Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 57 (2015) 527 – 536

Procedia Computer Science

3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)

Attribute Reduction in Decision-Theoretic Rough Set Model using Particle Swarm Optimization with the threshold parameters determined using LMS training rule

Srilatha Chebrolu[a]*, Sriram G Sanjeevi[a]

[a]*Department of Computer Science and Engineering, National Institute of Technology, Warangal 506004, India*

Abstract

Decision-theoretic rough set model is the probabilistic generalization of the Pawlak rough set model. In this paper, we have analyzed decision-theoretic rough set model (DTRSM) in the context of attribute reduction. DTRSM is based on Bayesian decision theory for classifying an object into a particular category. The risk associated with classifying an object is defined in terms of loss functions and conditional probabilities. We have used least mean squares learning algorithm to determine the Bayesian loss functions by taking expected overall risk as the learning function. With the loss functions ready, DTRSM can be applied to classification problems. We have proposed attribute reduction in DTRSM by optimizing the expected overall risk using particle swarm optimization algorithm. The proposed algorithm was tested on various data sets found in University of California, machine learning repository. The proposed algorithm has given good results for the cardinality of the reduct and classification accuracy during tests performed on the data sets. Experimental results obtained by the proposed algorithm have been found to give better reduced length of the reduct and classification accuracy in comparison to the results obtained by the consistency subset evaluation feature selection algorithm described in the literature.

Keywords: Attribute reduction; Rough set theory; Decision-theoretic rough set model; Particle swarm optimization; Least mean squares training rule.

* Corresponding author. Tel.: +91-833-096-7036; fax: +91-870-245-9547.
E-mail address: srilatha.chebrolu@nitw.ac.in

1. Introduction

Data collected from real world applications may contain irrelevant, redundant, noisy and unreliable attributes. Accuracy and performance of classifiers built on training such data sets will be poor. In-order to have a better classification accuracy and an improved computation performance, quality of the data sets has to be improved. Quality of the data sets can be improved by identifying and removing irrelevant and redundant attributes. This process is called attribute reduction. Attribute reduction is a pre-process step in data analysis. Attribute reduction is the process of finding a minimal subset of attributes that preserves a particular criterion of the original data set. Some of the criteria of attribute reduction are attribute correlation, entropy, consistency, inter class separability and minimum concept description length. Various approaches to attribute reduction have been proposed in the literature. In [4], Mark A. Hall has proposed correlation based feature subset selection (CFS). CFS uses feature-feature correlation and feature-class correlation to find an optimal subset of features. In [14], Huan Liu et al. have proposed consistency subset evaluation feature selection (CSE) method. In CSE, the metric for feature selection is the consistency level of the class values. In [11], Kira et al. have proposed ReliefF feature selection method. ReliefF selects a random sample instance and identifies the nearest neighbor from the same and different class by considering the attribute values.

Attribute reduction requires a search algorithm and an evaluator. The search algorithm, performs a search operation among all the possible attribute subsets. Commonly used search algorithms are best-first, exhaustive, genetic, greedy, random and rank algorithms. The attribute evaluator will evaluate the relative significance of the attribute subsets. Attribute reduction methods are mainly categorized into wrappers and filters. Wrappers are target learning algorithm dependent. They evaluate attribute subsets based on the score metrics provided by the target learning algorithm. Filters are independent of the learning algorithms. They evaluate an attribute subset based on the attribute filter metrics. Popular filter evaluators are correlation, consistency, chi-squared, information gain, gain ratio and principal components.

In this paper, we propose a filter based approach to attribute reduction. Expected overall risk in DTRSM is considered as scoring metric for attribute subset evaluation and particle swarm optimization algorithm (PSO) is used for searching across all the possible attribute subsets. As DTRSM is used in many applications, we are using this model for attribute evaluation. In [12], Wen Li et al. have proposed an instance centric hierarchical classification framework for text classification based on DTRSM. In [13], Dun Liu et al. have used DTRSM to propose a profit-based three-way approach to investment decision making. In [25], Hong Yu et al. have proposed an autonomous knowledge-oriented clustering technique using DTRSM. In [26], Hong Yu et al. have proposed an approach to automatically determine the number of clusters using DTRSM. In [28], Bing Zhou et al. have introduced three-way decision approach in email spam filtering. In [29], Xianzhong Zhou et al. have proposed a three-way view of decision model based on DTRSM, here optimistic, pessimistic, or equable decision is made based on the cost of error.

In [21, 24], Yiyu Yao has introduced DTRSM as a probabilistic generalization of Rough set theory (RST). DTRSM introduces two threshold parameters α and β, where $0 \leq \beta < \alpha \leq 1$. These threshold parameters probabilistically redefine RST approximations. In DTRSM, threshold parameters are formulated based on Bayesian decision theory loss functions. In this approach, classification decisions are made based on the cost and probability associated with the decision. The loss functions can be provided by the user or they can be determined using machine learning techniques. In [5, 6, 7], game theoretic rough set model (GTRSM) analyzes the problem of determining the threshold parameters based on game theory. GTRSM uses machine learning techniques to obtain a sequence of risk modifications and then finds the loss function values by optimizing one or more classification measures. GTRSM requires user to provide the measures of classification ability and the acceptable threshold to stop the learning procedure. In [8], Xiuyi Jia et al. have determined the threshold parameters by optimizing the decision cost. Here Xiuyi Jia et al. have assumed the search space of the threshold parameters as the set of probabilities of all objects instead of $(0,$. In this paper, we also determine the loss functions in DTRSM using least mean squares (LMS) learning algorithm.

The rest of the paper is organized as follows: section 2 introduces basic concepts of RST, section 3 describes the concepts of DTRSM, section 4 describes the process of determining the threshold parameters of DTRSM, section 5 describes the process of attribute reduction using PSO in DTRSM, section 6 shows the experimental results on different data sets from UCI machine learning repository and section 7 concludes the paper.

