On Feature Selection and Effective Classifiers *

Suresh K. Choubey[†] Jitender S. Deogun[‡] Vijay V. Raghavan[§]
Hayri Sever[¶]

Abstract

In this paper, we develop and analyze four algorithms for feature selection in the context of rough set methodology. The initial state and the feasibility criterion of all these algorithms are the same. That is, they start with a given feature set and progressively remove features, while controlling the amount of degradation in classification quality. These algorithms, however, differ in the heuristics used for pruning the search space of features. Our experimental results confirm the expected relationship between the time complexity of these algorithms and the classification accuracy of the resulting upper classifiers. Our experiments demonstrate that a θ-reduct of given feature set can be found efficiently. Although we have adopted *upper classifiers* in our investigations, the algorithms presented can however be used with any method of deriving a classifier where the quality of classification is a monotonically decreasing function of the size of the feature set.

We compare the performance of upper classifiers with those of lower classifiers. We find that upper classifiers perform better than lower classifiers for duodenal ulcer data set. This should be generally true when there is a small number of elements in the boundary region. An upper classifier has some important features that make it suitable for data mining applications. In particular, we have shown that the upper classifiers can be summarized at a desired level of abstraction by using extended decision tables. We also point out that an upper classifier results in an inconsistent decision algorithm, which can be *interpreted* deterministically or non-deterministically to obtain a consistent decision algorithm.

Keywords- Rough sets, feature selection, classification, data mining, knowledge discovery.

*This research was supported in part by the Army Research Office, Grant No. DAAH04-96-1-0325, under DEPSCoR program of Advanced Research Projects Agency, Department of Defense.

[†]skc@cacs.usl.edu, The Center for Advanced Computer Studies, University of SW Louisiana,Lafayette,LA 70504,USA

[‡]deogun@cse.unl.edu, The Department of Computer Science, University of Nebraska, Lincoln, NE 68588, USA

[§]raghavan@cacs.usl.edu, The Center for Advanced Computer Studies, University of SW Louisiana,La fayette, LA 70504, USA

[¶]sever@eti.cc.hun.edu.tr, The Department of Computer Science, Hacettepe University, 06532 Beytepe, Ankara, TR

1 Introduction

A data-mining process involves extracting valid, previously unknown, potentially useful, and comprehensible patterns from large databases. As described in Fayyad (1996) and Simoudis (1996), this process is typically made up of selection and sampling, preprocessing and cleaning, transformation and reduction, data mining, and evaluation steps. The first step in data-mining process is to select a target data set from a database (or a data warehouse) and to possibly sample the target data. The preprocessing and data cleaning step handles noise and unknown values as well as accounting for missing data fields, time sequence information, and so forth. The data reduction and transformation step involves finding relevant features depending on the goal of the task and certain transformations on the data such as converting one type of data to another (e.g., changing nominal values into numeric ones, discretizing continuous values), and/or defining new attributes. In the mining step, the user may apply one or more knowledge discovery techniques on the transformed data to extract valuable patterns. Finally, the evaluation step involves interpreting the result (or discovered pattern) with respect to the goal/task at hand. Note that the data-mining process is not linear and involves a variety of feedback loops, because any one step can result in changes in preceding or succeeding steps. Furthermore, the nature of a large, real-world data set, which may contain *noisy, incomplete, dynamic, redundant, spare, and missing values*, certainly requires that existing techniques and approaches be extended to cope with such problems (Deogun et al., 1996; Matheus, Chan & Piatetsky-Shapiro, 1993). This paper uses rough set model to address issues related to some aspects of real-world data and investigates the interactions between feature selection algorithms and rough classifiers.

The potential for using rough set methodology for investigating problems relating to very large and dynamic data in real-world databases is very strong and therefore it represents a novel area for research in data mining. The rough set theory provides a formal framework for knowledge discovery (Lin, 1996; Pawlak, 1981; Pawlak, 1991; Pawlak et al., 1995). It has several attractive features such as deriving rules only from facts present in data and finding a simpler representation of knowledge in the form of decision table (or rules). In this paper we consider classification methods only in algebraic approximation spaces, which do not require any preliminary or additional information about data. We use upper classifiers and extended decision tables to address special problems of *noisy and dynamic* data. Approaches for vertical and horizontal reduction of a data set are needed in the process of dealing with *very large* data sets. Hence, in this paper we develop and evaluate feature selection strategies for the vertical reduction of a data set.

