Confidence*$_r$) of an association rule r can be calculated using the Equation 5.2 and Equation 5.3.

Definition5.1: If C and D are two item sets corresponding to a database T and r: $C \rightarrow D$, an association rule then

$$Support\ (C \rightarrow D) = \frac{tuples\ containing\ both\ C\ and\ D}{Total\ number\ of\ tuples\ in\ T} \quad (5.2)$$

$$Confidence\ (C \rightarrow D) = \frac{tuples\ containing\ both\ C\ and\ D}{Total\ number\ of\ tuples\ containing\ C} \quad (5.3)$$

If the weighted value of a particular rule item is more than the experimental threshold value, then it is selected as a classification rule and stored in the rule set. Otherwise, the rule item is combined with the next noncore attributes values and the same process is repeated to decide if the new rule items are actually the rules of the final rule set or not. In this way, possible rules are obtained in the rule set. Now, to get the more compact rule set, rule pruning is also done to remove irrelevant rule components from the rules without affecting the rule quality.

The process is quite simple as iteratively one component at a time is removed and rule quality is recomputed. If the quality of a rule is not decreased after removing any component, then the component is permanently removed from the rule. Thus, all unnecessary rule components of the rules are deleted and finally, a more compact rule set is generated. In this way for each reduct of the reduct set a base classifier is constructed. These base classifiers are used to design ensemble classifier ECS.

b. Ensemble of Classifiers:

Here, Genetic algorithm (GA) is used [23] to construct an optimal ensemble classification System (ECS). In GA, population of candidate solutions contains chromosomes, and the size of the chromosome is set as the number of base classifiers (*N*) to be combined. The initial population is generated randomly as a collection of binary strings called chromosomes. Each chromosome in the initial population represents the combinations of some base classifiers randomly selected.

- **Genetic Algorithm Preliminaries:**

The method uses a steady state selection strategy with effective fitness function, uniform crossover operation, and mutation operation with a probability factor to maintain diversity in the population.

i. Population Generation:

Initially a random population is generated where each chromosome is encoded as a binary string of 0 and 1. The length of each chromosome is equal to the numbers of base classifiers used.

There is a one-to-one correspondence between the bits in chromosome and the classifier in the classifier set. The chromosome in the population is the candidate solution, where '0' means corresponding classifier does not take part in the ensemble process and '1' means corresponding classifier takes part. As the initial population is generated randomly, classifiers for ensemble process are initially selected at random basis, which after convergence of the GA give an optimal ensemble of classifiers

ii. Fitness Function:

As fitness function determines quality of solution (i.e., chromosome) in the population, so a strong global best fitness function is imperative for obtaining good result.

For the classifier, the classification accuracy is likely to be the best performance measure, so in the method Combined Classification Accuracy is used to define the fitness function.

A chromosome is evaluated by its fitness value computed as that is the accuracy of the associated classifier defined in equation (5.4) on the training dataset on which the model is learned.

$$\text{Classification accuracy} = \frac{TP+TN}{P+N} \qquad (5.4)$$

Where TP is number of the positive object classified as positive, FP is the number of negative objects classified as positive, P is the total number of positive objects and N is the total number of negative object.

iii. Genetic Operations:

As the convergence of genetic algorithm depends on the proper selection of parameters, so selection of parameter values is an important task here. Selection is the first genetic operator applied on the population. Here, ranks based roulette-wheel selection method [23] is used to select the chromosomes until it is continued until the mating pool is filled up. The population may lose the best chromosome by crossover and mutation. So, elitism operation includes 5% of chromosomes with the optimum fitness values into the mating pool. Crossover operator is applied to the mating pool hoping that it would create a better chromosome. Crossover operator used is uniform crossover with probability 0.9. New chromosomes generated by crossover are taken into the next generation population only if their fitness is better than that of their parent chromosomes. Mutation is also used to maintain diversity in the population. Mutation operation involves flipping of a bit in a chromosome, changing 0 to 1 and 1 to 0. It is done in a random bit of each chromosome with probability 0.001. All parameter values are determined experimentally, and the algorithm terminates when a predefined number of generations are exhausted. Here, the stopping criterion is set to 100 iterations and the genetic search is repeated by 50 generations. The best chromosome of the final population provides the ensemble classifier system with base classifiers associated to bit '1' in the chromosome. The ensemble classifier construction algorithm is defined below.

Algorithm: ECS (*DS*)

Input: *N* number of feature subsets and corresponding reduced decision subsystems obtained from *DS*

Output: Ensemble classifier system for DS

Step I: Construct *N* number of base classifiers from *N* feature subsets using RBCM method.

Step II: Initialize the population *P* of size $M = 100$

Step III: Set chromosome length as $|N|$, the number of base classifiers in *DS*.

Step IV: Calculate fitness value for each chromosome *ch* in *P* using equation (5.4).

Step V: Use Roulette Wheel Selection to select the best chromosomes into the mating pool based on their fitness values.

Step VI: Apply uniform crossover and mutation operations on chromosomes in themating pool with crossover probability of 0.9 and mutation rate of 0.001.

Step VII: Choose the chromosomes for the next generation with 50% replacement of the parent population.

Step VIII: Repeat Step IV to step VII until the GA converges (i.e., 50 generations are performed).

Step IX: The best chromosome of the final population forms the ensemble classification system ECS of the entire system *DS*.

c. Results of the ECS Method:

Performance evaluation of the ECS algorithm [60] and comparative study with some state-of-the-art classification methods are discussed using real world experimental datasets describe in the section 2.2.

Major motivation behind ensemble classifier is to achieve more classification accuracy in comparison with a single one.

- **Parameters Setup and Preprocessing**

The parameters used for GA in ECS method is shown in Table 5.7. These parameters are selected after several test evaluation of proposed method and dataset instance until reaches to the best configuration in terms of the quality of solutions and computational effort.

Table 5.7: Parameters of GA environment

Parameters	**Value**
Population size	100
Number of generations	50
Probability of crossover	0.9
Probability of mutation	0.001

- **Classifiers Used:**

The ECS method uses base classifiers obtained from each reduct using RBCM method for designing ensemble classifier. In the experiments, '*10-fold cross validation*' is used to evaluate classification performance where in each iteration 90% samples (9-fold) are used for training and 10% (1-fold) other samples are used for test purpose.

- **Comparative Study:**

The method ECS is compared with individual base classifier RBCM from which ECS is generated and with some popular ensemble classification methods, Bagging [191], Boosting [192], the classifiers proposed by Das et al. [241], and Zhang [242], where the last two are named here for reference as EOCDPG and EOCASD respectively.

i. **Comparison based on Classification Accuracy with the Single Classifier:**

Here, classification accuracy of each individual base classifier and Ensemble Classifiers are calculated using test data for each experimental benchmark dataset.

For different dataset maximum accuracy value achieved by each individual base classifier is compared with the accuracy value achieved by proposed ensemble classifier, which is listed in Table 5.8. It is seen from Table 5.8 that the ensemble classification system provides more accuracy than individual base classifiers in all the cases.

Table 5.8: Comparison of ensemble classifier with individual base classifier

Data set	No of classifier	Classifier Used	Maximum accuracy by individual base classifiers (%)	ECS
Wine	4	RBCM	96.00	**96.52**
Zoo	10	RBCM	95.00	**95.55**
Heart	8	RBCM	83.15	**84.89**
Dermatology	10	RBCM	96.89	**97.57**
Mushroom	10	RBCM	96.32	**97.69**
Coil20	50	RBCM	83.67	**84.35**
Orl	50	RBCM	60.01	**61.78**
Allaml	50	RBCM	83.01	**84.45**
Leukemia	50	RBCM	86.62	**86.99**

ii. Comparison with Popular Ensemble Classifiers:

The classification accuracies of the ECS method and other popular compared ensemble classifier methods are shown in Table 5.9. The best results are marked by bold font.

Table 5.9: Comparison of ECS with other ensemble classifiers

Data set	Bagging	Boosting	EOCDPG	EOCASD	ECS
Wine	**97.09**	95.05	94.74	94.74	96.52
Zoo	95.48	95.03	94.56	95.01	**95.55**
Heart	84.52	82.62	83.06	84.09	**84.89**
Dermatology	**98.51**	96.06	94.39	**98.59**	97.57
Mushroom	98.78	96.34	96.10	97.40	97.69
Coil20	82.32	83.12	82.16	83.77	**84.35**
Orl	61.02	60.04	59.12	59.87	**61.78**
Allaml	84.21	83.65	83.98	83.43	**84.45**
Leukemia	86.98	85.67	86.01	85.54	**86.99**

The Table 5.9 shows that the ECS method gives the best results for six datasets whereas EOCASD method gives the best results for dermatology dataset and Bagging method gives the best result for wine and mushroom dataset. The best results are marked by bold font in Table 5.9.

iii. Comparison based on Statistical Measures:

During classification accuracy computation, some statistical measurements [37, 38] like, Recall (Sensitivity), Fall_out, Specificity and F1_score are calculated using Equation (2.25) to Equation (2.28) respectively.

The calculated statistical measurements are given in Table 5.10 for experimental datasets using various classifiers.

It is observed that, the ECS method gives the satisfactory results for almost all of the experimental dataset while Bagging, Boosting, EOCDPG, and EOCASD provide comparatively poor result.

Table 5.10: Performance comparison of ECS with other ensemble algorithms

Dataset	Ensemble Classification Methods	Recall	Fall_out	Specificity	F1_Score
Wine	Bagging	**0.97**	0.03	**0.97**	0.96
	Boosting	0.95	0.04	0.96	0.95
	EOCDPG	0.95	0.05	0.95	0.96
	EOCASD	0.95	0.04	0.96	0.95
	ECS	**0.97**	**0.02**	**0.97**	**0.97**
Zoo	Bagging	**0.96**	0.04	**0.96**	0.95
	Boosting	0.95	0.05	0.94	0.95
	EOCDPG	0.95	0.04	0.95	0.94
	EOCASD	0.95	0.04	0.94	0.95
	ECS	**0.96**	**0.03**	0.95	**0.96**
Heart	Bagging	**0.85**	**0.15**	**0.85**	**0.86**
	Boosting	0.83	0.16	0.84	0.83
	EOCDPG	0.83	0.17	0.83	0.83
	EOCASD	0.84	0.16	0.84	0.84
	ECS	**0.85**	**0.15**	**0.85**	**0.86**
Dermatology	Bagging	**0.99**	**0.01**	0.98	0.98
	Boosting	0.96	**0.01**	0.96	0.95
	EOCDPG	0.94	0.05	0.95	0.94
	EOCASD	**0.99**	**0.01**	**0.99**	0.98
	ECS	0.98	0.02	0.98	**0.99**
Mushroom	Bagging	**0.99**	**0.01**	**0.98**	**0.98**
	Boosting	0.96	0.04	0.96	0.95
	EOCDPG	0.96	0.04	0.96	0.97

Dataset	Ensemble Classification Methods	Recall	Fall_out	Specificity	F1_Score
	EOCASD	0.97	0.03	0.96	0.97
	ECS	0.98	0.03	**0.98**	0.97
Coil20	Bagging	0.82	0.16	0.82	0.83
	Boosting	0.83	0.20	0.82	0.82
	EOCDPG	0.82	0.17	0.82	0.81
	EOCASD	**0.84**	0.16	0.83	**0.84**
	ECS	**0.84**	**0.15**	**0.84**	**0.84**
Orl	Bagging	0.61	0.39	**0.60**	**0.61**
	Boosting	0.60	0.39	0.59	0.60
	EOCDPG	0.59	0.39	0.59	0.60
	EOCASD	0.60	0.38	**0.60**	0.59
	ECS	**0.62**	**0.35**	**0.60**	**0.61**
Allaml	Bagging	**0.84**	0.17	0.83	0.83
	Boosting	**0.84**	**0.16**	0.83	0.83
	EOCDPG	**0.84**	**0.16**	0.83	0.83
	EOCASD	0.83	0.17	0.83	**0.84**
	ECS	**0.84**	**0.16**	**0.84**	**0.84**
Leukemia	Bagging	**0.87**	**0.12**	**0.88**	0.87
	Boosting	0.86	0.13	0.86	0.86
	EOCDPG	0.86	0.14	0.87	0.87
	EOCASD	0.86	0.13	0.87	0.86
	ECS	**0.87**	**0.12**	**0.88**	**0.88**

iv. Comparison based on Statistical Performance:

As the results show almost similar accuracy, so a statistical analysis is done to express the significance of the method. The Wilcoxon's rank sum test [39], a nonparametric test is used for checking statistical significance of the method.

It is used for two populations when samples are independent. If X and Y are independent samples with different sample sizes, the test returns the rank sum of the first sample, Wilcoxon rank sum test (shows in Table 5.11 for ECS) ranks the differences of performances of two classifiers and compares ranks for positive and negative differences.

Table 5.11: Wilcoxon's Rank Sum Test Results of ECS with other algorithms

SV	ECS	SVM	NB	KNN	J4.8	Bagging	Boosting	ECDPG	ECASD
p-values	NA	1.8767e-04	1.2567e-04	1.7677e-04	1.4666e-04	1.8277e-04	1.7677e-04	1.8329e-04	1.8218e-04
h-values	NA	1	1	1	1	1	1	1	1

Remark: *P-value* of the test is returned as a positive scalar from 0 to 1 where, p is the probability of observing a test statistic as or very extreme than the experimental value under the null hypothesis. Here, null hypothesis is stated as "X and Y both samples belong to same population and there is no statistical significance of the proposed algorithm over standard algorithm".

Result of the hypothesis test.

- If = 1, this indicates rejection of the null hypothesis.
- If $h = 0$, this indicates a failure to reject the null hypothesis.

For each pair of tests, the null hypothesis is rejected (as per Table 5.11). Therefore, for each of those cases the test shows that the proposed method is statistically significant over the standard classification algorithms.

5.3 Incremental Classifier Design using PSO Technique (IPSO):

Classifier construction depicts extraction of interesting patterns from the large repository of data and predicts the future trends based on the existing patterns. The time complexity of the classification system increases gradually, and the system becomes inefficient while it is learned repeatedly for adding new group of data with the existing one in a certain interval of time. Without learning the same classifier for the whole data, if the knowledge of old data extracted by the classifier is used together with the new group of data to design the updated classifier, called incremental classifier, then time complexity reduces drastically. Here, the concepts of Particle Swarm Optimization (PSO) [24, 67-69] technique and Association Rule Mining [61, 62] are used to design an incremental rule-based classification system. The algorithm handles incremental data effectively for upgrading the existing classifier by modifying the existing rule sets whenever new set of data is added with the previous dataset. At first, PSO [67- 69] based training process is performed on the existing dataset to find out the initial optimal set of classification rules. When new data arrives, if static PSO based training process is re-run on the whole dataset (consisting of both existing and new data) for developing the modified classifier, then not only the efficiency of the system degrades with increased data volume but also the previous classifier already trained by the existing dataset is totally unusable which increases the overhead time of the overall system. As the volume of the dataset increases with time, learning of whole data in dynamic environment rapidly increases the training time and makes the classification system inefficient. So, it is desirable to upgrade the classifier with the help of new group of data and existing knowledge extracted from the previous dataset.

Here, a Particle Swarm Optimization based incremental classification method (IPSO) [63] is proposed (IPSO), which analyzes the new dataset and updates the previous classification

rule set dynamically with a reduced training time. A suitable fitness function is also designed for the proposed IPSO method [63]. The details of the heuristics and their importance are provided in subsequent section. Figure 5.3 provides the schematic diagram of the proposed incremental classifier design technique (IPSO) for incremental datasets.

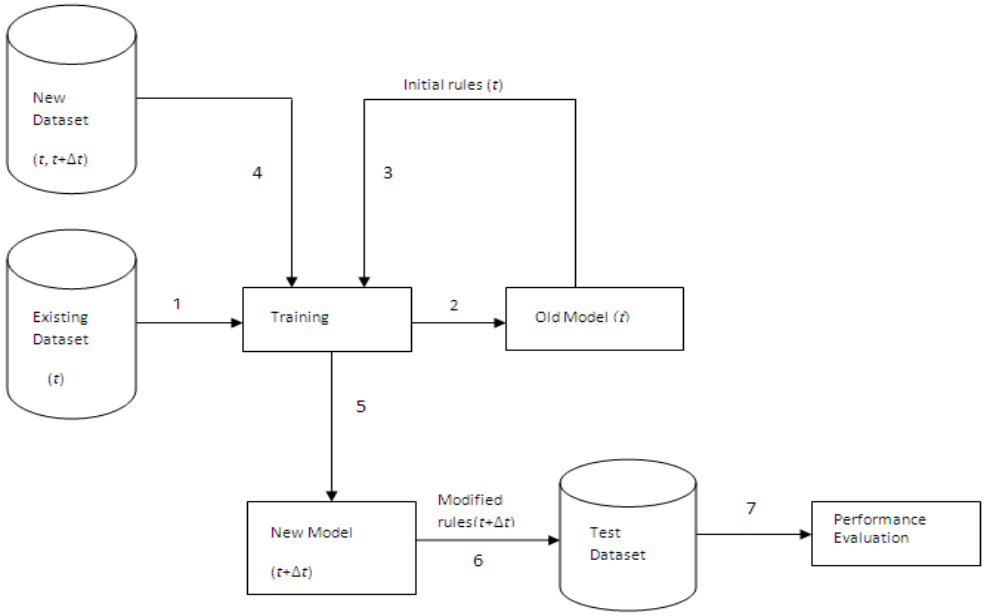


Figure 5.2: IPSO model

Figure.5.2 shows the sequence of steps in which the incremental classifier is designed and evaluated in the time interval (t, $t+\Delta t$). The dataset available at time t is considered as existing dataset and the classification model is trained on this dataset using the concept of PSO which provides initial rule set for dataset at instant of time t. Now after Δt time, new data, called incremental data are stored as new dataset in time interval (t, $t+\Delta t$). In the proposed IPSO technique, initial rules and new dataset are feed in training process and the modified classification model, called new model, is generated for whole dataset available at instant of time $t+\Delta t$. This incremental classification model is evaluated by test dataset for performance measurement. At next instant of time, the new model at ($t+\Delta t$) time is considered as old model and new group of data is taken into account for constructing further modified classifier. This process is continued after every interval of time when a group of data enters into the system. Thus, a rule based dynamic classifier is designed for incremental dataset in efficient way.