2. Rough Set Theory

RST was introduced by Pawlak in 1982 [15, 16, 17]. RST is a mathematical methodology in data analysis to deal with inconsistent and imperfect knowledge. In RST, real world data is represented as decision table. A decision table is defined in terms of attributes, objects, indiscernibility, concepts and consistency. Let $S = (U, C, d, V, f)$ be a decision table, where U is the universe of objects, C is the set of conditional attributes, d is a decision attribute, V is the value set defined by $V = \{V_c \mid c \in C\}$, where V_c is the value set of the conditional attribute c, $f : U \times C \to V$ is a mapping function, where $f(x_i, c)$ represents a value for object $x_i \in U$ on conditional attribute $c \in C$ in the domain V_c.

The main concept of RST is indiscernibility. Let $[x_i]_C$ be the set of indiscernible objects of an object $x_i \in U$. $[x_i]_C$ is also referred as the equivalence class of x_i with respect to C. $[x_i]_C$ is defined as $[x_i]_C = \{x_j \in U \mid \forall c \in C, f(x_i, c) = f(x_j, c)\}$. Let π_C be the partition of U with respect to the set of conditional attributes. $\pi_C = \{[x_i]_C \mid x_i \in U\}$. The set of objects those are of the same decision class are grouped into concepts. Let X_{d_i}, be the concept of decision class d_i. X_{d_i} is defined as $X_{d_i} = \{x_j \in U \mid f(x_j, d) = d_i\}$. Let π_D be the partition of U defined by the decision attribute d_i as $\pi_D = \{X_{d_i} \mid d_i \in V_d\}$. A decision table can be either consistent or inconsistent.

$$S = \begin{cases} consistent & if\ \forall [x_i]_C \in \pi_C, \exists X_{d_j} \in \pi_D \mid [x_i]_C \subseteq X_{d_j} \\ inconsistent & otherwise \end{cases}$$

RST can efficiently analyze inconsistent data sets by defining lower and upper approximations of concepts. Let $LA(X_{d_i} \mid \pi_C)$ and $UA(X_{d_i} \mid \pi_C)$ be the lower and upper approximations of X_{d_i}, respectively defined as follows

$$LA(X_{d_i} \mid \pi_C) = \bigcup_{[xj]_C \in \pi_C} \{[x_j]_C \mid [x_j]_C \subseteq X_{d_i}\}\ and\ UA(X_{d_i} \mid \pi_C) = \bigcup_{[xj]_C \in \pi_C} \{[x_j]_C \mid [x_j]_C \cap X_{d_i} \neq \phi\}.$$

Based on the definitions of lower and upper approximations, a concept $X_{d_i} \in \pi_D$ is defined in terms of three regions i.e., positive region, boundary region and negative region. Positive region of a concept is the set of objects that certainly belong to that concept. Boundary region of a concept is the set of objects that may or may not belong to that concept. Negative region of a concept is the set of objects that certainly does not belong to that concept. Let $POS(X_{d_i} \mid \pi_C)$, $BND(X_{d_i} \mid \pi_C)$ and $NEG(X_{d_i} \mid \pi_C)$ represent the positive, boundary and negative regions of X_{d_i} respectively. $POS(X_{d_i} \mid \pi_C)$, $BND(X_{d_i} \mid \pi_C)$ and $NEG(X_{d_i} \mid \pi_C)$ are defined as follows

$$POS(X_{d_i} \mid \pi_C) = LA(X_{d_i} \mid \pi_C), BND(X_{d_i} \mid \pi_C) = UA(X_{d_i} \mid \pi_C) - LA(X_{d_i} \mid \pi_C)\ and\ NEG(X_{d_i} \mid \pi_C) = U - UA(X_{d_i} \mid \pi_C).$$

3. Decision-Theoretic Rough Set Model

RST requires rigid participation specifications i.e., satisfying the constraint $[x_i]_C \subseteq X_{d_j}$ for deciding $[x_i]_C \subseteq LA(X_{d_j} \mid \pi_C)$ and the constraint $[x_i]_C \cap X_{d_j} \neq \phi$ for deciding $[x_i]_C \subseteq UA(X_{d_j} \mid \pi_C)$. RST is extended to DTRSM by relaxing these strict participation rules, through introducing a pair of threshold parameters α and β, $0 \le \beta < \alpha \le 1$ [20, 22, 23, 27]. These threshold parameters α and β, are formulated based on Bayesian decision theory loss functions. The loss functions can either be known from the domain knowledge or can be determined using machine learning techniques.

In [21, 24], Yiyu Yao has introduced Bayesian decision theory in DTRSM, to determine α and β threshold parameters. Bayesian decision theory is a statistical approach for solving classification problems [1]. In Bayesian decision theory, posterior probabilities are calculated using the prior probabilities and the conditional probabilities. Then the object is assigned to the class with maximum posterior probability, thus achieving minimum probability for the classification error. If there is a penalty or cost associated with each misclassification and if this cost is different for different classes, then weighted posterior probabilities has to be considered for classification decisions.