Feature selection is the problem of choosing a small subset of features that is necessary and sufficient to describe target concept(s). The importance of feature selection in a broader sense is due to the potential for speeding up the processes of both concept learning and classifying objects, reducing the cost of classification (e.g., eliminating redundant tests in medical diagnosis), and improving the quality of classification (Kira & Rendell, 1992). It is well known that searching for the smallest subset of features in the feature space takes time that is bounded by $O(2^l J)$, where: l is the number of features, and J is the computational effort required to evaluate each subset. This type of exhaustive search would be appropriate only

if l is small and J is computationally inexpensive. Greedy approaches like Sequential backward/forward techniques (James, 1985; Modrzejewski, 1993), and dynamic programming (Chang, 1973) are some of the efficient search techniques applied with some feature selection criterion. For near-optimal solutions or optimal solutions in special cases, weights of either individual features or combinations of features are computed with respect to some feature selection criteria (or measures) such as Bhattacharya coefficient, divergence, Kolmogorov variational distance, etc., in statistics (Devijver & Kittler, 1982; Miller, 1990); entropy or classification accuracy in pattern recognition and machine learning (Pal & Chakraborty, 1986; Duda & Hart, 1973; Fayyad & Irani, 1992); classification quality based on variations of MZ metric in information retrieval systems (Bollmann & Cherniavsky, 1981). In such procedures, irrelevant features are either eliminated or assigned small coefficients.

We have proposed in Deogun et al. (1995), ways to improve upper classifiers – one of the classification methods in rough set theory. The enhancement is achieved by a sequential backward feature selection algorithm to preprocess a given set of features. This is important because rough classification methods are incapable of removing superfluous features. We also proved that the sequential backward selection algorithm finds a small subset of relevant features that are ideally sufficient and necessary to define target concepts with respect to a given threshold. This threshold value indicates acceptable degradation of classification quality. We propose mechanism for making upper classifier adaptive. This is accomplished by associating some kind of frequency information with each tuple of a decision table. The frequency information associated with each tuple is called *incremental information*. The notion of decision table is also extended to represent an adaptive upper classifier. It is also used for interpreting the upper classifier either deterministically or nondeterministically. In Sever (1995), findings on only one heuristic, namely stepwise backward feature selection, was reported. We then proposed four heuristics and completed the experimental studies using various machine learning data sets. Preliminary results on these findings were reported in Choubey et al. (1996). There are two important goals that we intend to accomplish in this paper.

- First, we perform extensive experimental studies with a real life data set using a family of stepwise backward selection algorithms for upper classifiers.
- Second, we compare the performance of upper classifiers to that of lower classifiers on a real data set. The data set that we used have been used by Pawlak et al. (1986) for lower classifiers.

Rest of the paper is organized as follows: In Section 2, we present a brief discussion of classification methods in the context of rough set theory. In this section, we also develop the notion of accuracy of classification that can be used for upper or lower classification methods and formally define the feature selection problem. A formal discussion of upper classifiers is presented in Section 3. In Section 4, we develop four feature selection heuristics and discuss the family of stepwise backward selection algorithms, in detail, and analyze the worst case time complexity for all algorithms presented. This section is divided into four subsections, one for each of the four heuristics. In Section 5, experimental setup and results are discussed.

This section is divided into two subsections. In the first subsection, we discuss details of the experiments and results of using family of stepwise backward selection on learning data sets and duodenal ulcer data set. The second subsection includes the experimental setup and results of comparison of lower classifiers and upper classifiers on duodenal ulcer data set. A discussion on extended decision tables is presented in Section 6 and concluding remarks are presented in Section 7.

2 Classification Methods in Rough Set Theory

A decision table is defined as a quadruple $S = (U, Q, V, \rho)$ where: U is the finite set of objects; Q is the union of condition, denoted by CON, and decision attributes, denoted by DEC (i.e., $Q = CON \cup DEC$); V is the union of domains of attributes in Q; and $\rho : U \times Q \Rightarrow V$ is a total description function. For all $x \in U$ and $a \in Q$, let $\rho_x(a) = \rho(x, a)$. Let $P \subseteq Q$ and $P = \{P_1, P_2, \ldots, P_m\}$. Let us introduce following notations.