5.3.1 Dynamic Classifier for Incremental Data:

Incremental learning technique is a continuous learning process appropriate for software agent learning task where agents should adapt themselves incrementally and continuously with the dynamic environment.

Here an incremental classifier is designed with the objective that the number of classification rules will be minimal. As PSO is one of the most effective evolutionary algorithms, frequently used for solving optimization problem efficiently, so incremental PSO (IPSO) [63] is the suitable choice for designing incremental classifier in ad hoc basis. In the proposed incremental classifier design, optimized classification rules are generated for the incremental data dynamically using the concept of Association rule mining and PSO algorithm.

The algorithm handles incremental data effectively for an optimized rule set generation by modifying the existing knowledge base whenever new data are available. In IPSO model shown in Figure. 5.3, firstly, PSO based training process is performed on the existing dataset to find out the initial optimal classification rules. When a new group of data arrives, an incremental PSO (IPSO) is run using existing classifier and new group of data to develop a dynamic classifier and classification performance is evaluated on test data. So, the proposed IPSO algorithm analyzes the new dataset in every interval of time and updates the previous knowledge base dynamically with a reduced training time.

The steps of the proposed IPSO algorithm are described below:

a. Dataset Preparation:

To demonstrate the IPSO method [63] using benchmark static datasets [27, 28] are treated as incremental datasets. For this reason, any dataset named here as a decision system *DS* is divided randomly into three subsystems as DS_1, DS_2, and DS_3. DS_1 is considered as old dataset, DS_2 as new dataset or incremental dataset and DS_3 as test dataset used to evaluate the performance of the proposed IPSO algorithm. The algorithm deals with only discrete valued dataset and so the continuous dataset is discretized [219] before use of the algorithm.

b Initial Classifier Generation:

Initially, optimal set of classification rules is generated from the old dataset DS_1 using the PSO algorithm [67-69]. The details of rule set generation are discussed in IPSO algorithm in next section. The only difference of IPSO with static PSO is that, in static PSO (or simply, PSO), the fitness value is computed using the confidence of rules but in IPSO, confidence of rules together with the similarity between new rule and existing rules are taken into account. Every rule has two parts, antecedent and consequence where antecedent comprises of some conditional attributes together with their values and consequence has decision attribute together with the corresponding decision value or class label. In a rule like ($C_1 = p1 \wedge C_2 = p2) \Rightarrow (D = d$), each conditional attribute together with the corresponding value is termed as rule component, where C_1, C_2,…, are the conditional attributes, $p1$, $p2$, …, are the values of C_1,C_2, …, respectively, and $D = d$ is a class with label d. So, a rule is formed by some rule components.

c. Dynamic Classifier Generation:

When new dataset DS_2 arrives, the existing classifier must be upgraded to make it more powerful as more information is now available.

Thus, the upgraded classifier must be familiar about the whole dataset $DS_1 \cup DS_2$. To make the model efficient and scalable, instead of using whole dataset $DS_1 \cup DS_2$, only the previously extracted knowledge from the old dataset i.e., optimal rule set of the initial classifier trained in DS_1 is used instead of reusing DS_1 redundantly. Thus, the new dataset DS_2 and the previous rule set obtained by section 5.3.1.2 are used to generate the modified classifier called dynamic classifier for the whole dataset $DS_1 \cup DS_2$. Here, a discrete PSO [67] algorithm is used for this purpose. Let $DS_2 = \{U_2, A, D\}$, where U_2 is the set of objects, $A = \{C_1, C_2, \ldots, C_n\}$ is the set of condition attributes and $D = \{d_1, d_2, \ldots, d_m\}$ be the decision attribute with m class labels.

The PSO algorithm runs separately for each class of objects and provides a set of rules for that class. The objective function for a particle in PSO algorithm is defined using the rules generated by DS_1 and confidence value (a statistical measure) of the particle in DS_2. Thus, rules for that class are obtained from the final population, which is the new rule set for system $DS_1 \cup DS_2$. Similarly, PSO is run for all classes and all updated rules are obtained for the current system dynamically. All the steps of the IPSO algorithm are described below for the data subset of a particular class say, ($D = d_1$); same steps are used for the data subsets of other classes.

- **Generation of Initial Population:**

First of all, the set S_2 of objects with class value ($D = d_1$) are selected from DS_2 using relational algebra operation projection π [251], as given in equation (5.5).

$$S_2 = \pi_C \left(\sigma_{D= d_1}(DS_2)\right) \tag{5.5}$$

In S_2, let attribute C_i has c_i-distinct values ($\forall i = 1, 2, \ldots, n$) which are represented by $(C_i = c_{i1}^1)$, $(C_i = c_{i1}^2)$, …, $(C_i = c_{i1}^{c_i})$. Now for simplicity let, these values are represented by the set $C_i = \{(C_i = c_{i1}^1), (C_i = c_{i1}^2), \ldots, (C_i = c_{i1}^{c_i})\}$. Next, all $(c_1 + c_2 + \cdots + c_n)$ distinct attribute values in S_2 are indexed by natural number using equation (5.6).

$$\begin{aligned} index\left(C_i = c_{i1}^j\right) &= j && if\ 1 \le j \le c_1 \wedge \mathrm{i} = 1 \\ &= c_1 + j && if\ 1 \le j \le c_2 \wedge \mathrm{i} = 2 \\ &= c_1 + c_2 + j && if\ 1 \le j \le c_3 \wedge \mathrm{i} = 3 \\ &\ldots\ \ldots\ \ldots \\ &= c_1 + c_2 + \cdots + c_{n-1} + j && if\ 1 \le j \le c_n \wedge i = n \end{aligned} \tag{5.6}$$

The population size is considered as 20% of the size of S_2, the set of objects for which the rules are to be generated. So, population size $N = 0.2 \times |S_2|$. Thus, the initial N candidate solutions are randomly taken from the solution space SP as follows:

As the solution space contains all possible candidate solutions representing rules so it contains the elements which are of the form of 1-tuple, 2-tuples, 3-tuples, …, *n*-tuples. Let, B_1, B_2, …, B_n are the set of all 1-tuple, 2-tuples, …, *n*-tuples respectively, were,

$$B_1 = \bigcup_{i=1}^{n} C_i$$

$$B_2 = \bigcup_{i,j=1;i<j}^{n} C_i \times C_j$$

$$B_3 = \bigcup_{i,j,k=1;i<j<k}^{n} C_i \times C_j \times C_k$$

….. ….. ….. ….. ….. ….. ….. …..

$$B_n = C_1 \times C_2 \times \ldots \ldots C_n$$

Thus, the solution space is $SP = B_1 \cup B_2 \cup \ldots \cup B_n$

To apply the discrete PSO, the candidate solutions or particles are encoded by the index values of their components. For example, a particle $\{(C_1 = c_{11}^3), (C_2 = c_{21}^5), (C_3 = c_{31}^3)\}$ in the population is encoded as $\{3, c_1 + 5, c_1 + c_2 + 3\}$.

- **Velocities:**

Discrete PSO algorithm [67] applies the concept of proportional likelihoods. Principally, the idea of proportional likelihood used in the discrete PSO algorithm is more or less similar with the idea of velocity used in the standard PSO algorithm. Each particle *i* is associated with a 2 × *n* array of proportional likelihoods, where 2 and *n* represent number of rows and columns respectively. In this standard proportional likelihood array, each element in the first row of velocity vector *V*(*i*) represents the proportional likelihood, based on which a rule component be selected. The second row of *V*(*i*) has the indices of the rule components, which is associated with the respective proportional likelihoods of the first row of the vector *V*(*i*). There is a one-to-one correspondence between the columns of this array. Initially, for all particles in the population, all elements in the first row of *V*(*i*) are set to 1, e.g., *V*(*i*) = {{1, 1, 1, 1, 1,1,1}, {1, 2, 3, 4, 5, 6, 7}}.

After the initial population of particles is generated, *V*(*i*) is always updated based on *R*(*i*), *B*(*i*) (particle's previous best position) and *G* (global best position).

In addition to *R*(*i*), *B*(*i*) and *G*, three constant updating factors, namely, *a*, *b* and *c* are used to update the proportional likelihoods $v_{(i,d)}$, d^{th} component of the i^{th} particle. These factors determine the strength of the contribution of R_i, B_i and *G* for the adjustment of every coordinate $v_{(i,d)} \in V(i)$.

Parameter values of *a*, *b* and *c* are chosen experimentally for each dataset.

- **Calculation of Fitness Value:**

The main objective of the work is to generate an efficient dynamic rule-based classifier for the incremental dataset.

So, while new data arrives into the system, only the previous rules of the classifier for old dataset together with the new data are used to generate the modified rule-based classifier for the entire dataset without redundantly using the whole dataset.

Actually, the choice of fitness function depends upon the real problem that has to be solved. Every Rule is represented in the form ''IF condition THEN class". Fitness value of each rule depends upon the classification accuracy (considered as confidence value) on new dataset and the similarity of the rule with the existing rule set obtained from old dataset.

i. Confidence Measure*:*

To calculate the confidence value [61, 62] of a rule, let us consider $DS_1 = \{U_1, A, D\}$ and $DS_2 = \{U_2, A, D\}$ as old and new datasets respectively, where $|A|$ is the total number of conditional attributes and D is the decision attribute with different class value. And U_2 can be represented as $U_2 = \{O_1, O_2, \ldots, O_{|U_2|}\}$.

So confidence value of a rule j is $Conf(R_{new}^{j})$ for a decision class $D = d_j$ is calculated using equation (5.3).

Where, $R_{new}^{j} \rightarrow d_j$ for all $j = 1, 2\ldots, m$, all j may not distinct.

ii. Similarity Measure*:*

To calculate the similarity value of a rule with the existing rule set for a particular class, we have considered a set of rules $R_{old} = \{R_{old}^{1}, R_{old}^{2}, \ldots, R_{old}^{n}\}$ of existing classifier trained on old data (DS_1), where, $R_{old}^{1}, R_{old}^{2}, \ldots, R_{old}^{n}$ are the individual n rules in the old rule set R_{old}. Each old rule $R_{old}^{i} = \{r_{i1}^{old}, r_{i2}^{old}, \ldots\ldots, r_{ik}^{old}\}$, where each r_{ij}^{old} is the rule components in i^{th} rule, for $j = 1, 2, \ldots, k$. Let, the possible rule set for a particular class achieved from new dataset DS_2 as $R_{new} = \{R_{new}^{1}, R_{new}^{2}, \ldots, R_{new}^{m}\}$, where, $R_{new}^{1}, R_{new}^{2}, \ldots, R_{new}^{m}$ are the m rules present in the new rule set R_{new}. Each rule R_{new}^{j} is represented as $R_{new}^{j} = \{r_{j_1}^{new}, r_{j_2}^{new}, \ldots\ldots, r_{j_l}^{new}\}$, where, $r_{j_1}^{new}, r_{j_2}^{new}, \ldots, r_{j_l}^{new}$ are the rule components present in j^{th} rule.

The similarity value of new rule j with the old rule set R_{old} is calculated using equation (5.7).

$$S_j^{old} = \frac{1}{|R_{new}^{j}|} \max\left(R_{old}^{t} \cap R_{new}^{j}\right) \text{ where, } 1 \leq t \leq n, \text{ for all } j = 1, 2, \ldots, m. \quad (5.7)$$

iii. Fitness Value:

The *fitness function* is defined in equation (5.8) taking the weighted sum of the confidence value given in equation (5.3) and the similarity measure value computed in equation (5.7).

$$\text{Fitness} = w \times Conf\,(R_{new}^{j}) + (1\text{-}w) S_j^{old} \quad (5.8)$$

Where weight factor *w is* set experimentally for each dataset.

- **Generating New Particles:**

According to the fitness value, new particle will be generated from i^{th} particle by updating its previous best position $B(i)$, global best position G and velocity vector $V(i)$. The proportional likelihood array $V(i)$ is used to generate a new configuration of particle $R(i)$, that is, the particle associated to it. First, for a particle with velocity vector $V(i)$, all indices present in $R(i)$ have their corresponding proportional likelihood increased by '*a*'. Similarly, all indices present in $B(i)$ and G have their corresponding proportional likelihood increased by '*b*' and '*c*' respectively. Now the velocity vector $V(i)$ for each particle i is normalized [29] and sorted in non-increasing order of values in its first row. That is, the elements in the first row of the array are ranked in a decreasing order of value and the indices of the rule components in the second row of $V(i)$ follow their respective proportional likelihoods. Now if the i^{th} particle has p number of components, then first p indices would be selected from $V(i)$ which gives the new position of i^{th} particle. In this way new particles are formed in the search space generation wise for searching the optimal solutions of the problem.

5.3.2 Proposed IPSO Algorithm:

The IPSO algorithm is summarized below:

Algorithm: IPSO (DS_1, DS_2, R)

Inputs: Classification rule set R_1 of existing decision subsystem $DS_1 = (U_1, A, D)$, newly arrived decision subsystem $DS_2 = (U_2, A, D)$, and *swarm_size*

Outputs: Final Classification Rule set R of dataset $DS_1 \cup DS_2$

Begin

Set *Number_of_runs* = 50

for each *Number_of_runs* do

for each class d in D of new dataset DS_2 do

Generate initial swarm of size *swarm_size* using equation (5.5) & (5.6).

Initialize velocity array of each particle.

Repeat

for each particle in the swarm do

Calculate fitness value using equation (5.8).

pbest = current best position of the particle.

gbest = global best position of all particles.

Update velocity of the particle using *pbest* and *gbest*.

Replace old particle by new particle.

end-for

Until termination criteria is met.

Insert *gbest* into *R*.

end-for

end-for

Return (*R*)

End

5.3.3 Results of the IPSO Method:

To measure the performance of the proposed incremental classifier on benchmark experimental data, following experiments are carried out and results are given accordingly. The specification of the computer in these experiments are, Computer Model:

ACER emachines D725; CPU: Pentium(R) Dual-Core CPU T4400 @ 2.20GHz × 2; Memory: 1GB; OS: Ubuntu 12.04 LTS - 32 bit. Java is used as a programming Language for implementation of the work. Dataset is randomly (with uniform class distribution) divided into 3 parts where each part consists of 60%, 20% and 20% of whole dataset respectively.

The former 60 percent objects of each data set is used as old data, and the second part of 20 percent objects are considered as new data available after certain amount of time. And the rest 20% data is considered as test data to measure the performance of IPSO [63]. Firstly, static PSO is used to generate rule set for the original decision table or old data. Secondly, based on previous results, the proposed IPSO algorithm [63] is used to generate the rule set using new data and decision table for old data.

Finally, these rule sets have been applied on the test data and classification accuracy is measured and compared with the results generated by other standard state of the art classifiers.

a. Parameter Setup and Preprocessing:

The parameters used in IPSO are shown in Table 5.12.

These parameters are selected after several test evaluation of the proposed algorithm until reaches to the best configuration in terms of the quality of solutions.

Table 5.12: PSO parameter setting

Input parameters	Applied methods	Termination criteria	No. of independent run
• a = 0.12 (weight for individual particle) • b = 0.14 (weight for individual best particle) • c = 0.16 (weight for global best particle) • Population Size: 100	• Selection type: best • Replace if better gbest: true • Replace if better pbest: true • Velocity updation: true	Search stops in one run when the average fitness of a swarm does not change for 2 consecutive generations.	50

b. Performance Evaluation of the Classifier:

To evaluate the performance of the proposed IPSO method, it has been experimented on experimental benchmark datasets described in section 2.2.

Datasets are divided into old training set, new training set and test sets as mentioned in section 5.3.3.1 and different classifiers are trained and evaluated.

Table 5.13 shows the classification accuracies achieved by proposed IPSO method [63] and some popular state-of-the-art classifiers such as NB [7], SVM [6], KNN [37], Bagging [191], J48 [5], MLP [3], and one GA based incremental classifier[250] named as (IGA), where the best results are marked by bold font.

Table 5.13: Classifiers performances on experimental datasets

Dataset	Classification methods								
	NB	**SVM**	**KNN**	**Bagging**	**J48**	**MLP**	**Static PSO**	**IGA [250]**	**IPSO**
Wine	99.75	99.28	97.45	96.66	96.66	98.90	94.76	96.75	**99.88**
Heart	85.23	87.20	84.80	86.43	84.32	84.26	85.26	86.95	**87.23**
Glass	67.98	75.23	74.98	75.76	74.34	74.98	74.09	74.54	**75.98**
Zoo	95.59	95.54	94.46	94.46	93.48	95.26	94.45	94.67	**95.87**
Dermatology	95.05	**95.90**	95.87	94.23	95.55	94.00	94.01	93.67	94.95
Mushroom	96.34	98.97	98.30	99.98	99.98	97.87	96.97	98.76	**99.05**
Coil20	83.13	80.00	81.77	82.44	84.29	84.33	84.31	83.01	**84.95**
Orl	60.01	59.13	60.17	61.23	61.34	60.91	61.90	60.23	**62.02**
Allaml	83.43	83.67	84.32	84.21	83.70	83.96	83.50	83.34	**84.90**
Leukemia	87.77	87.98	86.97	86.90	88.90	88.09	87.08	87.98	**88.55**

All the standard existing classifiers are run in static environment where whole dataset is considered at once and 10-fold cross validation technique is used for measuring the accuracies.