Yao et al., have related Bayesian decision theory to the problem of deciding $[x_i]_C \subseteq POS(X_{d_j} \mid \pi_C)$ or $[x_i]_C \subseteq BND(X_{d_j} \mid \pi_C)$ or $[x_i]_C \subseteq NEG(X_{d_j} \mid \pi_C)$ as follows. Let a_P, a_B and a_N be the actions of deciding whether $[x_i]_C \subseteq POS(X_{d_j} \mid \pi_C)$ or $[x_i]_C \subseteq BND(X_{d_j} \mid \pi_C)$ or $[x_i]_C \subseteq NEG(X_{d_j} \mid \pi_C)$ respectively. Let $\lambda_{PX_{d_j}}$, $\lambda_{BX_{d_j}}$, $\lambda_{NX_{d_j}}$, $\lambda_{P\bar{X}_{d_j}}$, $\lambda_{B\bar{X}_{d_j}}$ and $\lambda_{N\bar{X}_{d_j}}$ be the loss functions. Let $\lambda_{PX_{d_j}}$, $\lambda_{BX_{d_j}}$ and $\lambda_{NX_{d_j}}$ be the costs incurred for taking actions a_P, a_B and a_N respectively when $x_i \in X_{d_j}$. Let $\lambda_{P\bar{X}_{d_j}}$, $\lambda_{B\bar{X}_{d_j}}$ and $\lambda_{N\bar{X}_{d_j}}$ be the costs incurred for taking actions a_P,

a_B and a_N when $x_i \in \overline{X}_{d_j}$, where $\overline{X}_{d_j}$ is the complement of concept X_{d_j}. $\overline{X}_{d_j} = U - X_{d_j}$. Let $R(a_P \mid [x_i]_C)$, $R(a_B \mid [x_i]_C)$ and $R(a_N \mid [x_i]_C)$ be the conditional risks incurred in taking actions a_P, a_B and a_N respectively. Bayesian decision theory defines these conditional risks as follows

$$R(a_P \mid [x_i]_C) = \lambda_{PX_{d_j}} P(X_{d_j} \mid [x_i]_C) + \lambda_{P\overline{X}_{d_j}} P(\overline{X}_{d_j} \mid [x_i]_C)$$

$$R(a_B \mid [x_i]_C) = \lambda_{BX_{d_j}} P(X_{d_j} \mid [x_i]_C) + \lambda_{B\overline{X}_{d_j}} P(\overline{X}_{d_j} \mid [x_i]_C)$$

$$R(a_N \mid [x_i]_C) = \lambda_{NX_{d_j}} P(X_{d_j} \mid [x_i]_C) + \lambda_{N\overline{X}_{d_j}} P(\overline{X}_{d_j} \mid [x_i]_C)$$

where $P(X_{d_j} \mid [x_i]_C) = \dfrac{|X_{d_j} \cap [x_i]_C|}{|[x_i]_C|}$. The expected overall risk R is defined as follows

$$R = \sum_{[x_i]_C \in \pi_C} R(a_P \mid [x_i]_C) + R(a_B \mid [x_i]_C) + R(a_N \mid [x_i]_C) \qquad (1)$$

Bayesian decision theory states that the action with minimum conditional risk has to be taken among all the possible actions i.e.,

$$\textit{if } R(a_P \mid [x_i]_C) < R(a_B \mid [x_i]_C) \textit{ and } R(a_P \mid [x_i]_C) < R(a_N \mid [x_i]_C) \textit{ then, } [x_i]_C \subseteq POS(X_{d_j} \mid \pi_C)$$

$$\textit{if } R(a_B \mid [x_i]_C) < R(a_P \mid [x_i]_C) \textit{ and } R(a_B \mid [x_i]_C) < R(a_N \mid [x_i]_C) \textit{ then, } [x_i]_C \subseteq BND(X_{d_j} \mid \pi_C)$$

$$\textit{if } R(a_N \mid [x_i]_C) < R(a_P \mid [x_i]_C) \textit{ and } R(a_N \mid [x_i]_C) < R(a_B \mid [x_i]_C) \textit{ then, } [x_i]_C \subseteq POS(X_{d_j} \mid \pi_C)$$

Yao et al., have restated these decision rules under the assumption, $\lambda_{PX_{d_j}} \le \lambda_{BX_{d_j}} < \lambda_{NX_{d_j}}$ and $\lambda_{N\overline{X}_{d_j}} \le \lambda_{B\overline{X}_{d_j}} < \lambda_{P\overline{X}_{d_j}}$ as,

$$POS_{(\alpha,\beta)}(X_{d_j} \mid \pi_C) = \bigcup_{[x_i]_C \in \pi_C} \{[x_i]_C \mid P(X_{d_j} \mid [x_i]_C) \ge \alpha\}$$

$$BND_{(\alpha,\beta)}(X_{d_j} \mid \pi_C) = \bigcup_{[x_i]_C \in \pi_C} \{[x_i]_C \mid \beta < P(X_{d_j} \mid [x_i]_C) < \alpha\}$$

$$NEG_{(\alpha,\beta)}(X_{d_j} \mid \pi_C) = \bigcup_{[x_i]_C \in \pi_C} \{[x_i]_C \mid P(X_{d_j} \mid [x_i]_C) \le \beta\}$$

where $\alpha = \dfrac{\lambda_{P\overline{X}_{d_j}} - \lambda_{B\overline{X}_{d_j}}}{\lambda_{P\overline{X}_{d_j}} - \lambda_{B\overline{X}_{d_j}} + \lambda_{BX_{d_j}} - \lambda_{PX_{d_j}}}$ and $\beta = \dfrac{\lambda_{B\overline{X}_{d_j}} - \lambda_{N\overline{X}_{d_j}}}{\lambda_{B\overline{X}_{d_j}} - \lambda_{N\overline{X}_{d_j}} + \lambda_{NX_{d_j}} - \lambda_{BX_{d_j}}}$ ..