(i) A P-tuple, $\bar{\rho}_x\ (P)$, is a mapping from P to V_P, union of domains of attributes in P, i.e., $\bar{\rho}_x$: $P \rightarrow V_P$.

(ii) The equivalence relation on U defined by values of P is denoted $\widetilde{P} = \{(x, y) :\ x, y \in U \wedge\ \bar{\rho}_x\ (P) = \bar{\rho}_y\ (P)\}$.

(iii) An equivalence class is denoted $[x]_{\widetilde{P}} = \{y\ :\ \bar{\rho}_x\ (P) = \bar{\rho}_y\ (P)\}$.

(iv) The set of all equivalence classes is denoted $U/\widetilde{P} = \{[x]_{\widetilde{P}} :\ x \in U\}$.

Let $X = [x]_{\widetilde{DEC}}$ be a concept of interest in S. A decision algorithm, denoted by $D_S(X)$, is induced from S such that, for a given object y, it yields one of these three answers: a) y is in X, b) y is not in X, c) *unknown*. Using anology with these answers, we say that the approximation of X in S yields a partition of U into three groups. Now, we define the groups of X in S corresponding to each answer. Let $POS_S(X)$ be the set of objects that is considered by the decision algorithm $D_S(X)$ to belong to the concept X. Let $BND_S(X)$ be the set of objects for which $D_S(X)$ gives the answer of *unknown*. Finally, let $NEG_S(X)$ be the set of objects that are not regarded as members of X by $D_S(X)$. It is easy to see that $NEG_S(X) = U - (POS_S(X) \bigcup BND_S(X))$ by definition.

The approximation quality of *a decision algorithm* $D_S(X)$, denoted by AD, is defined as the ratio of objects in $POS_S(X)$ that are correctly approximated to be in X to all objects in $POS_S(X)$, thus, $AD = |POS_S(X) \cap X| \,/\, |POS_S(X)|$. We also introduce a second quality measure that is called the approximation quality of *a concept* X in S and is denoted by AC. It is defined as the ratio of objects in X that are correctly approximated as $POS_S(X)$ to all objects in X, $AC = |POS_S(X) \cap X| \,/\, |X|$. To get the overall picture, we propose to

consider the normalized size of intersection between $POS_S(X)$ and X. This intuitive idea of overall approximation quality, denoted by $\mu_S(X)$, can be formalized as follows:

$$\mu_S(X) = \frac{1}{s_1 \frac{1}{AC} + s_2 \frac{1}{AD}} = \frac{|X \cap POS_S(X)|}{s_1 |X| + s_2 |POS_S(X)|},$$

where s_1 and s_2 are scaling factors and their sum must be equal to one. These scaling factors quantify the user's preference as to amount of increment in accuracy of $D_S(X)$ desired relative to a certain loss in accuracy of X (or vice versa). If $s_1 = s_2 = 0.5$, then the measure becomes equal to what we call Dice's coefficient in information retrieval systems.

In the following, we introduce positive regions of the three approximation methods.

1. *The lower bound approximation:* $POS_S^l(X) = \{x \in U : [x]_{\widetilde{P}} \subseteq X\}$.
2. *The upper bound approximation:* $POS_S^u(X) = \{x \in U : [x]_{\widetilde{P}} \bigcap X \neq \emptyset\}$.
3. *The elementary set approximation:* $POS_S^e(X) = \bigcup_{\frac{|R_i \cap X|}{|R_i|} \geq \tau} R_i$, where τ denotes a threshold value ranging in $(0.5, 1]$ and $R_i \in U/\widetilde{CON}$, where each R_i is called an elementary set.

For all approximation methods stated above, the boundary region $BND_S(X)$ is equal to $POS_S^u(X) - POS_S(X)$, where $POS_S(X)$ stands for POS_S^* and $*$ is a wildcard symbol used to represent different methods of defining positive regions. The set $POS_S(X)$ is the largest and fully contains X when upper approximation is used, while it is the smallest and is properly included by X when lower approximation is used. In the general case, X can have elements not in $POS_S(X)$, and vice-versa.