Only, the IGA [250] and the proposed IPSO classifier [63] are run in dynamic environment. Still, the accuracy of IPSO on wine dataset is nearly about 100%, which is achieved only by NB classifier.

Almost for all the datasets, except Dermatology, the proposed method gives the best result.

To judge the classifier, other than classification accuracy, some statistical measurements [37, 38], given in Equation (2.25) to (2.28) are also performed and the results for the classifiers are listed in Table 5.14.

These parameter values are calculated for all the standard and proposed classifiers for all experimental benchmark datasets.

Table 5.14: Statistical measures of IPSO and standard classification methods

Dataset	Classifier	Recall	Fall_out	Specificity	F1_Score
Wine	NB	**0.99**	**0.01**	0.98	0.98
	SVM	**0.99**	**0.01**	0.98	**0.99**
	KNN	0.97	0.03	0.96	0.97
	Bagging	0.97	0.03	0.97	0.96

Dataset	Classifier	Recall	Fall_out	Specificity	F1_Score
	J48	0.97	0.03	0.96	0.97
	MLP	**0.99**	0.01	**0.99**	**0.99**
	Static PSO	0.95	0.04	0.95	0.95
	IGA	0.97	0.03	0.97	0.97
	IPSO	**0.99**	**0.01**	**0.99**	**0.99**
Heart	NB	0.85	0.15	0.85	0.84
	SVM	**0.87**	0.14	0.86	**0.87**
	KNN	0.85	0.15	0.85	0.84
	Bagging	0.86	0.14	0.85	0.86
	J48	0.84	**0.13**	0.84	0.83
	MLP	0.84	0.14	0.84	0.84
	Static PSO	0.85	0.14	0.84	0.85
	IGA	**0.87**	**0.13**	**0.87**	0.86
	IPSO	**0.87**	**0.13**	0.86	**0.87**
Glass	NB	0.68	0.32	0.67	0.68
	SVM	0.75	0.25	0.75	0.74
	KNN	0.75	0.25	0.76	0.75
	Bagging	0.76	0.24	0.75	0.76
	J48	0.74	0.25	0.73	0.73
	MLP	0.75	0.25	0.75	0.75
	Static PSO	0.74	0.26	0.74	0.73
	IGA	**0.77**	0.23	0.76	**0.77**
	IPSO	0.76	**0.23**	**0.76**	0.76
Zoo	NB	**0.96**	0.04	0.95	**0.96**
	SVM	**0.96**	0.04	**0.96**	0.95
	KNN	0.94	0.06	0.93	0.94
	Bagging	0.94	0.06	0.93	0.93
	J48	0.93	0.07	0.93	0.92
	MLP	0.95	0.95	0.94	0.95
	Static PSO	0.94	**0.96**	0.94	0.93
	IGA	0.95	0.95	0.94	0.95
	IPSO	**0.96**	0.95	**0.96**	**0.96**
	NB	0.95	0.05	0.95	0.95

Dataset	Classifier	Recall	Fall_out	Specificity	F1_Score
Dermatology	SVM	**0.96**	**0.04**	**0.96**	**0.96**
	KNN	**0.96**	**0.04**	0.95	**0.96**
	Bagging	0.94	0.05	0.93	0.94
	J48	**0.96**	**0.04**	0.96	0.95
	MLP	0.94	0.06	0.93	0.94
	Static PSO	0.94	0.06	0.93	0.94
	IGA	**0.96**	**0.04**	**0.96**	**0.96**
	IPSO	0.95	0.05	0.95	0.95
Mushroom	NB	0.96	0.04	0.96	0.96
	SVM	**0.99**	**0.01**	**0.99**	**0.99**
	KNN	0.98	0.02	0.97	0.98
	Bagging	**0.99**	**0.01**	0.98	**0.99**
	J48	**0.99**	**0.01**	0.98	0.98
	MLP	0.98	0.02	0.98	0.98
	Static PSO	0.97	0.03	0.96	0.97
	IGA	0.98	0.02	0.98	0.97
	IPSO	**0.99**	**0.01**	**0.99**	**0.99**
Coil20	NB	0.83	0.16	0.82	0.83
	SVM	0.80	0.20	0.80	0.80
	KNN	0.82	0.18	0.81	0.82
	Bagging	0.82	0.18	0.82	0.81
	J48	0.84	0.16	0.83	0.84
	MLP	0.84	0.16	0.84	0.84
	Static PSO	0.84	0.15	0.84	0.83
	IGA	0.83	0.17	0.82	0.83
	IPSO	**0.85**	**0.15**	**0.85**	**0.85**
Orl	NB	0.60	**0.37**	0.60	0.60
	SVM	0.59	0.40	0.58	0.59
	KNN	0.60	0.40	0.60	0.59
	Bagging	0.61	0.38	0.60	**0.61**
	J48	0.61	0.39	0.60	0.60
	MLP	0.61	0.38	0.60	**0.61**
	Static PSO	**0.62**	0.38	**0.62**	**0.61**

Dataset	Classifier	Recall	Fall_out	Specificity	F1_Score
	IGA	0.60	0.39	0.60	0.60
	IPSO	**0.62**	**0.37**	**0.62**	**0.61**
Allaml	NB	0.83	0.17	0.83	0.84
	SVM	0.84	0.16	0.84	0.83
	KNN	0.84	0.16	0.85	0.84
	Bagging	0.84	0.16	0.83	0.83
	J48	0.84	0.16	0.84	0.83
	MLP	0.84	0.16	0.83	0.84
	Static PSO	0.84	0.16	0.84	0.84
	IGA	0.83	0.17	0.83	0.82
	IPSO	**0.85**	**0.15**	**0.85**	**0.85**
Leukemia	NB	0.88	0.12	0.88	0.87
	SVM	0.88	0.12	**0.89**	0.88
	KNN	0.87	0.13	0.88	0.87
	Bagging	0.87	0.13	0.87	0.87
	J48	**0.89**	**0.11**	0.88	**0.89**
	MLP	0.88	0.12	0.87	0.88
	Static PSO	0.87	0.13	0.86	0.86
	IGA	0.88	0.12	0.87	0.87
	IPSO	**0.89**	**0.11**	**0.89**	**0.89**

From Table 5.14, it is seen that proposed IPSO method works better than static PSO and most of the standard existing classifiers.

c. Comparison based on Statistical Performance: To test the statistical significance of the IPSO method, a statistical analysis using Wilcoxon's rank sum test [39], is done similar to ECS method describing section 5.2.2 and the test results are given in Table 5.15.

Table 5.15: Wilcoxon's Rank Sum Test Results of IPSO with other algorithms

SV	IPSO	SVM	NB	KNN	J4.8	Bagging	MLP	StaticPSO	IGA
p-values	NA	1.7877e-04	1.3587e-04	1.8697e-04	1.4756e-04	1.7387e-04	1.7688e-04	1.7318e-04	1.7718e-04
h-values	NA	1	1	1	1	1	1	1	1

Remark: *P-value* of the test is returned as a positive scalar from 0 to 1 where, p is the probability of observing a test statistic as or very extreme than the experimental value under

the null hypothesis. Here, null hypothesis is stated as "X and Y both samples belong to same population and there is no statistical significance of the proposed algorithm over standard algorithm".

Result of the hypothesis test.

- If $h = 1$, this indicates rejection of the null hypothesis.
- If $h = 0$, this indicates a failure to reject the null hypothesis.

For each pair of tests, the null hypothesis is rejected (as per Table 5.15). Therefore, for each of these cases the test shows that the proposed method is statistically significant over the standard classification algorithms.

5.4 Summary:

In recent era of big data, lots of data are being generated in every moment and at the same time data are not very structured. This has inspired the researchers for developing many classification algorithms to analyze the static and dynamic or time variant data. Classifier construction for static and incremental data is one of the major issues in this chapter. In the chapter, the classification rules are generated using decision matrix approach (CGRG) from an important informative feature subset to classify objects with high classification accuracy. But the single classifier system is not always the universal learner for different data mining job, in that case ensemble classifier system improves the performance over single classifier. So, in the chapter an ensemble classifier (ECS) is also designed based on the classifiers generated from the important feature subset obtained using single objective genetic algorithm. The objective of developing ECS is to maximize the classification accuracy, as it is the main target of a classifier. Then, to handle incremental data a novel incremental classifier (IPSO) has been designed using PSO algorithm to achieve higher classification accuracy. IPSO method uses new dataset and the knowledge from the existing data to develop the classifier with higher classification accuracy with a reduced training time. The statistical analysis is done for all the proposed and existing state of the art single classifier, ensemble classifier and incremental classifier systems to express the effectiveness of the proposed methods. Test results confirm that all the proposed classification methods are statistically significant

.

ISBN: 978-93-90847-37-2

Chapter 6

Application of Data Mining Techniques for the Designing of a Predictive Model in the Field of Agriculture

6.1 Introduction:

Due to the variety of crops and their associated diseases, application of modern technologies in the field of agriculture is still in research stage, which needs detail investigation for designing and developing an automated prediction model to classify the disease for crops. Therefore, with the help of modern tools and technologies and computing frameworks, researchers are trying to develop an efficient and cost-effective automated system to identify the crop diseases.

The system actually guides the farmers by detecting diseases efficiently, so that appropriate pesticides with accurate dosage can be given timely to increase their profit and at the same time it saves the environment. Due to the change of characteristics of the diseases in change of climate, biological and geographical factors, new disease data are always added with the existing data in an incremental manner. This has inspired to develop intelligent classification system for crop disease prediction in dynamic environment. This chapter demonstrates the application of the developed data mining techniques discussed in the book for developing an efficient, intelligent, and integrated classification system for prediction of different rice diseases both in static and dynamic environment.

Rice is an important crop worldwide and near about fifty percent of world population depends on rice as their main food. For more production of rice, losses may be reduced by using different pesticides but at the same time the cost of production increases and in addition food quality degrades. It also creates bad effect on the environment [252] as well. So instead of using those pesticides and to protect the crop, researchers are trying to develop some sustainable farming practice so that diseases and pest management can be controlled effectively by detecting the crop diseases in a timely manner. Detection of crop diseases in time at the fields is critical for precision on-farm disease management [253]. Faster development of the modern digital devices and various computational tools and techniques attract researchers for automatic detection of rice diseases [254]. Actually, in earlier days, farmers were identifying the diseases by observing the abnormalities occur in the infected plants in the field. Based on their experience they detect the diseases. General abnormalities like morphological changes, abnormal growth of the plant, and presence of lesions and formation of spots in the plant are seen due to various diseases. Major drawbacks were in the farmer's traditional methods like dependency on the observer's, detailed anomalies are often over looked in bare eyes and finally the success depends on the experience of the observers. In addition, uses of genetically modified seeds or change in climate bring changes in the appearance of symptoms of diseases, which causes the detection process very difficult.

Sustainable farming [255] is that one which can handle such difficulties along with global warming [256], very efficiently. As a result, it enhances environmental quality, proper use of the non-renewable resources and maintains the economic feasibility of farm operation. Sustainable farming leads to development of a cost effective, automated system for accurate detection of the diseases in a timely manner. Therefore, with the help of modern tools and technologies and computing frameworks, researchers are trying to develop an efficient and cost-effective automated system to identify the crop diseases. The system guides the farmers by detecting diseases earlier, so that appropriate pesticides with accurate dosage can be given timely to increase their profit and at the same time it saves the environment.

Various types of modern tools and technologies such as remote sensing [257], image processing [258] and soft computing techniques [259] have recently been investigated towards the development of disease detection system. Remote sensing technique is used to detect the infected plants through quantitative analysis of spectral differences. Presences of diseases have been identified based on the observations that infected plants behave in a different way in spectral reflectance and thermal emission as compared to the fit ones [260].

However, all these methods either detect the presence of the diseases and/or measure the degree of the diseases but not able to classify the diseases. There are many research proposals to develop expert systems [261-264] for classification and to detect the types of diseases in the field of agriculture. Such systems [262, 263] take user input about the infected plant to classify the crop diseases. These systems actually suffer from the biasness of the observers. To overcome these limitations, scientists in [265], extracts automatically the information related to disease symptoms or features to classify the diseases. The system entirely depends on the extracted feature information and a classifier is developed to predict the diseases. Though the scientists in [265] extracted many relevant features, but the heuristic applied for selecting the optimal feature subset allows some irrelevant features in the feature set. Also, the algorithm cannot be applied in dynamic environment where new disease data are added in incremental way to the existing data.

This chapter describes the development of automated and intelligent disease classification systems for prediction of rice diseases in the static and dynamic environment. Rice disease dataset [34] is prepared from 500 rice disease images having three disease classes such as *Leaf Brown Spot, Rice Blast and Sheath Rot* with total 37 extracted features discussed in the following sections. Experiments on the rice disease dataset demonstrate that the integrated methods obtain good results with fewer features, fast computation, and higher classification accuracies in comparison with other state-of- the- art classifiers.

Before addressing the issues of developing automated rice diseases classification system, this chapter provides brief introduction of very common rice diseases considered for experiments in section 6.2. Section 6.3 discusses the development of rice disease classification system and finally, summary of the chapter is given in section 6.4.

6.2 Rice Diseases:

Mainly bacteria, fungus, and virus cause rice diseases. The rice plant has four parts namely roots, stem, leaves and panicle.

To detect the diseases, locations of infection in different parts of the rice plant are to be diagnosed properly. Three different kinds of rice plant diseases, considered for the work are discussed here.

6.2.1 Leaf Brown Spot:

Fungus Bipolarisoryza is responsible for the disease *Brown spot* which results in both quantitative and qualitative losses [34, 266]. Standard *brown spot* symptoms are seen at tailoring phase and beyond. Circular to oval shaped small foliar lesions are found with light brown to gray at center and reddish-brown color at margin.

6.2.2 Rice Blast:

Rice blast [34, 266] is one of the most significant diseases of rice. Fungus Magnaportheoryzae is responsible for the Rice Blast disease. Generally, lesions with brown color are found on the leaves and it may enlarge and coalesce to destroy the entire leaves. At the beginning, white to grayish green circular lesions with dark green borders are found on the leaves.

Finally at the later stage the shape of the lesion's changes to elliptical or spindle shape with more or less pointed ends.

6.2.3 Sheath Rot:

Pathogen Sarocladiumoryzae is responsible for the rice disease called Sheath rot and found by [34, 266]. Rotting in leaf sheath affects the young panicles. It is seen that the lesions start as like irregular spots with variation in color from gray to light brown at centers, surrounded by distinct dark reddish brown margins [34]. As the disease grows, the lesions enlarge and coalesce and spread to most of the leaf sheath. Lesions may have diffuse reddish-brown discolorations in the sheath. A profuse whitish dusty growth is generally seen within the infected sheaths; still the leaf sheath appears normal from the external part.

The main goal of the development of rice disease classification system is that it identifies and classifies the rice diseases automatically. In this chapter, this problem has been tackled by developing automated classification system for static dataset as well as for incremental dataset, where intelligent techniques proposed in the thesis, are applied to predict the rice diseases.

The work is divided mainly in two parts namely, rice disease detection and classification of rice diseases. In disease detection task, at first features responsible for diseases are extracted from the diseased portion of the rice images using various feature extraction techniques [34].

As all the extracted features are not significant and presence of redundant features affect classification accuracy and increases complexity of the system, so at first important and relevant features are selected from the extracted features using proposed comparatively more effective feature selection methods [44, 48], discussed in chapter 3 and chapter 4 of the work. Finally, classification rules are generated from the reduced rice disease dataset using the proposed classification methods [58, 60, 63] discussed in chapter 5.

6.3 Development of Rice Disease Classification System:

The main goal of the development of rice disease classification system is that it identifies and classifies the rice diseases automatically. In this chapter, this problem has been tackled by developing automated classification system for static dataset as well as for incremental dataset, where intelligent techniques proposed in the thesis, are applied to predict the rice diseases. The work is divided mainly in two parts namely, rice disease detection and classification of rice diseases. In disease detection task, at first features responsible for diseases are extracted from the diseased portion of the rice images using various feature extraction techniques [34]. As all the extracted features are not significant and presence of redundant features affect classification accuracy and increases complexity of the system, so at first important and relevant features are selected from the extracted features using proposed comparatively more effective feature selection methods [44, 48], discussed in chapter 3 and chapter 4 of the work. Finally, classification rules are generated from the reduced rice disease dataset using the proposed classification methods [58, 60, 63] discussed in chapter 5.

6.3.1 Feature Extraction:

To develop an automated classification system, a set of features appearing in the infected rice plant image have been extracted and clubbed with some common features to evaluate performance of the system. Different types of diseases infect rice plants and different symptoms are developed. Based on the symptoms, the diseases are classified manually in earlier days where observer's biasness was associated. So, the focus is on the development of an expert system to classify the rice diseases properly. To achieve this, feature extraction is needed from the rice disease images. Feature extraction mainly deals with detection and localization of particular image patterns representing important features in the image. To automate the classification process, symptoms are mapped into features and features are extracted from the infected regions of the plant by designing efficient algorithms [34, 265, 267-269]. However, importance of the features is application dependent and classification performance of the system varies with the varied selected feature subset. Therefore, feature selection is a very important method to develop a disease classification system.