The decision rules of X_{d_j} can be extended to π_D as follows

$$POS_{(\alpha,\beta)}(\pi_D \mid \pi_C) = \bigcup_{X_{d_j} \in \pi_D} \bigcup_{[x_i]_C \in \pi_C} \{[x_i]_C \mid P(X_{d_j} \mid [x_i]_C) \ge \alpha\}$$

$$BND_{(\alpha,\beta)}(\pi_D \mid \pi_C) = \bigcup_{X_{d_j} \in \pi_D} \bigcup_{[x_i]_C \in \pi_C} \{[x_i]_C \mid \beta < P(X_{d_j} \mid [x_i]_C) < \alpha\}$$

$$NEG_{(\alpha,\beta)}(\pi_D \mid \pi_C) = \bigcup_{X_{d_j} \in \pi_D} \bigcup_{[x_i]_C \in \pi_C} \{[x_i]_C \mid P(X_{d_j} \mid [x_i]_C) \le \beta\}$$

Discriminant function is one way of representing a classifier. For each equivalence class $[x_i]_C$, a discriminant function $g([x_i]_C)$ is designed. $[x_i]_C$ is assigned to the region with maximum $g([x_i]_C)$ value. Consider the loss function, where there is no cost for correct classification. Let λ_s be the cost for an incorrect classification. Let λ_r be the cost for classifying an object into a boundary region i.e., $\lambda_{PX_{d_j}} = \lambda_{N\overline{X}_{d_j}} = 0$, $\lambda_{P\overline{X}_{d_j}} = \lambda_{NX_{d_j}} = \lambda_s$ and $\lambda_{BX_{d_j}} = \lambda_{B\overline{X}_{d_j}} = \lambda_r$. Conditional risks associated with each action are given as $R(a_P \mid [x_i]_C) = \lambda_s\,(1 - P(X_{d_j} \mid [x_i]_C))$, $R(a_B \mid [x_i]_C) = \lambda_r$ and $R(a_N \mid [x_i]_C) = \lambda_s\, P(X_{d_j} \mid [x_i]_C)$. Decision rules in this case are defined as

$$(P)\ \textit{if } P(X_{d_j} \mid [x_i]_C) \ge \frac{\lambda_s - \lambda_r}{\lambda_s} \textit{ then, } [x_i]_C \subseteq POS_{(\alpha,\beta)}(X_{d_j} \mid \pi_C)$$

$$(B)\ \textit{if } \frac{\lambda_r}{\lambda_s} < P(X_{d_j} \mid [x_i]_C) < \frac{\lambda_s - \lambda_r}{\lambda_s} \textit{ then, } [x_i]_C \subseteq BND_{(\alpha,\beta)}(X_{d_j} \mid \pi_C)$$

$$(N)\ \textit{if } P(X_{d_j} \mid [x_i]_C) \le \frac{\lambda_r}{\lambda_s} \textit{ then, } [x_i]_C \subseteq NEG_{(\alpha,\beta)}(X_{d_j} \mid \pi_C)$$

The corresponding threshold parameter values are $\alpha = \frac{\lambda_s - \lambda_r}{\lambda_s}$ and $\beta = \frac{\lambda_r}{\lambda_s}$.

Discriminant function can be defined as $g([x_i]_C) = P(X_{d_j} \mid [x_i]_C)$. Depending upon $P(X_{d_j} \mid [x_i]_C)$ value, $[x_i]_C$ is said to be of positive region or boundary region or negative region.

4. Determining threshold parameters using LMS training rule

Algorithm 1 LMS Algorithm to determine optimal loss functions

```
Input: $S$ - decision table, $\eta$ - learning rate, $\varepsilon$ - threshold value
Output: Optimal loss functions.
    1: Initialize each of the loss functions with a small random value
    2: Initialize error value $E$ to zero
    3: For each equivalence class $[x_i]_C$
        i)   Determine $V([x_i]_C)$ as $\begin{cases} 1 & \text{if } (P(X_{d_j} \mid [x_i]_C) = 1) \text{ or } (P(\overline{X}_{d_j} \mid [x_i]_C) = 1) \\ 0 & \text{otherwise} \end{cases}$
        ii)  Use the current loss functions and compute $\hat{V}([x_i]_C)$
                 $\hat{V}([x_i]_C) = \lambda_{PX_{d_j}} P(X_{d_j} \mid [x_i]_C) + \lambda_{P\overline{X}_{d_j}} P(\overline{X}_{d_j} \mid [x_i]_C) + \lambda_{BX_{d_j}} P(X_{d_j} \mid [x_i]_C) + \lambda_{B\overline{X}_{d_j}} P(\overline{X}_{d_j} \mid [x_i]_C) + \lambda_{NX_{d_j}} P(X_{d_j} \mid [x_i]_C) + \lambda_{N\overline{X}_{d_j}} P(\overline{X}_{d_j} \mid [x_i]_C)$
        iii) Update each loss function as
             $\lambda_{PX_{d_j}} = \lambda_{PX_{d_j}} + \eta\, (V([x_i]_C) - \hat{V}([x_i]_C))\, P(X_{d_j} \mid [x_i]_C)$, $\lambda_{P\overline{X}_{d_j}} = \lambda_{P\overline{X}_{d_j}} + \eta\, (V([x_i]_C) - \hat{V}([x_i]_C))\, P(\overline{X}_{d_j} \mid [x_i]_C)$
             $\lambda_{BX_{d_j}} = \lambda_{BX_{d_j}} + \eta\, (V([x_i]_C) - \hat{V}([x_i]_C))\, P(X_{d_j} \mid [x_i]_C)$, $\lambda_{B\overline{X}_{d_j}} = \lambda_{B\overline{X}_{d_j}} + \eta\, (V([x_i]_C) - \hat{V}([x_i]_C))\, P(\overline{X}_{d_j} \mid [x_i]_C)$
             $\lambda_{NX_{d_j}} = \lambda_{NX_{d_j}} + \eta\, (V([x_i]_C) - \hat{V}([x_i]_C))\, P(X_{d_j} \mid [x_i]_C)$, $\lambda_{N\overline{X}_{d_j}} = \lambda_{N\overline{X}_{d_j}} + \eta\, (V([x_i]_C) - \hat{V}([x_i]_C))\, P(\overline{X}_{d_j} \mid [x_i]_C)$
        iv)  Update error as $E = E + (V([x_i]_C) - \hat{V}([x_i]_C))^2$
        v)   If the constraints $\lambda_{PX_{d_j}} \le \lambda_{BX_{d_j}} < \lambda_{NX_{d_j}}$ and $\lambda_{N\overline{X}_{d_j}} \le \lambda_{B\overline{X}_{d_j}} < \lambda_{P\overline{X}_{d_j}}$ are satisfied then, continue learning
             otherwise stop learning and output the current loss function values.
    4: Repeat steps 2 - 3 till $E < \varepsilon$
    5: Return the final loss function
```