2.1 Multi-Concept Classification Methods

Let $U/\widetilde{DEC} = \{X_1, X_2, \ldots, X_n\}$. A classification problem is described as generating a decision algorithm from S, $D_S(U/\widetilde{DEC})$, that relates elements of $U/\widetilde{CON}$ to that of $U/\widetilde{DEC}$. If $D_S(U/\widetilde{DEC})$ is a relation then it is called *an inconsistent decision algorithm;* otherwise, it is said to be *a consistent decision algorithm.* Observe that an inconsistent decision algorithm might be interpreted deterministically or non-deterministically. Since $POS_S(U/\widetilde{DEC}) = \bigcup_{X \in U/\widetilde{DEC}} POS_S(X)$, the extension of an approximation method to its counterpart in classification problem is straightforward. Similarly, the classification quality $\varphi_S(U/\widetilde{DEC})$ is equal to $\frac{1}{|U|} \sum_{i=1}^{n} |X_i| \mu_S(X_i)$. Percentage accuracy of classification of $D_S(U/\widetilde{DEC})$ is

$$\gamma_S((U/\widetilde{DEC}) = \frac{\sum_{i=1}^{n} |X_i \bigcap POS_S(X_i)|}{|U|}$$

The classification methods based on the approximation methods mentioned above are called respectively, lower classifier, upper classifier, and elementary set classifier. We are now ready to formally define the feature selection problem that we want to focus on.

2.2 The Feature Selection Problem

Let S/P denotes a substructure of S such that $S/P = (U, Q' = P \cup DEC, \bigcup_{a \in P} V_a, \rho')$, where $P \subseteq CON$, ρ' is a restriction of ρ to set $U \times q'$. We say that $CON - P$ is *θ-superfluous* in S iff

$$\varphi_{S/P}(U/\widetilde{DEC}) = \varphi_S(U/\widetilde{DEC})(1 - \theta),$$

where $0 \leq \theta \leq 1$. Similarly, we say that P is a *θ-reduct* of CON iff $CON - P$ is a *θ-superfluous* in S and no $P' \subset P$ is *θ-superfluous* in S/P. Note that if $\theta = 0$ we simply call them superfluous or reduct. As we have stated before, the feature selection problem is to choose a small subset of features that is necessary and sufficient to define the target concept(s). In terms of these new definitions, feature selection problem can be re-expressed as finding a θ-reduct of CON in S.

3 Upper Classifiers

In this paper, we investigate the feature selection problem in the context of upper classifiers. Our motivation is two fold. First, from the point of data-mining applications, knowledge discovery methods must deal with incomplete or noisy data. The lower classification method, a traditional rough set approach, induces a consistent decision algorithm that covers only certain data. On the other hand elementary set classifiers provide us with a good consistent approximation to unknown concepts when data is uncertain, at the expense of more demand on disk space. Second, one of the characteristics of data mining is that data is dynamic. Neither lower nor elementary set classification methods provide a basis for adaptive classifiers since they weed out some portion of background knowledge. On the other hand, upper classification methods assume that such a decision is a matter of how its decision algorithm is interpreted; that is, an upper classifier keeps track of inconsistency given in the background knowledge. This feature of upper classification methods enables us to develop a truly adaptive classifiers. Additionally upper classifiers can be interpreted to yield the same decisions as the classification method using only lower bounds or elementary sets[1]. We will elaborate this point in Section 6.

Since the feature selection algorithms that will be presented in next section use the upper classifier in their evaluation criterion, it is appropriate to present relevant properties of the upper classifier. Remark 1 is based on the fact that for all $P, B \subseteq CON$, if $P \supseteq B$ then $U/\widetilde{P} \subseteq U/\widetilde{B}$, that is, $U/\widetilde{P}$ is refinement of $U/\widetilde{B}$.

Remark 1 *Let $X \subseteq U$ and $\mu_S^u(X)$ be quality of upper bound approximation of X in S. Then,*

$$\forall P, B \subseteq CON, \quad P \supseteq B \Longrightarrow \mu_{S/P}^u(X) \geq \mu_{S/B}^u(X)$$

In other words, the quality of upper classification is monotonically decreasing function while a given feature set is reduced. With respect to the upper classification algorithm shown in

[1] Any $[x]_{\widetilde{CON}}$ in S is called elementary set.

```
Algorithm UCQ(S)
1.  U/DEC~ = PARTITION(S, DEC)
2.  U/CON~ = PARTITION(S, CON)
3.  k = |U/CON~| ; n = |U/DEC~|
4.  for(i = 0; i < n; i++)
5.      for(j = 0; j < k; j++)
6.          if(X_i ∩ R_j ≠ ∅)
7.              POS(X_i) = POS(X_i) ∪ R_j
8.  return φ_S(U/DEC~)
```

Table 1: The upper classification algorithm

Table 1, we prove in Theorem 1 that time complexity of upper classifier in the worst case is bounded by $O(lm^2)$, where l and m are the number of condition attributes and objects in S.