It is observed that, when a brown spot infects a plant, small circular to oval shaped infected regions is generated with light brown to gray at center and reddish brown at margin. Rice Blast generates elliptical or spindle-shape and more or less pointed ends structures with gray to assay centers surrounded by yellow to dark brown margins. Therefore, colour and shape features are very important and play a crucial role for detecting the diseases. Not only colour and the shape, position of the infection is used for disease identification as brown spot and leaf blast disease generally occurs in the leaf blade. Diseases created by various pathogens produce different type of textures in the infected section of the plant, which is different from the other non-infected part of the plant. So, texture features are also important. In the work, the considered features are (i) colour features (ii) shape-based features (iii) position feature and (iv) texture features to classify different types of rice diseases. Fermi energy-based region detection algorithm [265] is used for image segmentation that separates the infected and non-infected regions accurately. In the work, feature extraction method used in [34, 265, 267-269] is applied to extract important features from the rice disease images.

a. Colour Feature:

Colour based features are very important to automate image-based infection detection and classification processes. It is also observed that, having different nutrient level of the soil different environmental condition and with the age of the plant colour of different regions of the spot varies even when attacked by a particular disease. This problem increases complexity of classifying diseases.

The used method [34] handles the problem by measuring the change in colour of the infected region or spot with respect to the background colour of the image.

Thus, a new set of colour features consisting of mean (*MN*) and standard deviation (*SD*) of the background, the spot as well as the change of color of the infected region with respect to the background region in three classical *R* (Red), *G*(Green), and *B* (Blue) planes are considered as final color features.

Here 18 color features are extracted from the rice disease data and following are the considered color features given in Table 6.1.

Table 6.1: Color features and Abbreviations

Color feature Description	Abbreviation
SD of spot pixels in *R* plane	*SP_ SD_R*
SD of spot pixels in *G* plane	*SP_ SD_G*
SD of spot pixels in *B* plane	*SP_ SD_B*
SD of background pixels in *R* plane	*BC_ SD_R*
SD of background pixels in *G* plane	*BC_ SD_G*
SD of background pixels in *B* plane	*BC_ SD_B*
MN of spot pixels in *R* plane	*SP_ MN_R*
MN of spot pixels in *G* plane	*SP_ MN_G*
MN of spot pixels in *B* plane	*SP_ MN_B*
MN of background pixels in *R* plane	*BC_ MN_R*
MN of background pixels in *G* plane	*BC_ MN_G*
MN of background pixels in *B* plane	*BC_ MN_B*
Ratio of *MN* of spot pixels to *MN* of background pixels in *R* plane	*SP_BC_MN_R*
Ratio of *MN* of spot pixels to *MN* of background pixels in *G* plane	*SP_BC_MN_G*
Ratio of *MN* of spot pixels to *MN* of background pixels in *B* plane	*SP_BC_MN_B*
Ratio of *SD* of spot pixels to *SD* of background pixels in *R* plane	*SP_BC_SD_R*
Ratio of *SD* of spot pixels to *SD* of background pixels in *G* plane	*SP_BC_SD_G*
Ratio of *SD* of spot pixels to *SD* of background pixels in *B* plane	*SP_BC_SD_B*

b. Shape Based Features and Position Feature:

After inspecting large number of samples of rice diseased images, it has been observed that spots created by different pathogens have varied shapes caused due to severity and nature of infection. Different features [34] including area (*AR*), perimeter (*P*), pointed end (*PEN*), Area-discrepancy (*AD*), best-matched primitive shapes (*BMPS*), Aspect-ratio (*ASR*) and different momentums (Φ1- Φ7) of infected region are extracted as shape-based features for disease classification. Position of infection (*PSI*) [265] is also computed and considered as one of the important features of disease images. Here, total **13** shape-based features and 1 position feature are extracted from the diseased images.

c. Texture Feature:

Texture feature is an essential property of any plant image, which is used for capturing the changes occur due to infection in the plant [34]. In spite of availability of different texture features, only the statistical texture features are considered as they are used worldwide, easy to understand and fast in computation.

Statistical texture features are generally calculated based on the co-occurrence matrix [269] of the image. In the work, gray level co-occurrence matrix (GLCM) [269] is used for texture feature extraction purpose, which is the oldest and efficient matrix proposed in [45]. Here, total **5** features are extracted, such as Energy (EG) [34], Entropy (ET) [34], Contrast (CT) [34], Homogeneity (HG) [34], and Co-relation (CR) [34].

Original rice disease dataset is prepared from 500 infected rice plant images having 37 features.

6.3.2 Feature Selection and Classification Analysis:

All the extracted features are not significant, and presence of redundant features affect accuracy and increases complexity of the classification system. Therefore, important and relevant features are selected from the extracted features and a rule-based classification model is developed using the selected features. Basically, a classification rule has two components, antecedent and consequent. Antecedent consists of literal or conjunction of literals where consequent represents the actual class label. A literal l is a (attribute, value) pair in the form (CA_i, v), where CA_i is a conditional attribute and v is the value of attribute CA_i in the decision system DS. An object ob satisfies a literal $l = (CA_i, v)$ if and only if $CA_i(ob) = v$, where $CA_i(ob)$ denotes the i^{th} attribute value of object ob. Classifier construction depicts extraction of interesting patterns from the large repository of data and predicts the future trends based on the existing patterns.

a. Feature Selection and Classification Rule Generation in Static Environment:

This section demonstrates an application of the proposed feature selection and classification methods to generate classification rule set for rice disease detection in static environment. As a first application, a single classifier CGRG [58] discussed in chapter 5 is used to generate classification rules using the single feature subset selected by GRG method [44].

As a second application, an ensemble classifier ECS method [60] discussed in chapter 5 is used to generate another classification system for the rice disease dataset based on the multiple feature subsets selected by MRG method [48] discussed in chapter 3. These two methods are used, as they are superior to other classification and feature selection methods proposed in the thesis.

i. Rice Disease Classification Using the Single Selected Feature Subset:

The classifier is an essential tool for predicting the class level of unknown samples in the dataset. In the work, a single classifier CGRG [58] discussed in chapter 5 is used to generate classification rules by using the feature subset selected by GRG method [44], discussed in chapter 3, to predict the actual class of the rice disease images.

- **GRG Method in Feature Selection:**

GRG method [44] is a single feature subset selection method which uses the concepts of Rough Set Theory [17-20] and Graph Theory [21] to select the most important feature subset from a dataset. The detail method is discussed in section 3.2.2. The GRG method selects 17 features from 37 extracted rice disease features, listed in Table 6.2.

Table 6.2: Selected features by GRG method

Features	*BC_ MN_G, BC_ SD_R, BC_ SD_G, Φ3, SP_ MN_B, BC_ SD_R, SP_ SD_G, SP_ MN_R, SP_BC_SD_R, Φ5, AD, BMPS, PSI, EG, Φ2, ET, CR*

ii. Classification using CGRG Method:

An automated disease classification system is developed using these 17 selected rice disease features using CGRG method [58]. In CGRG, the concept of decision matrix [59] based on Rough Set Theory is used for generation of classification rule set from the reduced rice disease dataset.

The detail of CGRG method is discussed in section 5.2.1. CGRG method [58] generates classification rule set from the reduced dataset and the classification accuracies are measured and compared with state-of-the-art classification methods like NB [7], SVM [6], KNN [37], Bagging [191] J48 [5], and MLP [3] and one existing integrated classification method Phadikar, et al. [265] named as PRG method, as listed in Table 6.3.

Table 6.3: Performance of CGRG method on rice disease data

Classifier	Accuracy (%)	Classifier	Accuracy (%)
NB	89.23	KNN	87.95
SVM	**89.59**	Bagging	85.54
J48	85.67	PRG	87.88

Classifier	Accuracy (%)	Classifier	Accuracy (%)
MLP	86.90	CGRG	89.57

Experimental results show that CGRG method produces satisfactory result in terms of classification accuracy. To judge the performance of the classifier, other than classification accuracy, some statistical measurements [37, 38] given in Equation (2.25) to (2.28) are also performed and the results for the classifiers are listed in Table 6.4.

Table 6.4: Statistical measure of CGRG on rice disease Data

Dataset	Classifier	Recall	Fall_out	Precision	F_Measure
rice disease dataset	NB	0.89	0.023	0.88	0.89
	SVM	0.90	0.023	0.90	0.90
	KNN	0.87	0.014	0.87	0.88
	Bagging	0.86	0.003	0.86	0.87
	J48	0.87	0.201	0.87	0.87
	MLP	0.89	0.025	0.88	0.89
	PRG	0.87	0.12	0.87	0.88
	CGRG	**0.90**	**0.003**	**0.90**	**0.90**

Experimental results show the effectiveness of CGRG method in rice disease classification based on classification accuracy and statistical parameter values.

b. Rice Disease Classification Using Multiple Selected Feature Subsets:

A set of feature subsets rather than a single feature subset is in general more useful for identifying the class level of unknown samples.

Therefore, combination of multiple classifiers is more powerful than a single one in terms of classification accuracy. In this section, the developed ensemble classifier ECS [60] discussed in chapter 5 is applied on rice disease dataset to classify the diseases and identify the class of unknown samples.

Each classifier of ECS is constructed using a feature subset of multiple feature subsets generated by MRG algorithm [48] described in chapter 3.

i. MRG Method in Feature Selection:

The MRG method selects the multiple feature subsets from a dataset using the concepts of indiscernibility relation of rough set theory [17-20], graph theory [21] and clustering algorithm [127, 130].

The detail method is given in section 3.3.2. The method selects 8 feature subsets from the rice disease dataset [34], listed in Table 6.5.

Table 6.5: Selected features by GRG method

Feature subset 1	*BC_ MN_R, BC_ SD_G, BC_ SD_G, Φ2, SP_ MN_B, BC_ SD_R, SP_ SD_R, SP_ MN_R, SP_BC_SD_B, Φ5, AD, BMPS, PSI, EG, Φ2, ET*
Feature subset 2	*BC_ MN_B, BC_ SD_R, BC_ SD_G, Φ1, SP_ MN_B, BC_ SD_G, SP_ SD_B, SP_ MN_R, SP_BC_SD_G, AD, BMPS, PSI, EG, Φ2, ET, HG*
Feature subset 3	*BC_ SD_R, BC_ SD_G, Φ3, SP_ MN_B, BC_ SD_R, SP_ SD_G, SP_ MN_R, SP_BC_SD_R, Φ5, AD, BMPS, PSI, EG, Φ2, ET, CR*
Feature subset 4	*BC_ MN_G, BC_ SD_B, BC_ SD_R, Φ4, SP_ MN_B, BC_ SD_G, SP_ SD_G, SP_ MN_R, SP_BC_SD_R, Φ5, AD, BMPS, PSI, EG, Φ2, ET*
Feature subset 5	*BC_ MN_G, BC_ SD_G, Φ3, SP_ MN_B, BC_ SD_R, SP_ SD_G, SP_ MN_B, SP_BC_SD_R, Φ5, AD, BMPS, PSI, EG, Φ2, ET, HG*
Feature subset 6	*BC_ MN_G, BC_ SD_R, BC_ SD_G, Φ1, SP_ MN_G, BC_ SD_B, SP_ SD_R, SP_ MN_R, SP_BC_SD_R, Φ4, AD, BMPS, PSI, EG, Φ2, ET*
Feature subset 7	*BC_ SD_B, BC_ SD_R, Φ3, SP_ MN_G, BC_ SD_R, SP_ SD_B, SP_ MN_R, SP_BC_SD_R, Φ5, AD, BMPS, PSI, EG, Φ2, ET, CT*
Feature subset 8	*BC_ MN_G, BC_ SD_B, BC_ SD_G, Φ3, SP_ MN_B, BC_ SD_R, SP_ SD_G, SP_ MN_G, SP_BC_SD_G, Φ5, AD, BMPS, PSI, EG, Φ4, ET*

ii. Ensemble Classification using ECS:

In ECS, classification rules are generated based on the multiple feature subsets generated by MRG algorithm [48], and Genetic Algorithm (GA) [23].

In the method, the best combination of classifiers from eight base classifiers each obtained using one selected feature subset is determined using genetic algorithm.

The algorithm searches a particular combination of classifiers that produces maximum classification accuracy.

The detail method is given in section 5.2.2.

The method applied on rice disease dataset and the accuracy is compared with state of the art classification methods like NB [7], SVM [6], KNN [37], J48 [5], MLP [3], and popular ensemble classification methods, Bagging [191], Boosting [192], the classifiers proposed by Das et al. [241] and Zhang [242], where the last two are named here for reference as EOCDPG and EOCASD respectively, as listed in Table 6.6.

In the experiments, '10-fold cross validation' is used to evaluate classification performance where in each iteration 90% samples (9-fold) are used for training and 10% (1-fold) other samples are used for test purpose.

Table 6.6: Performance of ECS on rice disease data

Classifier	Accuracy (%)	Classifier	Accuracy (%)
NB	89.23	KNN	87.95
SVM	89.59	J48	85.67
MLP	86.90	Boosting	87.23
Bagging	85.54	EOCDPG	89.87
EOCASD	89.76	ECS	**91.06**

Table 6.6 shows that the ECS method gives the best accuracy compared to other standard single and ensemble classifiers.

Some statistical measurements [37, 38] like, Recall (Sensitivity), Fall_out, Specificity and F1_score are also calculated using Equation (2.25) to Equation (2.28) respectively.

The calculated statistical measurements are given in Table 6.7 for rice disease dataset using the considered classifiers. It is observed that, the ECS method gives the best result for rice disease dataset.

Table 6.7: Statistical measures of ECS with standard classifiers

Dataset	Classifier	Recall	Fall_out	Precision	F_Measure
rice disease dataset	NB	0.89	0.023	0.88	0.89
	SVM	0.89	0.023	0.88	0.88
	KNN	0.87	0.014	0.87	0.88
	J48	0.87	0.201	0.87	0.87
	MLP	0.89	0.025	0.88	0.89
	Bagging	0.86	0.003	0.86	0.87
	Boosting	0.87	0.012	0.90	0.90
	EOCDPG	0.90	0.004	0.89	0.89
	EOCASD	0.90	0.005	0.90	0.89
	ECS	**0.91**	**0.003**	**0.91**	**0.91**

b. Feature Selection and Classification Analysis in Dynamic Environment:

As datasets changes with time, it is very time consuming or even infeasible to run repeatedly a knowledge acquisition algorithm.

In dynamic environment, without learning the same classifier for the whole data, the existing classifier and the new group of data are examined to develop an updated classifier, called incremental classifier that reduces the time complexity of the system developed.

In this section, firstly, an incremental feature selection method IFS [57] discussed in chapter 4 is applied on rice disease dataset to select important features and then different classifiers are applied for disease classification. An incremental rice disease classification system IPSO [63] discussed in chapter 5, is also applied for rice disease detection in dynamic environment and observed that the IPSO method provides more accuracy than the existing state-of-the-art classifiers on rice disease dataset.

- **IFS Method of Incremental Feature Selection for Rice Disease Classification:**

The IFS method [57] is an incremental feature selection technique developed using the concept of rough set theory (RST) [17-20] and genetic algorithm (GA) [23, 102]. The method is applied in a regular basis in the dynamic environment after small to moderate volume of data being added into the system and GA is applied not on the whole dataset but only on the new chunk of objects currently enters into the system thus the time complexity of GA, major issue of the algorithm does not affect the IFS method. Here single objective genetic algorithm is proposed by combining multiple criteria for obtaining single optimal solution, i.e., a single feature subset of a decision system which effectively reduces dimensionality of the dataset without sacrificing classification accuracy. The detail method is discussed in chapter 4.2.2.

At first IFS algorithm is applied on the rice disease dataset [34] having 500 objects with 37 extracted features. The algorithm deals with discrete valued dataset and so the continuous dataset is discretized [219] before use of the algorithm. Here, 80% of the dataset is considered as old dataset and remaining 20% as new dataset. IFS algorithm selects 16 features when GA runs initially on old dataset and due to addition of new dataset IFS algorithm selects 14 features based on previous 16 features and new dataset (i.e., dynamic environment). The IFS method selects the feature subset of 14 features, which include different colour, shape, position and texture feature to classify the three disease classes. Table 6.8 shows the list of selected features by IFS method.

Table 6.8: Selected features by IFS method

Feature subset	*BC_ MN_R, BC_ SD_R, BC_ SD_B, SP_ SD_G, SP_ MN_R, SP_BC_SD_G, Φ3, AD, BMPS, PSI, EG, Φ4, CR, CT*

To judge the effectiveness and the efficiency of the proposed IFS method for rice disease classification, the method is compared with common standard static or non-incremental attribute reduction methods such as 'Correlated Feature Subset Evaluator'(CFS) [95], 'Consistency Subset Evaluator'(CON) [213], 'Classical Attribute Reduction based on Shannon's information entropy'(CAR) [214],'Relief-F' [215], Static IFS method (static version of IFS method) and popular incremental algorithms such as IUAARI [50], IUAARS [52], Xu et al. [230], GIARC-L based on complementary entropy [214] and Shu et al. [231].

Results of all the existing and the proposed IFS method are evaluated and compared on the basis of classification accuracies on reduced rice disease dataset by the state-of-the-art classifiers available in weka tool [218].

In the work, considered classifiers are Naïve Bayes (NB) [7], Support vector machine (SVM) [6], K-nearest neighbors K-NN [37], Bagging [191], Tree based classifier (J48) [5], and Multilayer Perceptron (MLP) [3]. SVM is used with RBF Kernel; *K* value of K-NN is set to the square root of sample size of data.

Original number of attributes, number of attributes after applying proposed IFS and existing static feature selection methods and the accuracies (%) of the reduced rice disease dataset by the mentioned classifiers are computed and listed in Table 6.9. Original number of attributes, number of attributes after applying proposed IFS and existing incremental feature selection methods and the accuracies (%) of the reduced rice disease dataset by the mentioned classifiers are computed and listed in Table 6.10.