To determine optimal values for threshold parameters, we have to design a learning system. A learning system is provided with a set of training examples along with the ideal target function V and a representation to describe the target function. The learned function which we want to approximate the target function can be represented as a linear function of the form $\hat{V} = w_0 + w_1\, x_1 + w_2\, x_2 + \ldots + w_n\, x_n$, where x_1, x_2, …, x_n are features of the problem to be solved and w_0, w_1, w_2, …, w_n are the numerical coefficients or the weights of each feature. The learned function learns its weights w_0, w_1, w_2, …, w_n by using LMS training rule to approximate the target function V.

LMS training rule is a learning algorithm for choosing the weights to best fit the set of training examples. The best fit weights are those which minimize the squared error between the target function $V([x_i]_C)$ values and the learned function $\hat{V}([x_i]_C)$ values. Optimal values for the threshold parameters can be determined using LMS training rule. The target function $V([x_i]_C)$ is defined as

$$V([x_i]_C) = \begin{cases} 1 & \text{if } (P(X_{d_j} \mid [x_i]_C) = 1) \text{ or } (P(\overline{X}_{d_j} \mid [x_i]_C) = 1) \\ 0 & \text{otherwise} \end{cases}$$

Learned function $\hat{V}([x_i]_C)$ is defined as $\hat{V}([x_i]_C) = \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 + \lambda_4 x_4 + \lambda_5 x_5 + \lambda_6 x_6$ where $x_1 = x_2 = x_3 = P(X_{d_j} \mid [x_i]_C)$, $x_4 = x_5 = x_6 = P(\overline{X}_{d_j} \mid [x_i]_C)$, $\lambda_1 = \lambda_{PX_{d_j}}$, $\lambda_2 = \lambda_{BX_{d_j}}$, $\lambda_3 = \lambda_{NX_{d_j}}$, $\lambda_4 = \lambda_{P\overline{X}_{d_j}}$, $\lambda_5 = \lambda_{B\overline{X}_{d_j}}$ and $\lambda_6 = \lambda_{N\overline{X}_{d_j}}$. λ_1 , λ_2 , λ_3 , λ_4 , λ_5 , and λ_6 are the weights used by the learning function. These weights are the loss functions used in the calculation of expected overall risk shown in equation (1). The squared error E between the target function and the learned function is defined as $E = \sum_{[x_i]_C \in \pi_C} (V([x_i]_C) - \hat{V}([x_i]_C))^2$.

The LMS algorithm to determine the optimal values of the loss functions is shown in Algorithm 1. Initialize all the loss functions to a small random values while satisfying the constraints $\lambda_{PX_{d_j}} \le \lambda_{BX_{d_j}} < \lambda_{NX_{d_j}}$ and $\lambda_{N\overline{X}_{d_j}} \le \lambda_{B\overline{X}_{d_j}} < \lambda_{P\overline{X}_{d_j}}$. In this algorithm, in each iteration adjust the loss functions in the direction to reduce the error. The algorithm loops through steps 2 - 3 until the error E falls below the threshold value ε . ε is chosen to be a sufficiently small value. The algorithm returns optimal values for the loss functions. α and β values are determined from the loss functions.

5. Particle Swarm Optimization algorithm for attribute reduction in DTRSM

Algorithm 2 Finding the reduct in DTRSM using PSO

Input: S - decision table, α and β - threshold parameters, n - size of the particle swarm, $V_{\max}$ - maximum velocity of the particles in the swarm, cnt - iteration count, c_1 and c_2 - positive constants, r_1 and r_2 - random numbers, w - inertia factor.

Output: Reduct - an attribute set R with optimized (minimized) fitness value.

```
1: Initialize particles at random positions in the search space
2: Initialize velocities of the particles as a positive integer between $1$ and $V_{\max}$
3: For each particle initialize its best position as the current position
4: Initialize $t$ to zero
5: Repeat steps 6 - 8 till $t < cnt$
6: For each particle $i$ in the swarm
     i)   calculate the fitness value
            a. Let $A$ be the attribute set represented by the position of the particle $i$
            b. $f(A) = \frac{|C|-|A|}{|C|} + \sum_{[x_i]_A \in \pi_A} R(a_P \mid [x_i]_A) + R(a_B \mid [x_i]_A) + R(a_N \mid [x_i]_A)$
     ii)  if the current fitness value of the particle is greater than the fitness value of its previous best
          position then, update its best position to the current position
     iii) calculate its velocity $v_i(t)$
            a. Let $p_i(t-1)$ be the position of the particle at iteration $t-1$, let $pBest_i$ is the best
               position of the particle $i$ till iteration $t$ and let $pBest_{global}$ be the global best position of
               the swarm
            b. $v_i(t) = w * v_i(t-1) + c_1 * r_1 * (pBest_i - p_i(t-1)) + c_2 * r_2 * (pBest_{global} - p_i(t-1))$
     iv)  update its position as $p_i(t) = p_i(t-1) + v_i(t)$
7: Set the global best position $pBest_{global}$ as the position of the particle with the maximum fitness value.
8: Update $t$ to $t+1$.
9: Return the attribute set $R$, that is represented by the position of the global best particle as the reduct with
   the optimized fitness value.
```

Particle swarm optimization (PSO) is an evolutionary algorithm introduced by Kennedy and Eberhart [10]. PSO provides a methodology to solve optimization problems with a huge search space. PSO is motivated by the social behavior of a flock of birds trying to reach an unknown destination. In this paper, we use PSO as a search algorithm to the problem of attribute reduction. Search space for attribute reduction is $2^n - 1$ attribute subsets, where $n = |C|$. In [19], Xiangyang Wang et al. have solved the problem of attribute reduction using PSO.