Theorem 1 *The computation of upper classifier in Table 1 takes $O(lm^2)$ in the worst case, where l and m are the number of condition attributes and objects in S.*

[Proof] Let $U/\widetilde{DEC} = \{X_0, X_1, \ldots, X_{n-1}\}$ and $U/\widetilde{CON} = \{R_0, R_1, \ldots, R_{k-1}\}$ denote the set of blocks of equivalence relations by the values of DEC and CON on U, respectively. The time complexity of the algorithm UCQ in the worst case is computed as follows: Step 1 and step 2 take $O(mn)$ and $O(lmk)$, respectively, by the lemma on the nature of the partition (Sever, 1995). Step 8 takes $O(mn)$, in the worst case, by the lemma on the computation of quality(Sever, 1995). The computational effort of Steps 4-7 in the worst case would be expressed as follows.

$$\sum_{i=0}^{n-1}\sum_{j=0}^{k-1}(|X_i| + |R_j|) \tag{1}$$

Since both $U/\widetilde{DEC}$ and $U/\widetilde{CON}$ are partitions of U, Eq. (1) is bounded by $O(m^2)$. Then time complexity of whole computation in Table 1 is equal to

$$\underbrace{O(mn)}_{Step\,1} + \underbrace{O(lmk)}_{Step\,2} + \underbrace{O(m^2)}_{Steps\,4-7} + \underbrace{O(mn)}_{Step\,8} \in O(mn + lmk + m^2)$$

Since either n or k would be at most m, the time complexity of UCQ in worse case is $O(lm^2)$. □

4 Feature Selection Algorithms

We have developed and experimentally evaluated four algorithms for feature selection. All these algorithms fall under the class of Sequential Backward Selection (SBS) algorithms. These methods are:

(1.) *Best fit SBS* (BFS),

(2.) *Hybrid Heuristic SBS* (HHS),

(3.) *Alternating Heuristic SBS* (AHS), and

(4.) *K-level Best SBS* (KBS).

Before presenting each method, it is worth asserting that the class of SBS algorithms satisfies 'sufficient' and 'necessary' conditions of feature selection problem as explained in the following. According to Remark 1 we already know that the process of pruning a feature set down does not yield a better classification quality. This fact guarantees that the family of stepwise backward algorithms satisfies 'sufficient' condition of feature selection problem. Also, it can easily be verified that there is no θ-superfluous attribute in the returned set of relevant attributes, F. The last repetition of outer loop (Steps 3-16) in BFS algorithm, given in Table 2, verifies that no $f \in F$ can be extracted from F, and yet we have $\varphi^u_{F-\{f\}}(U/\widetilde{DEC}) \geq \varphi^u_S(U/\widetilde{DEC}) * (1-\theta)$. This concludes 'necessary' part of our claim in case of BFS algorithm, but the same reasoning can be carried out for the others.

Remark 1 also enables us to use *branch and bound* algorithm to find the smallest θ-reduct of relevant features. For a given decision table S and threshold value θ, let the state of a node in search space be made up of a subset of condition attributes CON, denoted by F, and quality of upper classification, denoted by $\varphi(POS^u_S(\tilde{F}, \widetilde{DEC}), \widetilde{DEC})$. State of the root node is, defined by CON and $\varphi(POS^u_S(\widetilde{CON}, \widetilde{DEC}), \widetilde{DEC})$. The feasibility condition for a current node to be expanded is imposed as $\varphi(POS^u_S(\tilde{F}, \widetilde{DEC}), \widetilde{DEC}) \geq \varphi(POS^u_S(\widetilde{CON}, \widetilde{DEC}), \widetilde{DEC}) * (1-\theta)$. Then, the objective function is to maximize the feasibility condition. Now we are going to describe the four algorithms of the family of sequential backward algorithms.