Table 6.9: Performance of IFS and existing static feature selection methods

Classifier	**rice disease dataset (37)**					
	CFS	**CON**	**CAR**	**Relief-F**	**Static IFS**	**IFS**
	15	15	17	18	17	14
NB	88.23	88.14	87.23	87.52	87.86	**89.23**
SVM	87.56	89.25	88.03	88.11	88.32	**89.35**
KNN	86.32	87.78	86.23	87.32	86.25	**87.95**
J48	84.50	85.50	84.23	84.32	83.92	**85.67**
MLP	81.12	85.62	85.68	85.32	85.05	**86.90**
Bagging	85.40	86.67	86.12	85.23	86.20	**87.80**
Average	85.54	87.25	86.28	86.51	86.28	**87.82**

Table 6.10: Performance of IFS and existing incremental feature selection methods

Classifier	**rice disease dataset (37)**					
	IUAARI	**IUAARS**	**Xu et al.**	**GIARC-L**	**Shu et al.**	**IFS**
	17	15	21	15	16	14
NB	88.23	88.04	87.23	88.14	88.22	**89.23**
SVM	87.65	88.75	88.56	89.05	88.23	**89.35**
KNN	87.32	87.68	86.32	87.60	87.60	**87.95**
J48	85.50	84.60	84.76	85.45	84.93	**85.67**
MLP	85.12	85.70	85.25	85.62	85.91	**86.90**
Bagging	86.70	86.87	86.23	87.10	87.01	**87.80**
Average	86.76	86.95	86.42	87.17	86.97	**87.82**

Experimental results show the efficiency of the proposed IFS method in rice disease classification by generating minimum number of features in the feature subset with the higher classification accuracies by the state-of-the-art existing classifiers in comparison with other methods. As accuracy is not only the measurement of effectiveness of the classifiers, some statistical measurements given in Equation (2.25) to (2.28) are performed and the average results for all six classifiers are listed in Table 6.11.

Table 6.11: Statistical measure of IFS method and related feature selection methods

Dataset	Methods(#features)	Recall	Fall_out	Specificity	F1_Score
rice disease dataset	CFS (15)	0.86	0.12	0.86	0.86
	CON (15)	0.87	0.08	0.87	0.86
	CAR (17)	0.86	0.12	0.86	0.87
	Relief-F (18)	0.87	0.10	0.86	0.87
	Static IFS (17)	0.86	0.05	0.87	0.86
	IUAARI (17)	0.87	0.09	0.86	0.87
	IUAARS (15)	0.87	0.11	0.87	0.86
	Xu et al. (21)	0.86	0.12	0.86	0.87
	GIARC-L (15)	0.87	0.13	0.86	0.86
	Shu et al. (16)	0.87	0.12	0.87	0.87
	IFS (14)	**0.88**	**0.04**	**0.88**	**0.88**

From the Table 6.11 it is seen that the performance of IFS is better than the other static and incremental attribute reduction techniques for rice disease classification in most of the cases.

- **Incremental Classification using IPSO Method:**

In IPSO, the concepts of Particle Swarm Optimization (PSO) technique [67-69] and Association Rule Mining [61, 62] are used to design an incremental rule-based classifier.

The algorithm handles incremental data effectively for upgrading the existing classifier by modifying the existing rule sets whenever new set of data is added with the previous dataset.

The detail IPSO method is discussed in chapter 5.3. Here IPSO method [63] has been applied on the rice disease dataset for generating classification rules to predict the rice diseases of unknown samples.

IPSO is applied on the rice disease dataset [34] of 500 objects and 37 features extracted from the rice disease images of three different classes, listed in Table 6.1. IPSO model is first learned by the 60% of the dataset as training data and initial classification rules are generated. Then from the remaining data, 20% data is used as incremental data and other 20% data is considered as test data to measure the performance of the incremental classification system.

Thus, while incremental data arrives, static PSO method named as PSO on whole data and IPSO method has been applied, where the processes terminate when the value of the average fitness does not change for 2 consecutive generations.

The detail parameter settings of IPSO method have given in Table 5.12. After the training of the classification system, average classification accuracy is measured on test dataset. The accuracies of the proposed IPSO method and PSO method, genetic algorithm based incremental classifier [250] named as IGA, Phadikar, et al. [265] named as PRG method and some state-of-the-art classification methods in weka tool [218] are computed and listed in Table 6.12.

Table 6.12: Performance evaluation of IPSO on rice disease data

Classifier	Accuracy (%)	Classifier	Accuracy (%)
NB	89.23	KNN	87.95
SVM	89.35	Bagging	85.54
J48	85.67	IGA	90.37
MLP	86.90	PSO	91.47
IPSO	**92.02**	PRG	87.88

To judge the classifier, other than classification accuracy, some statistical measurements [37, 38] given in Equation (2.25) to (2.28) be also performed and the results for the classifiers are listed in Table 6.13.

Table 6.13: Statistical measure of IPSO on rice disease data.

Dataset	Classifier	Recall	Fall_out	Precision	F_Measure
rice disease dataset	NB	0.89	0.023	0.88	0.89
	SVM	0.89	0.023	0.88	0.88
	KNN	0.87	0.014	0.87	0.88
	Bagging	0.86	0.023	0.86	0.86
	J48	0.86	**0.003**	0.86	0.87
	MLP	0.87	0.201	0.87	0.87
	PRG	0.88	0.120	0.89	0.88
	IGA	0.90	0.025	0.90	0.89
	PSO	0.91	0.014	0.91	0.90
	IPSO	**0.92**	**0.003**	**0.92**	**0.92**

Experimental results show the effectiveness of the application of proposed IPSO method in rice disease dataset.

ISBN: 978-93-90847-37-2

Chapter 7

Conclusions and Future Research

Data mining has been gaining importance for organizing and summarizing the large datasets in a comprehensive way for data modeling and Knowledge discovery. Data modeling plays an important role for understanding various fact hiding in the large datasets by transforming data into useful knowledge. Data mining research should concentrate on developing scalable algorithms and techniques capable to handling large dimensional data in the static and dynamic environment efficiently. Designing of data mining tools covers a wide spectrum of data analysis methods for discovering intrinsic knowledge that include characterization, discrimination, association, classification, clustering, and prediction of data. The main aim of writing the book is to discuss the effective data mining tools and techniques [1, 2] such as feature selection, classification, and extraction of meaningful information in order to efficient analysis of various benchmark dataset in the static and dynamic environment. The major challenges like high dimensionality, dimensionality reduction, informative feature selection, classification rule generation and ensemble of classifiers have been addressed in the thesis using the concept of Rough Set Theory [17- 20], Graph Theory [21], Genetic Algorithm [23, 102-103], Particle Swarm Optimization [24, 67-69,105-110], and other probabilistic, mathematical, and statistical approaches [25, 26]. Developed algorithms have been applied in various benchmark datasets and rice disease datasets to classify the objects and predict the unknown objects efficiently. The book concludes by summarizing the related works and significant contributions of the author for knowledge discovery in static and dynamic environment, along with the directions of future research.

7.1 Conclusions:

The objective of writing the book is to provide different data analysis methods using data mining techniques in static and dynamic environment for solving different prediction problem including the rice disease analysis.

The significant contributions are to design an integrated system consisting of mainly the following major components: feature selection, and classifier construction in static and dynamic environment. Each component has a distinct set of functionalities and plays a specific role within the system towards achieving goal of data analysis.

To design the classifier for data analysis, components like dimensionality reduction and selection of important features are the prior steps to build the overall system efficiently.

Hence the book is organized with the interconnection of all the components for designing the entire integrated system.

7.1.1 Feature Selection in Static Environment:

Efficient feature selection technique has a great importance to knowledge discovery.

Experimental dataset contains large number of features, many of which are irrelevant for efficient classification of the dataset and as a result classification model including all features degrades accuracy. Therefore, automated discovery of small and informative feature subset is highly desirable. In feature selection literature, various methods [12, 13, 75-79, 86] are provided to filter out the redundant features, but the classification results are not satisfactory.

In the book, many novel and efficient static feature selection methods [43, 44, 47, 48] are discussed in chapter 3 which selects only the important features from the dataset for achieving higher classification accuracy in comparison compared to the existing methods [12, 13, 86]. A feature selection method (SRG) [43] is described to select a single important feature subset for classifying the objects with using the concept of Rough Set Theory. Another graph based novel feature selection method (GRG) [44] has also been discussed to select single important informative feature subset, using Rough Set Theory and minimal spanning tree of graph theory, which classifies the test objects more accurately compared to SRG method. The above mentioned two methods select single feature subset, whereas multiple feature subset selection method (FSBR) [47] has been developed using only Rough Set Theory to select compact feature subsets. Another graph based novel multiple feature subset selection method (MRG) [48] using Rough Set Theory and clustering algorithm has been discussed which provides better classification accuracy compared to FSBR method. To show the effectiveness of the discussed methods, a performance comparison is made between the proposed [43, 44, 47, 48] and the existing state-of-the-art feature selection methods [95, 213, 214, 215, 216, 217]. Statistical analysis is performed for measuring the statistical significance of the proposed methods in comparison with other methods. The comparisons of all the developed methods [43, 44, 47, 48] are also made in Chapter 3 of the book.

7.1.2 Feature Selection in Dynamic Environment:

As per as dynamic environment is concerned, volume of dataset is growing rapidly with respect to time which bring great difficulty to data mining and pattern recognition. As datasets changes with time, it is very time consuming or even infeasible to run repeatedly a knowledge acquisition algorithm. Incremental learning is a technique where the learning process applies after an interval of time using the information extracted by previous run of the process and the new dataset.

Dimension reduction in dynamic environment uses newly generated data together with the information extracted from the previous data to select the important features with respect to whole dataset. As a result, applicability and acceptability of the system increases.

In the book, incremental feature subset selection algorithm is proposed in dynamic environment integrating the concepts of Rough Set Theory and Genetic Algorithm. Two efficient incremental feature selection methods [49, 57] are reported in chapter 4 for finding out the most important feature subset from incremental data. An incremental feature selection method (DRED) [49] is described to select multiple important feature subsets from incremental data for classifying objects with high accuracy using the concept of Rough Set Theory. A genetic algorithm (GA) based group incremental feature selection method (IFS) [57] is proposed in the thesis, where the method selects the features dynamically using the concept of rough set theory and the genetic algorithm.

Here GA [23, 101] is applied only on newly added group of objects of small to moderate size on regular basis so the great issue of using it for its larger complexity may be avoidable in most of the applications. The novelty of the algorithm is that it can select features both in static and dynamic environment and no prior statistical information of the data is required. Algorithms have been applied on benchmark datasets to demonstrate its effectiveness. To judge the efficacy of the developed incremental methods [49, 57], a performance comparison is made between the developed and other standard incremental [50, 52, 214, 230, 231] and non-incremental feature selection methods [95, 213. 214, 215, 216, 217]. Statistical analysis [39] is also performed for measuring the statistical significance of the proposed methods in comparison with other methods.

7.1.3 Classification Analysis:

The book demonstrates the classification analysis for analyzing dataset [2] to predict patterns of the data in static and dynamic environment. There are various static classification methods [14, 15, 80] but none of them can efficiently handle big datasets. Building efficient classifier [14, 15, 80] to extract meaningful knowledge from the huge amount of data is the primary concern of the data mining research community. Feature selection methods discussed in Chapter 3 providing the single and multiple feature subsets relevant for classification purpose are used for developing classifiers.

In Chapter 5 of the book, a classification rule generation method (CGRG) [58] has been discussed based on important informative feature subset identified by GRG method [44] discussed in Chapter 3 to classify objects efficiently. The primary goal of classifier design is to achieve more accurate prediction. It is seen that One of the best performer classifiers is not always suitable for prediction rather ensemble of different classifiers [179] may lead to better classification accuracy. Several methods for constructing ensemble classifier system [179, 185-187] have been developed by researchers, some are general, and some are specific to particular problems. Here, an ensemble classifier (ECS) [60] is constructed to overcome the demerits of some individual base classifiers and increases the overall classification accuracy. This classifier is built based on the reduced dataset obtained from MRG method [48], discussed in Chapter 3, using Genetic Algorithm and the association rule mining technique. The objective of the proposed ensemble classifier [60] is to maximize the classification accuracy. As single classifier system is not always acted as a generalized learner for different data mining problem; the proposed combined classification [60] system applied on various datasets demonstrates better performance over single classifiers.

In the chapter 5 of the book, a classifier for incremental data has also been proposed (IPSO) [63] with an objective to develop a rule based incremental classifier (IPSO) [63] for the incremental datasets. In the method, the incremental classifier is designed with the aim that the number of classification rules will be minimal. In this method, optimized classification rules are generated for the incremental data dynamically using the concept of Association rule mining and PSO algorithm. Here firstly, PSO [67-69] based training process is performed on the existing dataset to find out the initial optimal classification rules for existing dataset. When a new group of data arrives, IPSO is run using existing classifier and new group of data to develop a dynamic classifier. So, the IPSO algorithm analyzes the new dataset in every interval of time and updates the previous knowledge base dynamically with a sufficiently reduced training time.

To judge the effectiveness of the proposed incremental method, a performance comparison is made between the proposed and the other standard incremental and non-incremental classification methods [3, 5, 6, 7, 37, 191, 250] and observed that the proposed method provides a satisfactory result.

The comparisons of all the developed classifiers [58, 60, 63] with some state-of-the-art classifiers are provided in Chapter 5. Statistical analysis is also performed for measuring the statistical significance of the developed methods in comparison with other methods.

7.1.4 Application of the work in the field of Agriculture:

In the chapter 6 of the book application of the developed data mining techniques for feature selection and classification methods in rice disease prediction is addressed. Application of data mining technology in agricultural field for disease prediction is a challenging task due to the wide variation of crops, associated diseases, and dependency on human being to collect information from the field. As day-by-day the characteristics of the diseases change with the time due to changes of climate, biological, and geographical factor new disease data are added with the existing data so to predict the rice diseases in this dynamic environment, an efficient incremental automated intelligent system is necessary. For the rice disease dataset, features based on color, shape, position, and texture are extracted [34, 265, 267-269] from the infected rice plant images [34]. Then the incremental algorithm IFS [57] selects only the important features from the extracted features by removing irrelevant and redundant features for classification of rice diseases. Finally, the rule based incremental classifier IPSO [63] is applied on the reduced data based on the result of IFS method. IPSO method generates a set of optimized classification rule set to classify the different rice diseases with higher classification accuracy in comparison with other existing methods.

7.2 Future Research:

The results presented in this book demonstrate that the developed data mining algorithms for feature selection and classification for experimental datasets is capable of achieving high performance and meet the user requirements both in static and dynamic environment.

However, the developed work has few limitations. This section discusses these limitations and also gives direction for further research work to resolve these issues and enhance the performance. As the developed data mining algorithms mainly concerned with two major components such as feature selection and classification, so the limitations and further enhancement of these components are discussed.

7.2.1 Feature Selection:

In the graph based static feature selection methods [44, 48] only the degree of the nodes is considered for selecting important features while other measures like weight, degree of centrality and so on may be considered to determine the importance of the nodes.

The method [57] proposed for incremental feature selection considers two criteria's for defining a single objective function of GA.

Here, a weight factor *w* is assigned to the fitness function to provide the relative importance during incremental feature selection. The major demerit of giving a threshold value to *w* may be solved using neural network or probability theory for computing and fixing theoretically the value of *w*.

7.2.2 Classification Analysis:

An ensemble classifier [60] is designed for efficient analysis of static data, but recently, in the era of big data, an ensemble classification system for dynamic environment can be developed by involving more than one competing or conflicting objective functions for finding many optimal solutions. The use of semi-supervised machine learning has emerged recently which lies somewhere between supervised and unsupervised, where the class information is learned from the labeled data and the structure of the data from the unlabeled data. The classification algorithms may be used in this framework to produce an efficient and high-performance tool.

Rough Set Theory [17-20] is mostly used to develop the data mining methods for feature selection and classifier construction to handle uncertain and vague data. Furthermore, rough-fuzzy based integrated system [270, 271] can be devised to design personalized classifiers based on selected important features greatly depends on individual perception. Incremental methods used for classifying the different rice diseases may be applied for the prediction of other crop or plant diseases.

ISBN: 978-93-90847-37-2

8. Bibliography

[1] J. Han, M. Kamber, "Data Mining: Concepts and Techniques", Morgan Kaufmann, San Francisco, 2001.

[2] A. Sharma, R. Sharma, V. K. Sharma, V. Shrivatava, "Application of Data Mining – A Survey Paper", International Journal of Computer Science and Information Technologies, Vol. 5 (2), pp. 2023-2025, 2014.

[3] S. K. Pal, S. Mitra, "Multi-Layer perceptron, fuzzy sets and classification", IEEE Trans. Neural Networks, Vol. 3, pp. 683-697, 1992.

[4] J.R. Cano, F. Herrera, M. Lozano, "Strategies for Scaling Up Evolutionary Instance Reduction Algorithms for Data Mining." In: L.C. Jain, A. Ghosh (Eds.) Evolutionary Computation in Data Mining, Springer, pp. 21-39, 2005.

[5] S. Teli, P. Kanikar, "A Survey on Decision Tree Based Approaches in Data Mining", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 5 (4), pp. 613-617, 2015.

[6] I. H. Witten, E. Frank, "Data mining: Practical machine learning tools and techniques", San Francisco: Morgan Kaufmann, 2005.

[7] V. Roth, T. Lange, "Bayesian class discovery in microarray dataset", IEEE Transaction on Biomed Eng., Vol. 51 (5), pp. 707-718, 2004.

[8] P. A. Devijver, J. Kittler, "Pattern Recognition: A Statistical Approach", Englewood Cliffs, NJ: Prentice Hall, 1982.

[9] T. Y. Lin, N. Carcone (Eds.), "Rough Sets and Data Mining: Analysis of Imprecise Data", Kluwer Academic Publishers, 1997**.**

[10] S. Kotsiantis, D. Kanellopoulos, P. Pintelas, "Data Preprocessing for Supervised Learning", International Journal of Computer Science, Vol. 1 (2), pp. 111–117**,** 2006.

[11] C. E. Brodley, M. A. Friedl, "Identifying Mislabeled Training Data", AIR, Vol. 11, pp. 131-167, 1999.