Xiangyang Wang et al. have represented the position of a particle by a bit string of length $|C|$. Let $C = \{c_1, c_2, \ldots, c_m\}$, be the set of conditional attributes. A value of 1 in the bit string at position i represents c_i is present in the attribute set and a value of 0 in the bit string at position i represents c_i is not present in the attribute set. Hence, an attribute set is represented by the position of a particle. Different attribute sets are represented by the respective positions of particles in the swarm. The difference between two positions is the number of distinct bits between the two positions.

Xiangyang Wang et al. have defined velocity of a particle as the positive integer between 1 and $V_{\max}$. The velocity of a particle i at iteration t is defined as follows as

$$v_i(t) = w * v_i(t-1) + c_1 * r_1 * (pBest_i - p_i(t-1)) + c_2 * r_2 * (pBest_{global} - p_i(t-1))$$

where w is the inertia factor, $v_i(t-1)$ is the velocity of the particle i at iteration $t-1$, c_1 and c_2 are the positive constants, r_1 and r_2 are the random numbers, $pBest_i$ is the best position of the particle i till iteration $t-1$, $p_i(t-1)$ is the position of the particle i at iteration $t-1$ and $pBest_{global}$ is the global best position of the swarm till iteration $t-1$. The position of the particle i at iteration t is updated as $p_i(t) = p_i(t-1) + v_i(t)$, where $p_i(t)$ is the position of the particle i at iteration t, $p_i(t-1)$ is the position of the particle i at iteration $t-1$ and $v_i(t)$ is the velocity of the particle i at iteration t.

Fitness function of an attribute set $A \subseteq C$ is denoted by $f(A)$. An attribute set with a minimum number of attributes and with minimum expected overall risk will be the fittest attribute set. Such attribute sets will survive through PSO iterations. The attribute set with minimum fitness value will be selected as the reduct. The fitness function $f(A)$ is defined as

$$f(A) = \frac{|C| - |A|}{|C|} + \sum_{[x_i]_A \in \pi_A} R(a_P \mid [x_i]_A) + R(a_B \mid [x_i]_A) + R(a_N \mid [x_i]_A).$$

The first term of $f(A)$ corresponds to a measure of the cardinality of the reduct and the second term corresponds to the expected overall risk. The PSO algorithm to find reduct in DTRSM is as shown in Algorithm 2. The inputs for this algorithm are decision table S, threshold parameters α and β determined using the LMS training rule, size of the particle swarm n, maximum velocity of the particles in the swarm $V_{\max}$, positive constants c_1 and c_2, random numbers r_1 and r_2 and iteration count. The output of this algorithm is the reduct R with minimized fitness value.

6. Experimental results and analysis

In this section, we analyze the effectiveness of our proposed DTRSM-PSO algorithm for attribute reduction by conducting a series of experiments on 10 data sets from machine learning repository at University of California, Irvine [2]. The details of all the 10 data sets selected for our experimentation are shown in table 1. All the chosen data sets are of multi-category, with the number of classes ranging from 2 to 24, the number of objects ranging from 148 to 8124 and the number of conditional attributes ranging from 9 to 69. The experiments were conducted on a 2.27 GHz PC running Windows 7 with Intel Core $i3$ processor with 4 GB RAM.

Table 1. Data sets description.

S. No	Data set	No. of Conditional Attributes	No. of Objects	No. of Classes
1.	Audiology	69	226	24
2.	Breast-cancer	9	286	2
3.	Dermatology	34	366	6
4.	German-credit	20	1000	2
5.	Hepatitis	19	155	2
6.	Lymphography	18	148	4
7.	Mushroom	22	8124	2
8.	Primary-tumor	17	339	22
9.	Vehicle	18	846	4
10.	Vote	16	435	2

We conducted a series of experiments using DTRSM-PSO algorithm to determine a reduct on all the 10 data sets shown in table 1. We also compared our proposed DTRSM-PSO algorithm with the reduct obtained using the CSE algorithm [14] described in the literature. Criteria for comparison chosen are i) cardinality of the reduct ii) attributes of reduct iii) classification accuracies obtained, when tested using the C4.5 classifier [18] and iv) classification accuracies obtained, when tested using the Naive Bayes classifier [9].

Table 2 shows the comparisons on cardinality of the reduct determined by CSE algorithm and DTRSM-PSO algorithm. These results show that DTRSM-PSO algorithm is achieving a minimal length reduct on 6 out of 10 data sets. CSE algorithm is achieving a minimal length reduct on 3 out of 10 data sets. Both the algorithms are achieving equal length reduct on remaining 1 data set. The bold values in this table indicate the minimal length reduct.

Table 2. Comparisons of the cardinality of the reduct determined by CSE and DTRSM-PSO algorithm.

S. No	Data set	CSE	DTRSM-PSO
1.	Audiology	**13**	16
2.	Breast-cancer	8	**4**
3.	Dermatology	**9**	23
4.	German-credit	11	**6**
5.	Hepatitis	6	**5**
6.	Lymphography	**7**	12
7.	Mushroom	5	**3**
8.	Primary-tumor	16	**11**
9.	Vehicle	**7**	**7**
10.	Vote	12	**4**

Reduced data sets are obtained by retaining the attributes of reduct in the original data sets and removing other attributes not in the reduct. The performance of DTRSM-PSO algorithm is evaluated by training C4.5 classifier and Naive Bayes classifier on the reduced data sets. We have used the implementation of C4.5 classifier and Naive Bayes classifier provided by Weka software tool [3]. Classification accuracy was obtained by using C4.5 classifier and Naive Bayes classifier with 10 fold cross-validation approach for validation.