4.1 Best Fit SBS Algorithm (BFS)

In BFS, given in Table 2, one starts with the set of all features and features are removed one at a time. To justify that BFS algorithm finds a θ-reduct of CON in S, we recall that the quality of upper classifier decreases as the feature set is pruned down (Deogun, Raghavan, & Sever, 1995; Sever, 1995). In BFS algorithm, we initialize the current node to the root node. That is, the state of root node corresponds to all the features in the given set, F, and the *CurrentQuality* is the quality of classification using these features. We select as the current node a feasible node whose quality of classification is the highest (best fit) in that level and then expand this node to get the nodes at the next level. This process is repeated until the node has a feature set of size at most one or no node at that level meets the feasibility condition. The time complexity analysis of this algorithm is given by theorem below.

Theorem 2 *The time complexity of BFS algorithm, given in Table 2, in the worst case is equal to $O(l^2 J)$, where $l = |CON|$ is the size of CON and J is the computational effort to evaluate the classification quality of a given S.*

```
Algorithm BFS(S, θ)
1.  F = CON
2.  Threshold = multiply the quality of F by (1 − θ)
3.  while |F| > 1
4.          MinQuality = Threshold
5.          Found = FALSE
6.          for each c ∈ F
7.              F = F − {c}
8.              evaluate the quality of F and assign to CurrentQuality
9.              if (CurrentQuality > MinQuality)
10.                 MinQuality = CurrentQuality
11.                 KeepAttribute = {c}
12.                 Found = TRUE
13.                 F = F ∪ {c}
14.         if(Found == true)  F = F − KeepAttribute
15.             else break
16. return F
```

Table 2: The Best Fit SBS

Proof. Step 2 takes computational effort of $J = O(lm^2)$ as proved in by Theorem 1. Assume there are t θ-superfluous features in S, where t varies over $[0, 1, \ldots, l-1]$. Let F_j be a set of condition attributes in S/F_j at the jth iteration of the outer loop, where $j \in [l, l-1, \ldots, l-t]$. Similarly, F_{ji} is obtained by removing an attribute c_i from $F_j = \{c_0, c_1, \ldots, c_i, \ldots, c_j\}$ at step 7. Thus, the time complexity of BFS algorithm is equal to

$$\underbrace{O(lm^2)}_{step\,2} + \underbrace{\sum_{j=1}^{l} i}_{steps\ 3-16} \quad . \tag{2}$$

That proves that the time complexity is $O(l^2 J)$ or $O(l^3 m^2)$ □

4.2 Hybrid Heuristic SBS Algorithm (HHS)

For the HHS algorithm given in Table 3, also, the current node is initialized to root node. The current node is expanded to its successors which have one less condition attributes than its predecessors. We then make the first node that meets feasibility condition (first fit) as the current node. If none of the successor nodes meets the feasibility condition, then we backtrack only one level and try to find another current node. We continue this process until the termination condition for the search process is reached. The termination condition for the search process is either the cardinality of condition attributes becoming one or no feasible nodes are left to explore in this controlled search space.

```
Algorithm HHS(S, θ)
1.  F = CON
2.  backtrack = 0
3.  Threshold = multiply the quality of F by (1 − θ)
4.  while |F| > 1
5.    Found = FALSE
6.    for each c ∈ F,
7.        evaluate the quality of F − {c}
8.        if quality of F − {c} > Threshold then
            c_min = c
            F = F − {c_min}
            Found = TRUE
            backtrack = 0
            exit the for loop
9.    endfor
10.   if not Found
11.      if backtrack == 0
            backtrack = backtrack + 1
            F = F ∪ {c_min}
         else return F
      else;
12. endwhile
13. Return F
```

Table 3: The Hybrid Heuristic SBS

The time complexity of HHS algorithm, given in Table 3, in the worst case is equal to $O(l^2J)$, where l is the size of CON and J is the computational effort to evaluate the classification quality of a given S.

Step 7 takes computational effort of $J = O(lm^2)$ as proved in by Theorem 1. Step 4 through step 12 takes $2\sum_{i=1}^{l} i + C$, where C is the cost to backtrack, and i is the cost to do the comparison again, when backtracked. This give the complexity of $O(l^2J)$ or $O(l^3m^2)$.