[12] K. Arunasakthi, L. Kamatchi-Priya, "A Review on Linear and Non-linear Dimentionality Reduction Techniques", Machine Learning and Applications: An International Journal (MLAIJ), Vol. 1 (1), pp. 65-76, 2014.

[13] K.Thangavel, A. Pethalakshmi, "Dimensionality reduction based on rough set theory: A Review", Applied Soft Computing, Vol. 9 (1), pp. 1–12, 2009.

[14] M. Dash, H. Liu, "Feature Selection for Classification", Intelligent Data Analysis, pp. 131-156, 1997.

[15] M. Sujatha, S. Prabhakar, G. Lavanya Devi, "A Survey of Classification Techniques in Data Mining", International Journal of Innovations in Engineering and Technology (IJIET), Vol. 2 (4), pp. 86-92, 2013.

[16] L. I. Kuncheva, "Combining Pattern Classifiers, Methods and Algorithms", New York, NY: Wiley Interscience, 2005.

[17] Z. Pawlak, "Rough sets", International journal of information and computer sciences, Vol. 11, pp. 341-356, 1982.

[18] L. Polkowski, "Rough sets: Mathematical foundations", Advances in soft computing, 2002.

[19] Z. Pawlak, "Rough set theory and its applications to data analysis", Cybernetics and systems, Vol. 29, pp. 661-688, 1998.

[20] T. Y. Lin, N. Cercone, (Eds.) "Rough sets and data mining: Analysis of imprecise data", Springer Science & Business Media, 2012.
[21] N. Deo, "Graph Theory with Applications to Engineering and Computer Science", Prentice-Hall of India Pvt., ISBN-81-203-0145-5, 1995.
[22] J. A. Hartigan, "Clustering algorithms", John Wiley & Sons, Inc., New York, NY, USA, 1975.
[23] J. H. Holland, "Genetic algorithms", Sci. Am. Vol. 267 (1), pp. 44-150, 1992.
[24] J. Kennedy, R. C. Eberhart, "Swarm Intelligence", Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.
[25] A. Webb, "Statistical Pattern Recognition", London: Arnold, 1999.
[26] V. Vapnik, "Statistical learning theory", New York: Wiley, 1998.
[27] P. Murphy, W. Aha, "UCI repository of machine learning databases (1996)", http://www.ics.uci.edu/mlearn/MLRepository.html.
[28] A repository for feature selection datasets: http://featureselection.asu.edu/datasets.php.
[29] C. Saranya, G. Manikandan, "A Study on Normalization Techniques for Privacy Preserving Data Mining", International Journal of Engineering and Technology (IJET), Vol. 5 (3), pp. 2701-2704, 2013.
[30] E. Xu, S. Liangshan, R. Yongchang, W. Hao, Q. Feng, "A new Discretization approach of Continuous attributes", Asia-Pacific Conference on Wearable Computing Systems, Vol. 5 (2), pp. 136-138, 2010.
[31] Lu. Yijun, "Concept Hierarchy in Data Mining: Specification, Generation and Implementation", 1998.
[32] T. Mitchell, "Machine Learning", New York: McGraw-Hill, 1997.
[33] C. Giraud-Carrier, "A note on the utility of incremental learning", AI Commun., Vol. 13 (4), pp. 215-223, 2000.
[34] S. Phadikar, "Intelligent Algorithms for Classification of Crop Diseases", PhD thesis, Dept. of Computer Science, IIEST, Shibpur, Howrah, India, 2012.
[35] L. Zadeh, "Fuzzy sets", Information and Control, Vol. 8, pp. 338-353, 1965.
[36] L. Devroye, L. Gyorfi, G. Lugosi, "A Probabilistic Theory of Pattern Recognition", Newyork: Springer-Verlag, 1996.
[37] E. Alpaydin, "Introduction to Machine Learning", PHI, 2010.
[38] T. Fawcett, "An introduction to ROC analysis", Pattern Recognition Letters, Vol. 27, pp. 861-874, 2006.
[39] E. L. Lehmann, J. P. Romano, "Testing Statistical Hypotheses", Springer, Vol. 64 (2), pp. 255-256, 2006.
[40] M. P. Fay, M. A. Proschan, "Wilcoxon-MannWhitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules", Statistics Surveys, Vol. 4, pp. 1-39, 2010.
[41] C. R. Kothari, G. Garg, "Research Methodology: methods and techniques", New Age, Third edition, 2014.
[42] S. C. Gupta, V. K. Kapoor, "Fundamental of Mathematical Statistics", Published by: Sultan Chand & Sons, A.S. Printing Press, India, 1994.
[43] S. Sengupta, A. K. Das, "Single Reduct Generation by Attribute Similarity Measurement based on Relative Indiscernibility", Proceedings of the Second International Conference, Part II. Springer LNICST Series, Vol. 85, pp. 476-487, 2012.

[44] A. K. Das, S. Sengupta, S. Chakrabarty, "Reduct Generation by Formation of Directed Minimal Spanning Tree using Rough Set Theory", Advances in Intelligent and Soft Computing Springer Vol. 132, pp.127-135, 2012.

[45] J. Bang-Jensen, G. Gutin, "Digraphs: Theory, Algorithms and Applications", Springer, Heidelberg, ISBN 1-85233-268-9.

[46] Y. J. Chu, T. H. Liu, "On the shortest arborescence of a directed graph", Science Sinica 14, pp. 1396–1400, 1965.

[47] A. K. Das, S. Chakrabarty, S. Sengupta, "Formation of a Compact Reduct Set Based on Discernibility Relation and Attribute Dependency of Rough Set Theory", Proceedings of the Sixth International Conference on Information Processing, Wireless Network and Computational Intelligence Springer, pp. 253-261, 2012.

[48] S. Sengupta, A. K. Das, "Dimension reduction using clustering algorithm and Rough Set Theory", Proceedings of the third International Conference, SEMCCO 2012, Springer-Verlag Berlin heidelberg, LNCS 7677, pp. 705-712, 2012.

[49] S. Sengupta, A. K. Das, "Reduct generation for the incremental data using Rough Set Theory", Fourth International Conference on Artificial Intelligence, Soft Computing and Applications.Volume Editors: Dhinaharan Nagamalai, Sundarapandian Vaidyanathan, Vol. 4, pp. 291-299, DOI: 10.5121/csit.2014.4529, ISBN: 978-1-921987-22-9, 2014.

[50] M. Yang, "An incremental updating algorithm for attributes reduction based on the improved discernibility matrix", Chinese Journal of Computers, Vol. 30 (5), pp. 815–822, 2007.

[51] G. Bazan, "Dynamic reducts and statistical Inference", Proceedings of the 6th International conference on Information Processing and Management of uncertainity in knowledge based system, pp. 1147-1152, 1996.

[52] L. Guan, "An Incremental Updating Algorithm of Attribute Reduction set in Decision Tables", Proceedings of the 6th IEEE International Conference on Fuzzy Systems and Knowledge Discovery, pp. 421-425, 2009.

[53] F. Hu, G. Wang, H. Huang, Y. Wu, "Incremental attribute reduction based on elementary sets", Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, pp. 185–193, 2005.

[54] Z. Zheng, G. Y. Wang, "RRIA: A Rough Set and Rule Tree Based Incremental knowledge Acquisition Algorithm", Fundamenta Informaticae, Vol. 59, pp. 299- 313. 2004.

[55] H. M. Chen, T. R. Li, D. Ruan, J. H. Lin, C. X. Hu," A Rough-Set Based Incremental Approach for Updating Approximations under Dynamic Maintenance Environments", IEEE Trans. Knowledge and Data Eng., Vol. 25 (2), pp. 274-284, 2013.

[56] W. C. Bang, B. Zeungnam, "New Incremental Learning Algorithm in the Framework of Rough Set Theory", Int'l J. Fuzzy Systems, Vol. 1 (1), pp. 25-36, 1999.

[57] A. K. Das, S. Sengupta, S. Bhattacharyya, "A Group Incremental Feature Selection for Classification using Rough Set Theory based Genetic Algorithm", Applied soft Computing. Elsevier (Revised), 2017.

[58] A. K. Das, S. Sengupta, "Compact Reduct Formation for Classification Rule Set Generation using Rough Set Theory", International Journal of Information Processing (IJIP) Vol. 6 (4), pp. 64-74, 2012.

[59] Z, Wojciech, S, Ning, "Discovering attribute relationships, dependencies and rules by using rough sets", Proceedings of the 28th Annual Hawaii International Conference on System Sciences (HICSS'95). Hawaii. pp. 293-299, 1995.

[60] S. Sengupta, A. K. Das, "An Approach to Development of an Ensemble Classification System", Second IEEE International Conference on Research in Computational intelligence and Communication Networks (ICRCICN), pp. 218-223, 978-1-5090-1047-9/16/$31.00 ©2016 IEEE, 2016.

[61] R. Agrawal, T. lmielinsk, A. Swami, "Mining Association Rules between Sets of Items in Large Databases", Proceedings of the ACM SIGMOD International Conference on the Management of Data, pp. 207 – 216, 1993.

[62] R. Agrawal, R. Srikant, "Fast Algorithms for Mining Association rules", Proc. 20th VLDB conference, Santiago, Chile, 1994.

[63] S. Sengupta, A. K. Das, "Particle Swarm Optimization based incremental classifier design for rice disease prediction", Computers and Electronics in Agriculture, Vol. 140, pp. 443–451, 2017.

[64] W. Ziarko, N. Shan,"A Rough Set-Based Method For Computing All Minimal Deterministic Rules in Attribute-Value Systems", Technical CS-93-02, Department of Computer Science, University of Regina, Canada, 1993.

[65] D. Cheung, S. Lee, B. Kao, "A general Incremental Technique for Mining Discovered Association Rules", in Proc. 5th International Conference on Database System for Advanced Applications, Melbourne, pp. 185-194, 1997.

[66] A. Ulas, M. Semerci, O. T. Yildiz, E. Alpaydin," Incremental construction of classifier and discriminant ensembles", Information Sciences, Vol. 179 (9), pp. 1298-1318, 2009.

[67] Elon S. Correa, Alex A. Freitas, Colin G. Johnson, "A New Discrete Particle Swarm Algorithm Applied to Attribute Selection in a Bioinformatics Data Set", in the proceedings of GECCO'06, Seattle, Washington, USA, pp. 35-42, 2006.

[68] S. Sengupta, A. K. Das, "Optimal rule set generation using pso algorithm", Fourth International Conference on Artificial Intelligence, Soft Computing and Applications Volume Editors: Dhinaharan Nagamalai, Sundarapandian Vaidyanathan Vol. 4, pp. 301-306, DOI: 10.5121/csit.2014.4530, ISBN-978-1-921987-22-9, 2014.

[69] S. Sengupta, A. K. Das," A Study on Rough Set Theory based Dynamic Reduct for Classification System Optimization", in the International Journal on Artificial Intelligence and applications(IJAIA), Vol. 5 (4), pp. 35-49, 2014.

[70] J. Luengo, S. García, F. Herrera, "On the choice of the best imputation methods for missing values considering three groups of classification methods", Knowledge and Information Systems, Vol. 32, pp. 77-108, 2011.

[71] S. Zhang, Z. Jin, X. Zhu, "Missing data imputation by utilizing information within incomplete instances", System Software, Vol. 84 (3), pp. 452-459, 2011.

[72] A. A. Alizadeh, "Distinct Types of Diffuse Large B-Cell Lymphoma Identified by Gene Expression Profiling", Nature, Vol. 403, pp. 503- 511, 2000.

[73] J. L. Schafer, J. W. Graham, "Missing data: our view of the state of the art", Psychol. Methods, Vol. 7, pp. 144-177, 2002.

[74] I. B. Aydilek, A. Arslan, "A Novel Hybrid Approach to Estimating Missing Values in Databases using K-nearest Neighbors and Neural Networks", International Journal of Innovative Computing, Vol. 8 (7(A)), pp. 4705-4717, 2012.

[75] A. K. Das and J. Sil, "Dimensionality Reduction and Optimum Feature Selection in Designing Efficient Classifiers", Springer Verlag Lecture Notes, International Conference on Swarm Evolutionary and Memetic Computing, India, 2010.

[76] M. L. Raymer et al., "Dimensionality reduction using genetic algorithms", IEEE Transactions on Evolutionary Computation, Vol. 4(2), pp. 164-171, 2000.

[77] M. A. Carreira-Perpinan, "A review of dimension reduction techniques", Technical report CS-96-09, Department of Computer Science, University of Sheffield, 1997.

[78] Y. Huang, X.Huang and N. Cercone, "Feature Selection with Rough Sets for Web Page Classification", Transactions on Rough Sets, SpringerLink Publishers, Vol. 2, pp. 1-13, 2004.

[79] A. Jain, R. Dubes, "Algorithms for Clustering Data", Englewood Cliffs, NJ: Prentice-Hall, 1988.

[80] A. D. Gordon, "Classification, 2nd ed.", Series: Chapman & Hall/CRC Monographs on Statistics & Applied Probability, London, U.K., ISBN: 9781584880134, 1999.

[81] E. Baralis, S. Chiusano, "Essential classification rule sets", ACM Trans. Database Systems, Vol. 29 (4), pp. 635-674, 2004.

[82] W. Li, J. Han, J. Pei, "CMAR: Accurate and efficient classification based on multiple class-association rules", In Cercone, N., Lin, T. Y., Wu, X., eds., Proceedings of IEEE International Conference on Data Mining, San Jose, California, USA, IEEE Computer Society, pp. 369-376, 2001.

[83] S. Goswami, A. Chakrabarti, B. Chakraborty, "A Proposal for Recommendation of Feature Selection Algorithm based on Data Set Characteristics", J. UCS, Vol. 22 (6), pp. 760-781, 2016.

[84] J. Jager, R. Sengupta, W. L. Ruzzo, "Improved gene selection for classification of microarrays", In Biocomputing 2003: Proceedings of the Pacific Symposium Hawaii, USA 3-7, pp. 53-64, 2003.

[85] X. Liu, Li. M. "Integrated constraint based clustering algorithm for high dimensional data", Neurocomputing, Vol. 142, pp. 478-485, 2014.

[86] G. Chandrashekar, F. Sahin, "A survey on feature selection methods" Computers and Electrical Engineering, Vol. 40, pp. 16-28, 2014.

[87] J. H. Hong, S. B. Cho, "The classification of cancer based on DNA microarray data that uses diverse ensemble genetic programming", Artificial Intelligence in Medicine (Elsevier), Vol. 36, pp. 43-58, 2006.

[88] S. Wang, D. Li, Y. Wei, H. Li, "A Feature Selection Method Based on Fisher's Discriminant Ratio for Text Sentiment Classification", Web Information Systems and Mining, Lecture Notes in Computer Science 5854, pp. 88-97, 2009.

[89] M. Doshi, S. K. Chaturvedi, "Correlation based Feature Selection (CFS) Technique to Predict Student Performance", International Journal of Computer Networks & Communications (IJCNC), Vol. 6 (3), 2014.

[90] R. Kohavi, G. John, "Wrappers for feature subset selection", Artificial Intelligence, Vol. 97(1–2), pp. 273–324, 1996.

[91] R. Kohavi, "Wrappers for Performance Enhancement and Oblivious Decision Graphs", PhD thesis, Stanford University, 1995.

[92] A. L. Blum, P. Langley, "Selection of relevant features and examples in machine learning", Artificial Intelligence, Vol. 97 (1–2), pp. 245-271, 1997.

[93] J. E. Jackson, "A User's Guide to Principal Components", New York: John Wiley and Sons, ISBN 0-471-62267-2, 1991.

[94] J. Harmouche, C. Delpha, D. Diallo, "Incipient fault detection and diagnosis based on Kullback-Leibler divergence using Principal Component Analysis: Part I", Signal Processing, Vol. 94, pp. 278-287, 2014.

[95] A. M. Hall, "Correlation-based feature selection for machine learning", PhD thesis, New Zealand, The University of Waikato, 1999.

[96] Y. Yang, J. O. Pedersen, "A comparative study on feature selection in text categorization", ICML, Vol. 97, pp. 412-420, 1997.

[97] J. Novakovic, P. Strbac, D. Bulatovic, "Toward Optimal Feature Selection using Ranking Methods and Classification Algorithms", Yugoslav Journal of Operations Research, Vol. 21, pp. 119-135, 2011.

[98] Michael J. Todd "The many facets of linear programming", Mathematical Programming, Vol. 91 (3), pp. 417–436, 2002.

[99] R. Marinescu, "Exploiting problem decomposition in multi-objective constraint optimization", International Conference on Principles and Practice of Constraint Programming (CP), pp. 592-607, 2009.

[100] B. Scott, A. Krste, A. David, "Patterson. Searching for a parent instead of fighting over children: A fast breadth-first search implementation for graph500", Technical Report UCB/EECS-2011-117, EECS Department, University of California, Berkeley, 2011.

[101] Karl R. Gegenfurtner, "PRAXIS: Brent's algorithm for function minimization, Behavior Research methods, Vol. 24 (4), pp. 560-564, 1992.

[102] D. E. Goldberg, J. H. Holland, "Genetic algorithms and machine learning", Machine learning, Vol. 3 (2), pp. 95-99, 1988.

[103] D. Gong, G. Wang, X. Sun, Y. Han, "A set-based genetic algorithm for solving the many-objective optimization problem", Soft Computing, Vol. 19 (6), pp. 1477-1495, 2015.

[104] W. Sheng, Y. Liu, X. Meng, T. Zhang, "An Improved Strength Pareto Evolutionary Algorithm 2 with application to the optimization of distributed generations", Computers & Mathematics with Applications, Vol. 64 (5), pp. 944-955, 2012.