Table 3. Comparisons of the reduct attributes obtained using CSE and DTRSM-PSO algorithm

S. No	Data set	CSE	DTRSM-PSO
1.	Audiology	$\{c_1, c_2, c_4, c_5, c_6, c_7, c_{10}, c_{14}, c_{15}, c_{40}, c_{47}, c_{64}, c_{66}\}$	$\{c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_{10}, c_{11}, c_{15}, c_{57}, c_{58}, c_{59}, c_{60}, c_{64}, c_{66}\}$
2.	Breast-cancer	$\{c_1, c_2, c_3, c_4, c_6, c_7, c_8, c_9\}$	$\{c_4, c_5, c_6, c_9\}$
3.	Dermatology	$\{c_1, c_4, c_9, c_{14}, c_{15}, c_{21}, c_{32}, c_{33}, c_{34}\}$	$\{c_2, c_3, c_4, c_5, c_6, c_8, c_9, c_{10}, c_{12}, c_{14}, c_{15}, c_{16}, c_{19}, c_{20}, c_{21}, c_{22}, c_{24}, c_{25}, c_{26}, c_{27}, c_{28}, c_{29}, c_{33}\}$
4.	German-credit	$\{c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9, c_{13}, c_{14}\}$	$\{c_1, c_2, c_3, c_4, c_5, c_6\}$
5.	Hepatitis	$\{c_1, c_{11}, c_{12}, c_{14}, c_{15}, c_{16}\}$	$\{c_6, c_{12}, c_{13}, c_{18}, c_{19}\}$
6.	Lymphography	$\{c_1, c_2, c_8, c_{10}, c_{13}, c_{14}, c_{18}\}$	$\{c_1, c_2, c_7, c_8, c_9, c_{10}, c_{11}, c_{12}, c_{13}, c_{14}, c_{15}, c_{18}\}$
7.	Mushroom	$\{c_2, c_3, c_5, c_{12}, c_{20}\}$	$\{c_5, c_9, c_{20}\}$
8.	Primary-tumor	$\{c_1, c_2, c_3, c_4, c_5, c_7, c_8, c_9, c_{10}, c_{11}, c_{12}, c_{13}, c_{14}, c_{15}, c_{16}, c_{17}\}$	$\{c_1, c_2, c_3, c_4, c_5, c_9, c_{10}, c_{13}, c_{15}, c_{16}, c_{17}\}$
9.	Vehicle	$\{c_2, c_3, c_9, c_{13}, c_{15}, c_{16}, c_{18}\}$	$\{c_1, c_3, c_7, c_8, c_9, c_{11}, c_{12}\}$
10.	Vote	$\{c_1, c_2, c_3, c_4, c_5, c_7, c_{10}, c_{11}, c_{12}, c_{13}, c_{15}, c_{16}\}$	$\{c_3, c_4, c_5, c_{12}\}$

Next, we present the comparison analysis of the performance of our proposed DTRSM-PSO algorithm with the CSE algorithm. Table 3 shows the comparisons on the reducts obtained using CSE algorithm and DTRSM-PSO algorithm. For the different data sets, columns 3 and 4 of table 4 shows the comparisons of classification accuracies obtained by training C4.5 classifier for the reduct determined by DTRSM-PSO algorithm with the reduct obtained using CSE algorithm. These results show that DTRSM-PSO algorithm is achieving a reduct with higher classification accuracies on majority i.e., 6 out of 10 data sets. CSE algorithm is achieving reduct with higher classification accuracies on 4 out of 10 data sets. For the different data sets, columns 5 and 6 of table 4 shows the comparisons of classification accuracies obtained for the reduct determined by DTRSM-PSO algorithm with the reduct obtained using CSE algorithm. These results show that DTRSM-PSO algorithm is achieving a reduct with higher classification accuracies on majority i.e., 7 out of 10 data sets. CSE algorithm is achieving a reduct with higher classification accuracies on 2 out of 10 data sets. Both the algorithms are achieving equal classification accuracy on remaining 1 data set. The bold values in this table indicate the higher classification accuracies.

Table 4. Comparisons of the classification accuracy obtained by CSE and DTRSM-PSO algorithm

S. No	Data set	C4.5		Naive Bayes	
		CSE	DTRSM-PSO	CSE	DTRSM-PSO
1.	Audiology	75.67	**76.30**	69.09	**73.28**
2.	Breast-cancer	72.51	**73.75**	**72.29**	**72.27**
3.	Dermatology	89.45	**92.05**	90.71	**97.21**
4.	German-credit	73.38	**74.44**	**76.23**	75.29
5.	Hepatitis	**83.65**	83.13	79.88	**83.99**
6.	Lymphography	75.95	**80.41**	83.56	**85.09**
7.	Mushroom	**100**	99.41	98.52	**98.88**
8.	Primary-tumor	41.04	**42.63**	**47.14**	46.46
9.	Vehicle	**65.16**	64.54	56.43	**57.48**
10.	Vote	**96.32**	95.63	91.82	**92.85**

From the experimental results obtained, we conclude that DTRSM-PSO algorithm is giving better results compared to CSE algorithm in terms of i) cardinality of the reducts found for different data sets ii) classification accuracies obtained for different data sets, when tested using the C4.5 classifier and iii) classification accuracies obtained for different data sets, when tested using the Naive Bayes classifier.