4.3 Alternating Heuristic SBS Algorithm (AHS)

AHS algorithm initializes current node to root node. Then it alternatively uses two heuristics to determine the current node at alternate levels in the search tree. One of the heuristics is BFS, defined earlier. In the other heuristic, the first fit strategy is used to determine the current node at a level. This process is continued until either the node has only one condition attribute or none of the nodes at the current level meets the feasibility condition. The algorithm is given in Table 4.

The time complexity of AHS algorithm, given in Table 4, in the worst case is equal to $O(l^2J)$, where l is the size of CON and J is the computational effort to evaluate the classification quality of a given S.

This algorithm alternatively uses the Best search algorithm and the First search algorithm at alternate levels. We have already analyzed these two algorithms. The complexity of each one of these is $O(l^2J)$. Thus, the worst time complexity is $O(l^2J)$ or $O(l^3m^2)$.

4.4 K-level Best SBS Algorithm (KBS)

KBS algorithm divides the search space from root node level (or level-1) to leaf node level (or level-$|CON|$) into $\lceil \frac{|CON|}{K} \rceil$ groups. That is each group of levels has K levels with an exception of last group which will have less than or equal to K levels. We call root node as the current node to begin with. Then we determine the best node in first group of levels by exhaustive search and then kill all other nodes except the best node at the end of searching this group of levels. Now this node becomes the root node for next group of levels. This process of finding the best node and eliminating all other nodes at the end of each group of levels is repeated until the node has only one condition attribute or no feasible nodes are left in the current group of levels to expand. The algorithm is given in Table 5.

The time complexity of KBS algorithm, given in Table 5, in the worst case is equal to $O(l^{K+1}J)$, where $l = |CON|$, K is the number of levels in each group, and J is the computational effort to evaluate the classification quality.

Step 10 takes computational effort of $J = O(lm^2)$ as proved in by Theorem 1. Given that each group consists of K levels and the size of the feature set is l, the computational effort required for each group of levels is l^K. We have $\frac{l}{K}$ levels. Hence the total computational effort is $\frac{l}{K}l^K$. Thus the worst time complexity is $O(l^{K+1}J)$. If K is controlled as 2, then the KBS algorithm has a complexity of $O(l^3J)$. Our experimental study is based on this value of K. It may be noted that $K = 1$ makes the KBS algorithm identical to BFS algorithm with the complexity of $O(l^2J)$.

5 Experimental Setup and Results

We have performed experiments with two goals. We wanted to study the performance of family of Stepwise Backward Selection algorithms on learning data and some real-life data. Then we wanted to compare the performance of our algorithms with that of some other approach to classification. Hence, we have divided this section into two subsections. In first subsection, we discuss the experimental setup that we have used to study the performance of the algorithms along with some important results. In next subsection, we discuss some background needed to understand our experiment on data of highly selective vagotomy for duodenal ulcer, followed by experimental details and results of comparison of our approach of upper classification to that of lower classification.

Algorithm AHS(S, θ)

```
1.  F = CON
2.  Threshold = multiply the quality of F by (1 − θ)
3.  Level = 1
4.  while |F| > 1
5.    Found = FALSE
6.    if level is odd then
7.       for each c ∈ F,
8.           evaluate the quality of F − {c} and assign to CurrentQuality
9.           if quality of F − {c} > Threshold then
                KeepAttribute = c
                exit the for loop
             endif
          endfor
      else
          MinQuality = Threshold
          for each c ∈ F,
             evaluate the quality of F − {c} and assign to CurrentQuality
             if quality of F − {c} > Threshold then
                MinQuality = CurrentQuality
                KeepAttribute = c
                Found = TRUE
             endif
          endfor
      endifthenelse
10.   if Found
          F = F − KeepAttribute
      else exit the while loop
11.   level level + 1
12. endwhile
13. return F
```

Table 4: The Alternating Heuristic SBS

```
Algorithm KBS(S,θ)
1.  F = CON
2.  K = 2
3.  Nlevel = ⌈|CON|/K⌉
4.  Threshold = multiply the quality of F by (1 − θ)
5.  For kindex = 1 to Nlevel do
6.    Found = FALSE
7.    MinQuality=Threshold
8.    for j = K to 1
9.        for each {c_i}_{1≤i≤j}
10.         evaluate the quality of F − {c_i} and assign to CurrentQuality
11.         if CurrentQuality > MinQuality then
              MinQuality = CurrentQuality
              KeepAttribute = {c_i}
              Found = TRUE
            endif
          endfor
12.       if Found then exit the for loop
      endfor
13.   if Found then F = F − KeepAttribute
14.   if |KeepAttribute| < K
         return F
15. endfor
```