[105] P. Moradi, M. Gholampour, "A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy", Applied Soft Computing, Vol. 43, pp. 117–130, 2016.

[106] L. F. Chen, C. T. Su, K. H. Chen, P. C. Wang, "Particle swarm optimization for feature selection with application in obstructive sleep apnea diagnosis", Neural Computing & Applications, Vol. 8, pp. 2087–2096, 2012.

[107] B. Xue, M. J. Zhang, W. N. Browne, "Particle swarm optimization for feature selection in classification: a multi-objective approach", IEEE Transactions on Cybernetic, Vol. 6, pp. 1656-1671, 2013.

[108] P. K. Tripathi, S. Bandyopadhyay, S. K. Pal, "Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients", Information Sciences, Vol. 22, pp. 5033–5049, 2007.

[109] X. Wang, J. Yang, X. Teng, W. Xia, R. Jensen, "Feature selection based on rough sets and particle swarm optimization", Neurocomputing, Vol. 148, pp. 150–157, 2015.

[110] Zhang, Y. & Gong, D. W. "Feature selection algorithm based on bare bones particle swarm optimization" Pattern Recognition Letters, Vol. 4, pp. 459-471, 2007.

[111] http://www.ncbi.nlm.nih.gov/Genbank/genbankstats.html.

[112] Kent Ridge Bio-medical Data Set Repository, http://datam.i2r.a star.edu.sg/ datasetss/krbd.

[113] http://www.ebi.ac.uk/microarray-as/ae.

[114] U. M. Fayyad, D. Haussler and Z. Stolorz, "KDD for Science Data Analysis: Issues and Examples", Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Menlo Park, Calif.: American Association for Artificial Intelligence, pp. 50 – 56, 1996.

[115] C. Kamath et. al., "Scientific Data Analysis: Scientific Data Management", (eds.) A. Shoshani and D. Rotem, Taylor and Francis, pp. 263-301, 2009.

[116] N. Singh, N. Garg, J. Pant, "A Comprehensive Study of Challenges and Approaches for Clustering High Dimensional Data", International Journal of Computer Applications (0975 – 8887), Vol. 92 (4), pp. 7-10, 2014.

[117] L. Silva, R. Moura, A. M. P. Canuto, R. H. N. Santiago, B. Bedregal, "An Interval-Based Framework for Fuzzy Clustering Applications", IEEE Transactions on Fuzzy Systems, Vol. 23 (6), pp. 2174-2187, doi: 10.1109/TFUZZ.2015.2407901, 2015.

[118] R. Kaur, G. S. Bhathal, "A Survey of Clustering Techniques", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3 (5), pp. 153-157, 2013.

[119] S. H. Cha, "Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions", International Journal of Mathematical Models and Methods in applied Sciences, Vol. 1 (4), pp. 300-307, 2007.

[120] A. Vimal, S. R. Valluri, K. Karlapalem**,** "An Experiment with Distance Measures for Clustering", International Conference on Management of Data (COMAD), pp. 241-244, 2008.

[121] N. Iam-On, T. Boongeon, S. Garrett, C. Price, "A Link-Based Cluster Ensemble Approach for Categorical Data Clustering", IEEE Transactions on Knowledge and Data Engineering, Vol. 24 (3), pp. 413-425, doi: 10.1109/TKDE.2010.268, 2012.

[122] F. Murtagh, "A Survey of recent advances in hierarchical clustering algorithms," The Computer Journal, Vol. 26 (4), pp. 354-359, 1983.

[123] A. Baraldi, P. Blonda, "A Survey of fuzzy clustering algorithms for pattern recognition – part I and II," IEEE Trans. Syst., Man, Cybern. B, Cybern., Vol. 29 (6), pp. 778-801, 1999.

[124] D. Xu, Y. Tian, "A Comprehensive Survey of Clustering Algorithms", Annals of Data Science, Vol. 2 (2), pp. 165–193, 2015.

[125] R. Xu, D. Wunsch II, "Survey of Clustering Algorithms", IEEE Transactions on Neural Networks, Vol. 16 (3), pp. 645-678, 2005.

[126] W. Pedrycz, K. Hirota, "Fuzzy vector quantization with the practicle swarm optimization: A study in fuzzy granulation-degranulation information processing," Signal Processing, Vol. 87 (9), pp. 2061-2071, 2007.

[127] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, A. Y. Wu, "An Efficient k-Means Clustering Algorithm: Analysis and Implementation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24 (7), pp. 881-892, 2002.

[128] R. M. Suresh, K. Dinakaran, P. Valarmathie, "Model based modified k-means clustering for microarray data", ICIME, Vol. 53, pp. 271-273, 2009.

[129] A. Bhat, "K-Medoids Clustering using Partitioning Around Mediods Performing Face Recognition", International Journal of Soft Computing, Mathematics and Control (IJSCMC), Vol. 3 (3), pp. 1-12, 2014.

[130] H. Liu, B. Dai, H. He, Y. Yan, "The k-prototype algorithm of clustering high dimensional and large scale mixed data", doi: 10.1142/9789812772763_0110, pp. 738-743.

[131] R. T. Ng, J. Han, "CLARANS: A Method for Clustering Objects for Spatial DataMining", IEEE Transactions on Knowledge and Data Engineering, Vol. 14 (5), pp.1003-1016, 2002.

[132] S. Rajasekaran, "Efficient parallel hierarchical clustering algorithms", IEEE Transactions on Parallel and Distributed Systems, Vol. 16 (6), pp. 497-502, 2005.

[133] B. Eriksson, G. Dasarathy, A. Singh, R. Nowak, "Active Clustering: Robust and Efficient Hierarchical Clustering using Adaptively Selected Similarities", Proceedings of the 14-th International Conference on Artificial Intelligence and Statistics (AISTATS), USA, Vol. 15, pp. 260-268, 2011.

[134] L. Kaufman, P. J. Rousseeuw, "Finding Groups in Data: an Introduction to Cluster Analysis", John Wiley and Sons, 1990.

[135] S. Guha, R. Rastogi, K. Shim, "CURE: An efficient clustering algorithm for large databases", Proc. ACM SIGMOD Int. Conf. Management of Data, pp. 73-84, 1998.

[136] S. Guha, R. Rastogi, K. Shim, "ROCK: A robust clustering algorithm for categorical attributes", Inf. Syst., Vol. 25 (5), pp. 345-366, 2000.

[137] G. Karypis, E. Han, V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," IEEE Computer, Vol. 32 (8), pp. 68-75, 1999.

[138] T. Zhang, R. Ramkrishnan, M. Livny, "BIRCH: An efficient data clustering method for very large databases", Proc. ACM SIGMOD Conf. Management of Data, pp. 103-114, 1996.

[139] L. Kaufman, P. Rousseeuw, "Finding Groups in Data: An Introduction to Cluster Analysis", Wiley Series in Probability and Statistics, ISBN: 9780471878766, 2008.

[140] R. Prabahari, V. Thiagarasu, "Density Based Clustering Using Gaussian EstimationTechnique", International Journal on Recent and Innovation Trends in Computing and Communication, Vol. 2 (12), pp. 4078-4081, 2014.

[141] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, M. Palaniswami, "Fuzzy c-Means Algorithms for Very Large Data", IEEE Transactions on Fuzzy Systems, Vol. 20 (6), pp. 1130–1146, 2012.

[142] A. Bertoni, V. Giorgio, "Fuzzy ensemble clustering for DNA microarray data analysis", Lecture Notes in Computer Science, Vol. 3931, pp. 537-543, 2007.

[143] S. K. Adhikari, J. K. Sing, D. K. Basu, M. Nasipuri, "Conditional spatial fuzzy Cmeans clustering algorithm for segmentation of MRI images", Applied Soft Computing, Vol. 34, pp. 758-769, 2015.

[144] D. L. Davies, D. W. Bouldin, "A cluster separation measure", IEEE Transactions on Pattern Recognition and Machine Intelligence, Vol. 1 (2), pp. 224-227, 1979.

[145] U. Maulik, S. Bandyopadhyay, "Performance Evaluation of Some Clustering Algorithms and Validity Indices", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24 (12), pp. 1650-1654, 2002.

[146] P. L. Odell, B. S. Duran, "Cluster Analysis: A Survey", Springer-Verlag, 1974.

[147] N. Zahid, M. Limouri, A. Essaid, "A new cluster-validity for fuzzy clustering", Pattern Recognition, Vol. 32, pp. 1089-1097, 1999.

[148] E. Rendón, I. Abundez, A. Arizmendi, E. M. Quiroz, "Internal versus External cluster validation indexes", International Journal of Computers and Computations, Vol. 5 (1), pp. 27-34, 2011.

[149] J. Wang, G. Karypis. "Harmony: Efficiently mining the best rules for classification". In SDM, 2005.

[150] R. W. Swiniarski, "Rough sets methods in feature reduction and classification", International Journal of Applied Mathematics and Computer Science, Vol. 11 (3), pp. 565-582, 2001.

[151] S.H. Nguyen, T.T. Nguyen, H.S. Nguyen, "Rough Set Approach to Sunspot Classification Problem", Proceedings of the 2005 International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing - Lecture Notes in Artificial Intelligence 3642, ISBN 978-3-540-28653-0, pp. 263–272, 2005.

[152] I. S. Sitanggang, R. Yaakob, N. Mustapha, A. A. B. Nuruddin, "An extended ID3 decision tree algorithm for spatial data", IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM), pp. 48-53, 2011.

[153] Q. Zhang, X. Lua, M. Lia, Y. Denga, S. Mahadevan, "Network Structure Entropy and Its Application to Scale-free Networks", Systems Engineering- Theory and Practice, Vol. 24 (6), pp. 1-3, 2004.

[154] B. Bonet, H. Geffner, "Planning as heuristic search", Artificial Intelligence, Vol.129 (2), pp. 5-33, 2001.

[155] J. H. Friedman, "A variable metric decision rule for nonparametric classification", Stanford Linear Accelerator Center-PUB-1573, CS-75-487, 1975.

[156] M. Garey, D. Johnson, "Computers and intractability - A guide to the theory of NPcompleteness", Freeman, New York, 1979.

[157] K. Kalpakis, K. Dasgupta, O. Wolfson, "Optimal Placement of Replicas in Trees with Read, Write, and Storage Costs", IEEE Trans. Parallel and Distributed Systems, Vol. 12 (6), pp. 628-637, 2001.

[158] G. W. Corder and D. I. Foreman, "Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach", New Jersey: Wiley, 2009.

[159] F. Wilcoxon, "Individual comparisons by ranking methods", Biometrics, Vol. 1 (6), pp. 80-83, 1945.

[160] M. Hollander, and D. A. Wolfe, "Nonparametric Statistical Methods", Hoboken, NJ: John Wiley and Sons, Inc., 1999.

[161] G. Shanks, "The Challenges of Strategic Data Planning in Practice: An Interpretive Case Study", Journal of Strategic Information Systems, Vol. 6 (1), pp. 69-90, 1997.

[162] S. Tripathi, R. Prakash, N. Melkani, S. Kumar, "Study of Bayes Theorem for Classification of Synthetic Aperture Data", Advances in Computing and Communication Engineering (ICACCE), pp. 187-191, 2015.

[163] R. R. Bouckaert, M. Studeny, "Racing algorithms for conditional independence inference", Int. J. Approx. Reasoning, Vol. 45 (2), pp. 386-401, 2007.

[164] A. Papoulis, "Probability, Random variables and Stochastic processes", McGraw-Hill, 1965.

[165] J. M. Chambers, W. S. Cleveland, B. Kleiner, P. A. Tukey, "Graphical Methods for Data Analysis", Duxbury Press, Boston, 1983.

[166] S. Arora, D. Bhattacharjee, M. Nasipuri, L. Malik, M. Kundu, D. K. Basu, "Performance Comparison of SVM and ANN for Handwritten Devnagari Character

Recognition", IJCSI International Journal of Computer Science Issues, Vol. 7 (3), pp. 1-10, 2010.

[167] A. R. Webb, "Statistical Pattern Recognition", London: Arnold, ISBNs: 0-470-84513-9, 2004.

[168] J. T. Kwok, I. W. Tsang, "Learning with idealized kernels", Proceedings of the 20-th International Conference on Machine Learning, pp. 400-407, 2003.

[169] A. J. Schölkopf, R. Smola, P. Bartlett, "New support vector algorithms", Neural Comput., Vol. 12 (5), pp. 1207–1245, 2000.

[170] M. Sugiyama, H. Ogawa, "Optimal design of regularization term and regularization parameter by subspace information criterion", Neural Networks, Vol. 15 (3), pp. 349-361, 2002.

[171] O. H. Choon, L. C. Hoong, T. S. Huey, "A Functional Approximation Comparison between neural network and polynomial regression", WSEAS Transaction on Mathematics, Vol. 7 (6), pp. 353-363, 2008.

[172] A. K. Das and J. Sil, "Pattern Evaluation using Polynomial Regression – A Clustering and Probabilistic Approach," IEEE International Conference on Granular Computing, pp. 373-376, USA, 2006.

[173] W. D. Wong, S. D. Wexner, A. Lowry, M. Burnstein, F. Denstman, V. Fazio, B. Kerner, C. Simmang, "Practice parameters for sigmoid diverticulitis-supporting documentation", The Standards Task Force, The American Society of Colon and Rectal Surgeons, Dis. Colon Rectum, Vol. 43 (3), pp. 290–297, 2000.

[174] E. D. Andersen, Y. Ye, "A computational study of the homogeneous algorithm for large scale convex optimization", Computational Optimization and Applications Vol. 10, pp. 243-269, 1998.

[175] F. Rosenblatt, "Principle of Neuro dynamics", New York: Spartan Books, 1959.

[176] G. Rama Murthy, "Multi-Dimensional Neural Networks: Unified Theory of Control, Communication and Computation", Research Monograph considered by Pearson Education Publishers, New York.

[177] M. Smith, "Neural Networks for Statistical Modeling", Van Nostrand Reinhold, ISBN 0-442-01310-8, 1993.

[178] A.Saltelli, K. Chan, E. M. Scott, "Sensitivity Analysis", John Wiley and Sons Ltd., 2004.

[179] S. Saha, A. Ekbal, U. K. Sikdar, "Named entity recognition and classification in biomedical text using classifier ensemble", Int. J. of Data Mining and Bioinformatics, Vol. 11 (4), pp.365- 391, 2015.

[180] R. Polikar, "Ensemble based systems in decision making", IEEE Circuits and Systems Magazine 6, pp. 21-45, 2006.

[181] T. T. Soong, "fundamentals of probability and statistics for engineers", John Wiley & Son's Ltd, 2004.

[182] B. V. Dasarathy, B. V. Sheela, "Composite classifier system design: concepts and me-thodology", Proceedings of the IEEE, Vol. 67 (5), pp. 708-713, 2005.

[183] A. X. Carvalho, M. A. Tanner, "Hypothesis testing in mixtures-of-experts of generalized linear time series", IEEE International Conference on Computational Intelligence for Financial Engineering, pp. 285-292, 2003.

[184] L. Peppoloni, M. Satler, E. Luchetti, C. A. Avizzano, P. Tripicchio, "Stacked generalization for scene analysis and object recognition", IEEE 18th International Conference on Intelligent Engineering Systems (INES), Tihany, pp. 215-220, 2014.

[185] O. Abbaszadeh, A. Amiri, A. R. Khanteymoori, "An ensemble method for data stream classification in the presence of concept drift.", Frontiers of Information Technology & Electronic Engineering, Vol. 16 (2), pp.1059-1068.

[186] S. K. Pati, A. K. Das, "Ensemble Classifier Design Selecting Important Genes", International Journal of Data mining and Bioinformatics, Inder Science, 2017.

[187] R. Bryll, R. G. Osuna, F. Quek, "Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets", Pattern Recognition, Vol. 36 (6), pp.1291-1302, 2003.

[188] A. Santana, R. G. F. Soares, A. M. P. Canuto, M. C. P. D. Souto, "A Dynamic Classifier Selection Method to Build Ensembles using Accuracy and Diversity", Ninth Brazilian Symposium on Neural Networks (SBRN'06), Brazil, pp. 36-41, 2006, doi: 10.1109/SBRN.2006.1.

[189] M. Kazemian, B. Moshiri, V. Palade, H. Nikbakht, C. Lucas, "Using classifier fusion techniques for protein secondary structure prediction", Int. J. Comput. Intelligence in Bioinformatics and Systems Biology, Vol. 1 (4), pp. 418-434, 2010.

[190] R. Benmokhtar, B. Huet, "Classifier Fusion: Combination Methods for Semantic Indexing in Video Content", ICANN, Part II, LNCS 4132, pp. 65–74, 2006.

[191] E. Bauer, R. Kohavi, "An empirical comparison of voting classification algorithms: bagging, boosting, and variants", Machine Learning, Vol. 36 (1), pp 105–139, 1999.

[192] X. Sun, H. Zhou, "Experiments with Two new boosting algorithms", Intelligent Information Management, Vol. 2, pp. 386-390, 2010.

[193] J. R. Quinlan, "Bagging, Boosting and C4.5", Proceedings of the thirteenth national conference on Artificial (AAAI'96), Vol. 1, pp. 725-730, 1996.

[194] S. Abdelazeem, "A greedy approach for building classification cascades", In Proceedings of the Seventh International Conference on Machine Learning and Applications, San Diego, CA, USA, pp. 115–120, 2008.

[195] H. Shi, Y. Lv, "An ensemble classifier based on attribute selection and diversity measure", In Proceedings of the 5th International Conference on Fuzzy Systems and Knowledge Discovery, Shandong, China, pp. 106–110, 2008.

[196] N. arcia-Pedrajas, "Constructing ensembles of classifiers by means of weighted instance selection", IEEE Transactions on Neural Networks Vol. 20 (2), pp. 258–277, 2009.