7. Conclusion

The contribution of this paper is the proposed DTRSM-PSO algorithm. In this algorithm, the attribute subset evaluation is done by optimizing the expected overall risk in DTRSM. Particle swarm optimization is used as the search algorithm for finding the reduct. We have analyzed the performance of our proposed DTRSM-PSO algorithm by conducting a series of experiments on 10 data sets from machine learning repository at University of California, Irvine. Initially, we conducted a series of experiments using DTRSM-PSO algorithm to determine reducts for each of the 10 data sets. Reduced data sets from each of the 10 data sets are obtained by retaining the attributes of reduct in the original data sets and removing other attributes not in the reduct. Testing for classification accuracy was done on all 10 reduced data sets using C4.5 classifier and Naive Bayes classifier respectively. We compared our proposed DTRSM-PSO algorithm with the consistency subset evaluation feature selection algorithm proposed by Huan Liu et al. We conclude that DTRSM-PSO algorithm is giving better results compared to consistency subset evaluation feature selection algorithm as shown by the experimental results obtained for i) cardinality of the reducts found for different data sets ii) classification accuracies obtained for different data sets, when tested using the C4.5 classifier and iii) classification accuracies obtained for different data sets, when tested using the Naive Bayes classifier.

Acknowledgements

This research work was supported by Tata Consultancy Services under the TCS Research Scholar Program Scheme.

References

1. Duda, R. O., Hart, P. E., Stork, D. G., 2000. Pattern Classification. Wiley-Interscience.
2. Frank, A., Asuncion, A., 2010. Uci machine learning repository. University of California, Irvine, School of Information and Computer Sciences. URL http://archive.ics.uci.edu/ml
3. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H., 2009. The weka data mining software: An update. SIGKDD Explorations 11, 10–18.
4. Hall, M. A., 2000. Correlation-based feature selection for discrete and numeric class machine learning. Morgan Kaufmann, pp. 359–366.
5. Herbert, J. P., Yao, J., 2009. Learning optimal parameters in decision-theoretic rough sets. In: Rough Sets and Knowledge Technology. Vol. 5589 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 610–617.
6. Herbert, J. P., Yao, J., 2011. Analysis of data-driven parameters in game-theoretic rough sets. In: Rough Sets and Knowledge Technology. Vol. 6954 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 447–456.
7. Herbert, J. P., Yao, J., 2011. Game-theoretic rough sets. Fundamenta Informaticae 108 (3-4), 267–286.
8. Jia, X., Li, W., Shang, L., Chen, J., 2011. An optimization viewpoint of decision-theoretic rough set model. In: Rough Sets and Knowledge Technology. Vol. 6954 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 457–465.
9. John, G. H., Langley, P., 1995. Estimating continuous distributions in bayesian classifiers. In: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. pp. 338–345.
10. Kennedy, J., Eberhart, R., Nov 1995. Particle swarm optimization. In: Neural Networks, 1995. Proceedings., IEEE International Conference on. Vol. 4. pp. 1942–1948.
11. Kira, K., Rendell, L. A., 1992. A practical approach to feature selection. In: Proceedings of the Ninth International Workshop on Machine Learning. pp. 249–256.
12. Li, W., Miao, D., Wang, W., Zhang, N., 2010. Hierarchical rough decision theoretic framework for text classification. In: Cognitive Informatics (ICCI), 2010 9th IEEE International Conference on. pp. 484–489.
13. Liu, D., Yao, Y., Li, T., 2011. Three-way investment decisions with decision-theoretic rough sets. International Journal of Computational Intelligence Systems 4 (1), 66–74.
14. Liu, H., Setiono, R., 1996. A probabilistic approach to feature selection - a filter solution. In: 13th International Conference on Machine Learning. Morgan Kaufmann, pp. 319–327.
15. Pawlak, Z., Skowron, A., January 2007. Rough sets and boolean reasoning. Information Sciences 177 (1), 41–73.
16. Pawlak, Z., Skowron, A., January 2007. Rough sets: Some extensions. Information Sciences 177 (1), 28–40.
17. Pawlak, Z., Skowron, A., January 2007. Rudiments of rough sets. Information Sciences 177 (1), 3–27.
18. Quinlan, J. R., 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
19. Wang, X., Yang, J., Teng, X., Xia, W., Jensen, R., 2007. Feature selection based on rough sets and particle swarm optimization. Pattern Recognition Letters 28 (4), 459 – 471.
20. Yao, Y., 2004. Information granulation and approximation in a decision-theoretical model of rough sets. In: Rough-Neural Computing. Cognitive Technologies. Springer Berlin Heidelberg, pp. 491–516.
21. Yao, Y., 2007. Decision-theoretic rough set models. In: Rough Sets and Knowledge Technology. Vol. 4481 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 1–12.
22. Yao, Y., 2009. Three-way decision: an interpretation of rules in rough set theory. In: Rough Sets and Knowledge Technology. Vol. 5589 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 642–649.
23. Yao, Y., 2011. Two semantic issues in a probabilistic rough set model. Fundamenta Informaticae 108, 249–265.
24. Yao, Y., Wong, S. K. M., Lingras, P. J., 1990. A decision-theoretic rough set model. In: ISMIS. pp. 17–25.
25. Yu, H., Chu, S., Yang, D., 2010. Autonomous knowledge-oriented clustering using decision-theoretic rough set theory. In: Rough Set and Knowledge Technology. Vol. 6401 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 687–694.
26. Yu, H., Liu, Z., Wang, G., 2011. Automatically determining the number of clusters using decision-theoretic rough set. In: Rough Sets and Knowledge Technology. Vol. 6954 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 504–513.
27. Zhou, B., Yao, Y., 2010. In search for e_ective granularity with dtrs. In: Cognitive Informatics (ICCI), 2010 9th IEEE International Conference on. pp. 464–470.
28. Zhou, B., Yao, Y., Luo, J., 2010. A three-way decision approach to email spam filtering. In: Advances in Artificial Intelligence. Vol. 6085 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 28–39.
29. Zhou, X., Li, H., 2009. A multi-view decision model based on decision-theoretic rough set. In: Rough Sets and Knowledge Technology. Vol. 5589 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 650–657.