[197] G. McLachlan, T. Krishnan, "The EM Algorithm and Extensions", New York: Wiley, 1997.

[198] P. Tan, M. Steinbach, V. Kumar, "Introduction to Data Mining", Pearson Education, ISBN: 978-81-317-1472-0, 2009.

[199] S. F. Hu, G.Y. Wang, "Quick reduction algorithm based on attribute order", Chinese Journal of Computers, Vol. 30, pp. 1429-1435, 2007.

[200] Y.Y. Yao, Y. Zhao, "Discernibility matrix simplification for constructing attribute reducts", Information ciences, Vol. 179 (5), pp. 867-882, 2009.

[201] G. Qu, S. Hariri, M. Yousif, "A new Dependency and Correlation Analysis for Features", IEEE Transactions on Knowledge and Data Engineering, Vol. 17(9), pp. 1199-1207, 2005.

[202] J. Novovicova et al., "Conditional Mutual Information Based Feature Selection for Classification Task", 12th Iberoamericann Congress on Pattern Recognition, Valparaiso, Chile, pp. 417-426, 2007.

[203] N. Hoque, D. K. Bhattacharyya, J. K. Kalita, "MIFS-ND: A mutual information-based feature selection method", Vol. 41(14), pp. 6371-6385, 2014.

[204] J. Quinlan, R. Rivest, "Inferring decision trees using the minimum description length principle", Inf Comput, Vol. 80, pp. 227–248, 1989.

[205] M. Hansen, B. Yu, "Model selection and the principle of minimum description length", J Am Stat Assoc, Vol. 96, pp. 746–774, 2001.

[206] J. R. Quinlan, "The minimum description length and categorical theories" Proceedings 11th International Conference on Machine learning, New Brunswick, pp. 233-241. San Francisco: Morgan Kaufmann, 1994.

[207] W. S. Roman, H. Larry: Rough sets as a frontend as neural-networks texture classifiers. Neuro-computing, Vol, 36, pp. 85-102, 2001.

[208] X. Hu, T. Y. Lin, J. Jianchao, "A New Rough Sets Model Based on DatabaseSystems", Fundamental Informaticae, pp.1-18, 2004.

[209] R. Jensen, Q. Shen, "Fuzzy-Rough Attribute Reduction with Application to WebCategorization", Fuzzy Sets and Systems, Vol. 141 (3), pp. 469-485, 2004.

[210] N. Zhong, A. Skowron, "A Rough Set-Based Knowledge Discovery Process", International Journal of Applied Mathematics and Computer Science, Vol. 11 (3), pp. 603-619, 2005.

[211] J. Komorowski, A.Ohrn, "Modelling Prognostic Power of Cardiac tests using rough sets", Artificial Intelligence in Medicine, 15, 167-191, 1999.

[212] U. Carlin, J. Komorowski, A. Ohrn, "Rough Set Analysis of Patients with Suspected Acute Appendicitis", Proc., IPMU, 1998.

[213] Dash, M., Liu, H., Motoda, H. "Consistency based feature selection", Proceedings of Fourth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pp. 98–109, 2000.

[214] J. Liang, F. Wang, C. Dang, Y. Qian, "Group Incremental Approach to Feature Selection Applying Rough Set Technique", IEEE Transactions on Knowledge & Data Engineering, Vol. 26 (2), pp. 294-308, 2014.

[215] I. Kononenko, "Estimating attributes: analysis and extensions of relief", Proceedings of the 1994 European Conference on Machine Learning, pp. 171–182, 1994.

[216] M. Petrou, P. Bosdogianni, "Image Processing: The Fundamentals-an example of SVD", John Wiley, pp. 37-44, 2000.

[217] J. García-Nieto, E. Alba, L. Jourdan, E. Talbi, "Sensitivity and specificity based multi-objective approach for feature selection: Application to cancer diagnosis", Information Processing Letters, Vol. 109 (16), pp. 887-896, 2009.

[218] WEKA:Machine Learning Software, http://www.cs.waikato.ac.nz/~ml/

[219] R. Kerber, "ChiMerge: Discretization of Numeric Attributes", Proceedings of ninth Int'l Conf. Artificial Intelligence, AAAI-Press, pp. 123-128, 1992.

[220] R. C. Prim, "Shortest connection networks and some generalizations", Bell System Technical Journal, pp. 1389-1401, 1957.

[221] B. J. Kruskal, "On the shortest Spanning Subtree of a graph and the traveling salesman problem", Proceedings of the American Mathematical Society, Vol. 7, pp. 48-50, 1956.

[222] J. Edmonds, "Optimum branching", Research of the National Bureau of Standards, Vol. 71B, pp. 233-240, 1967.

[223] F. Bock, "An algorithm to construct a minimum spanning tree in a directed network", Developments in Operations Research, Gordon and Breach, NY, pp. 29-44, 1971.

[224] P. Humblet, "A distributed algorithm for minimum weighted directed spanning trees", IEEE Trans. on Communications, Vol. COM-31 (6), pp.756-762, 1983.

[225] A. Gepperth, B. Hammer, "Incremental learning algorithms and applications", ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8, 2016.

[226] F. Wang, J. Liang, C. Dang, "Attribute reduction for dynamic datasets", Applied Soft Computing, Vol. 13, pp. 676-689, 2013.

[227] D. Deng, D. Yan, J. Wang, "Parallel Reducts based on Attribute significance", LNAI 6401, pp. 336-343, 2010.

[228] X. Jun, X. Shen, H. Liu, X. Xu, "Research on an Incremental Attribute Reduction Based on Relative Positive Region", Journal of Computational Information Systems, Vol. 9 (16), pp. 6621–6628, 2013.

[229] L. Dun, L. Tianrui, Z. Junbo, "A rough set-based incremental approach for learning knowledge in dynamic incomplete information systems", International Journal of Approximate Reasoning, Vol. 55, pp. 1764-1786, 2014.

[230] Y. Xu, L. Wang, R. Zhang, "A dynamic attribute reduction algorithm based on 0-1 integer programming", Knowledge-Based Systems, Vol. 24 (8), pp. 1341-1347, 2011.

[231] Shu. Wenhao, S. Hong, "Incremental feature selection based on rough set in dynamic incomplete data", Pattern Recognition, Vol. 47 (12), pp. 3890-3906, 2014.

[232] G.Y. Wang, Z. Zheng, Y. Zhang, "RIDAS-A rough set based intelligent data analysis system", Proceedings of the 1st International conference on machine Learning and Cybernatics, Beiing, Vol. 2, pp. 646-649, 2002.

[233] T. K. Ho, "Complexity of classification problems and comparative advantages of combined classifiers", Int.Workshop on Multiple Classifier Systems, lecture Notes on Computer Science, Vol. 1857, pp. 97-106, Springer verlag, 2000B.

[234] Gabrys, D. Ruta, "Genetic algorithms in classifier fusion", Applied Soft Computing, Vol. 6 (4), pp. 337- 47, 2006.

[235] Z-H. Zhou, J-X. Wu, W. Tang, "Ensembling neural networks: Many could be better than all", Artificial Intelligence, Vol. 137 (1–2), pp. 239–263, 2002.

[236] Myoung-Jong. Kim, Dae-Ki. Kang, "Classifiers selection in ensembles using genetic algorithms for bankruptcy prediction", Expert Systems with Application Vol. 39, pp. 9308-9314, 2012.

[237] S. Kim, F. Scalzo, D. Telesca, X. Hu, "Ensemble of sparse classifiers for high dimensional biological data", Int. J. of Data Mining and Bioinformatics, Vol. 12 (2), pp. 167-183, 2015.

[238] E. Bauer, R. Kohavi, "An empirical comparison of voting classification algorithms: bagging, boosting, and variants", Machine Learning, Vol. 36 (1), pp 105-139, 1999.

[239] X. Sun, H. Zhou, "Experiments with Two new boosting algorithms", Intelligent Information Management, Vol. 2, pp. 386-390, 2010.

[240] R. E. Schapire, Y. Freund, "Boosting of margin: A new explanation for the Effective of voting methods", Vol. 26 (5), pp. 1651-1686, 1998.

[241] A. K. Das, J. Sil, "An efficient classifier design integrating rough set and set oriented database operations", Applied Soft Computing, Vol. 11, pp. 2279-2285, 2011.

[242] Z. Zhang, P. Yang, "An ensemble of classifiers with genetic algorithm based feature selection", The IEEE Intelligent Informatics Bulletin, Vol. 9(1), pp. 18-24, 2008.

[243] W. Ziarko, N. Shan, "A Rough Set-Based Method for Computing All Minimal Deterministic Rules in Attribute-Value Systems", Technical CS-93-02, Department of Computer Science, University of Regina, Canada, 1993.

[244] A. Ulas, M. Semerci, O.T. Yildiz, E. Alpaydin, "Incremental construction of classifier and discriminant ensembles", Information Sciences, Vol. 179 (9), pp. 1298-1318, 2009.

[245] S. Ozawa, S. Pang, N. Kasabov, "Incremental Learning of chunk data for online pattern classification systems", IEEE Transactions on Neural Networks, Vol. 19 (6), pp. 1061-1074, 2008.

[246] Z. Chen, L. Huang, Y. Murphey, "Incremental learning for text document classification", Proc. of IEEE Conference on Neural Networks, pp. 2592-2597, 2007.

[247] M. Seera, P.C. Lim, "A hybrid intelligent system for medical data classification", Expert Systems with Applications, Vol. 41 (5), pp. 2239–2249, 2014.

[248] S. Wenzel, W. Forstner, "Semi-supervised incremental learning of hierarchical appearance models", 21st Congress of the International Society for Photogrammetry and Remote Sensing (ISPRS) pp. 399–404 Part B3b-2, 2008.

[249] S. Ozawa, S. Toh, S. Abe, S. Pang, N. Kasabov, "Incremental Learning for online face recognition", Proc. Of IEEE Conference on Neural Networks, Vol. 5, pp. 3174-3179, 2005.

[250] G. Bakırlı, D. Birant, A. Kut, "An incremental genetic algorithm for classification and sensitivity analysis of its parameters", Expert Systems with Applications, Vol. 38, pp. 2609–2620, 2011.

[251] R. Elmasri, S. B. Navathe, "Fundamentals of Database Systems", third eds., Addison Wesley, ISBN 981-405330-9, 2000.

[252] A. K. Dahama, "Organic Farming for Sustainable Agriculture", Agrobios (India), Jodhpur, 2007.

[253] N. W. Schaad, R. D. Frederick, "Real-Time PCR and Its Application for Rapid Plant Disease Diagnostics", Canadian Journal of Plant Pathology, Vol. 24, pp. 250–258, 2002.

[254] Y.Yang, R. Chai, Y. He, "Early Detection of Rice Blast (Pyricularia) at Seedling Stage in Nipponbare Rice Variety Using Near-Infrared Hyper-Spectral Image", African Journal of Biotechnology, Vol. 11 (26), pp. 1809-1817, 2012.

[255] B. Mohamed, S. Shabbir, A. D'silva J, "Sustainable Agricultural Development", Springer Dordrecht Heidelberg, London, New York, 2011.

[256] A. Meyer-Aurich, "Economic and Environmental Analysis of Sustainable farming Practices – A Bavarian Case Study", Agricultural Systems, Vol. 86 (2), pp. 190-206, 2005.

[257] T. Kobayashi, E. Kanda, K. Kitada, K. Ishiguro, Y. Torigoe, "Detection of Rice Panicle Blast with Multispectral Radiometer and the Potential of Using Airborne Multispectral Scanners", Phytopathology, Vol. 91 (3), pp. 316- 323, 2000.

[258] J. K. Patil, R. Kumar, "Advances in Image Processing For Detection Of Plant Diseases", Journal of Advanced Bioinformatics Applications and Research, Vol. 2, pp. 135-141, 2011.

[259] A. Konar, "Artificial Intelligence and Soft Computing", Taylor & Francis, 1999.

[260] Jr. Pinter, P. J. Hatfield, J. L. Schepers, J. S. Barnes, E. M. Moran, M. S. Daughtry, C. S. T., D. R. Upchurch, "Remote Sensing for Crop Management", Photogrammetric Engineering & Remote Sensing, Vol. 69 (6), pp. 647-664, 2003.

[261] A. Aakif, M. F. Khan, "Automatic classification of plants based on their leaves", Biosystems Engineering, Vol. 139, pp. 66-75, 2015.

[262] S. S. Sannakki, V. S. Rajpurohit, V. B. Nargund, Kumar R. Arun, P. S. Yallur, "A Hybrid Intelligent System for Automated Pomegranate Disease Detection and Grading" International Journal of Machine Intelligence, Vol. 3, pp. 36-44, 2011.

[263] S. K. Sarma, K. R. Singh, A. Singh, "An Expert System for diagnosis of diseases in Rice Plant", International Journal of Artificial Intelligence, Vol. 1, pp. 26-31, 2010.

[264] R. Kaundal, A. S, Kapoor, G. P. S. Raghava,"Machine learning techniques in disease forecasting: a case study on rice blast prediction", BMC Bioinformatics, 2006.

[265] S. Phadikar, J. Sil, A. K. Das, "Rice diseases classification using feature selection and rule generation technique", Computer and Electronics in Agriculture, Vol. 90, pp. 76-85, 2013.

[266] S. Ishiyama, "Studies on white leaf disease of rice plants", Rep. Agric. Exp. Stn, Tokyo, Vol. 45, pp. 233-251, 1992.

[267] S. Phadikar, J. Sil, "Texture Based Classification of Diseased Rice Leaves", IADIS International Conferences Computer Graphics, Visualization, Computer Vision and Image Processing, pp. 523-525, 2010.

[268] X. Tang, "Texture Information in Run-Length Matrices", IEEE transactions on image processing, Vol. 3 (11), pp. 1602-1609, 1998.

[269] M. Robert, K. Haralick, Shanmugam, D. Itshak, "Texture Features for Image Classification", IEEE Transaction on Systems, Man and Cybernetic, SMC-3 (6), pp. 610-621, 1973.

[270] R. Jensen, S. Vluymans, N. Mac Parthaláin, C. Cornelis, Y. Saeys, "Semi-supervised fuzzy-rough feature selection", In Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Springer International Publishing, pp. 185-195, 2015.

[271] R. Diao, Q. Shen, "Fuzzy-rough classifier ensemble selection", IEEE International Conference on Fuzzy Systems (FUZZ), pp. 1516-1522, 2011

About The Book

Data Mining is an established important data analysis technique used for discovering interesting knowledge from huge amount of data in search of consistent patterns and/or systematic relationships between the variables. During this pandemic due to Covid-19 we are observing that all online activities are creating huge amount of data through every application. So extraction of important information from these data is a very challenging task. Due to some inherent chaotic characteristics like noise, high dimension etc. data mining is the appropriate tool for perfect knowledge discovery and analysis of the datasets. Data mining is an iterative process that typically involves phases like problem definition, data searching, data preparation, modelling, assessment and finally deployment of the results. By performing data mining operations, interesting knowledge, regularities and high level information are extracted from datasets which can be applied to decision making, process control, management of information and query processing.

During this pandemic, everything is being done through electronic media which generates huge amount of data in every moment. Developing of some intelligent tools to retrieve the interesting information from these data is the need of the hour. Generally, nowadays data set contains a large number of instances with huge number of features for both static and dynamic data. Therefore, relevant feature selection and classification is one of the main objectives of data mining technique to design the intelligent prediction system for knowledge discovery. Though many research works have been conducted for the data analysis of static and incremental data, still it is an ongoing research to handle newly generated high dimensional data sets to obtain meaningful interpretations. The concerned issues are major requirements and challenges will be addressed in this book by discussing different feature selection and classification techniques to design the prediction systems using the concepts of Rough Set Theory, Graph Theory, Genetic Algorithm, Particle Swarm Optimization.

The objective of this book is to present different data mining approaches for solving prediction problems including the case studies on agricultural fields. The significant contribution of this book is the discussion on designing of different integrated intelligent prediction systems consisting of the major modules such as data dimensionality reduction in terms of feature selection, classifier construction and the application of those intelligent systems to solve real life prediction problems related to agricultural field. Each module has a distinct set of functionalities and plays a specific role within the system towards achieving the goal of the data analysis. Hence the book is organized in such a way where interconnection of all the modules are thoroughly discussed for designing the efficient and intelligent prediction system to solve the real life problems.

About The Author

Dr. Shampa Sengupta

She has received the B.Tech (Hons.) degree from Haldia Institute of Technology and M.Tech degree in Information Technology from Bengal Engineering and Science University. She has received her PhD degree in Engineering from IIEST Shibpur Howrah. Currently she is an Assistant Professor of Information Technology at MCKV Institute of Engineering, Liluah, Howrah. Her research interest domains include Data Mining, Pattern Recognition and Big Data Analytics. She has published several Book Chapters, Conference Papers and Journal Papers in reputed peer-reviewed Journals and International Conferences.

Dr. Asit kumar Das

He has received the B.Tech. and M.Tech. degrees in computer science and technology from Calcutta University and the Ph.D. degree in engineering from Bengal Engineering and Science University Shibpur, Howrah. He is a Professor of the Department of Computer Science and Technology, Indian Institute of Engineering Science and Technology Shibpur, Howrah. He has published one research monograph, five edited books, many book chapters and over 100 research articles in peer-reviewed journals and international conferences. His current research interests include data mining and pattern recognition, social network analysis, evolutionary computing, text, audio and video processing. Prof. Das has already guided ten PhD scholars and five more scholars are currently working under him.

KRIPA DRISHTI PUBLICATIONS

Kripa-Drishti Publications
A-503 Poorva Heights, Pashan-Sus Road, Near Sai Chowk,
Pune – 411021, Maharashtra, India.
Mob: +91 8007068686
Email: editor@kdpublications.in
Web: https://www.kdpublications.in

ISBN: 978-93-90847-37-2