Table 5: The K-level Best SBS

5.1 Comparison of Feature Selection Algorithms

A set of experiments that use traditional machine learning data sets, and a real life data set, were designed and used to test the performance of upper classifiers based on these algorithms. The data sets used are from the UC Irvine's machine learning repository (Murphy and Aha, 1995), except for parity 5+10, XOR, and duodenal ulcer data set. Parity and XOR data sets are artificial data sets where the parity 5+10 is the concept of parity of five bits with ten irrelevant bits and XOR is the concept of parity 'exclusive or' with thirteen irrelevant bits. The duodenal ulcer data set has earlier been used by Pawlak et al. (1986) and is the data about 77 patients with duodenal ulcer treated by highly selective vagotomy (HSV). If the data set has a test set, then we use that, otherwise, we randomly choose two third of the objects from each class of the corresponding data set.

Table 6 shows the data sets used with their number of attributes, training and test set sizes. Table 7 and Table 8 show the percentage of accuracy (γ_S) for different data sets and different methods using *same* data sets for test and training and *different* data sets for test and training. Before percentage of accuracy is computed for an upper classifier, an inconsis-

data set	*Size*		*No. of Attributes*
	Training set	*Test set*	
Glass	66	148	9
Breast cancer	211	488	9
Parity 5+10	226	524	15
Iris	45	105	4
Monk 1	124	432	6
Monk 2	169	432	6
Monk 3	122	432	6
Vote	132	303	16
Soybean-s	15	32	35
XOR	226	524	15
Mushroom	2439	5685	22
Hepatitis	47	108	19
Duodenal ulcer	21	56	11

Table 6: Test and Training sets for different various data sets.

tent decision algorithm must be interpreted. As we explain in section 6, the interpretation can be deterministic or non-deterministic. In our experiments, the deterministic interpretation is applied. The test sets used in the first group of experiments are used both as training and test sets in the second group of experiments. We call the first group as *Predictive Experiments* and the second group as *Upperbound Experiments*. Upperbound Experiments are seen as providing upper bound on the accuracy achievable by the Predictive Experiments. These tables give the accuracies of the upper classifier without and with the feature selection algorithms. Table 9 and Table 10 present the feature subsets and percentage of reduction in feature sets for different data sets using different feature selection algorithms. Table 10 refers to the results of Predictive Experiments whereas Table 9 refers to the results of Upperbound Experiments. Table 11 and Table 12 show the number of nodes visited for different data sets using different methods. Table 11 has the result for Predictive Experiments and Table 12 has the result of Upperbound Experiments. Note that for all training sets we set the threshold θ to 0.5%.

When a data set contains a missing value, we assume that it is a non-quantitative value and distinct from any other value, including other occurrences of missing values. One may raise the objection to the feasibility of our assumption on the base that missing values may come from unknown attribute values. We would like to point out that such treatment of missing values does not harm our experiment because our purpose in this experiment is to simply compare the performance of the upper classifier with and without the family of SBS algorithms. Given a test set, the accuracy of an upper classifier is defined by $\gamma_S^u(U/\widetilde{DEC})$. When the description of a given object does not match to known concepts we use 5NNR classification scheme with Euclidean distance function to determine the five closest concepts. The difference between two values of an attribute are computed as suggested in *Relief* algorithm (Kira & Rendell, 1992); that is, the difference between two non-quantitative values

data set	*Accuracy in %*				
	UC	*UC+BFS*	*UC+AHS*	*UC+HHS*	*UC+KBS*
Glass	100	100	100	100	100
Breast cancer	100	99.6	99.6	99.6	99.6
Parity 5+10	100	100	98.7	98.7	100
Iris	92.2	95.2	95.2	95.2	95.2
Monk 1	100	83.3	75	75	83.3
Monk 2	100	84.7	78.9	78.9	84.7
Monk 3	100	100	100	100	100
Vote	100	99.3	99.3	99.3	99.3
Soybean-s	100	100	100	100	100
XOR	100	100	100	100	100
Hepatitis	100	100	100	100	100
Duodenal ulcer	100	98.2	98.2	98.2	98.2

Table 7: Comparison of classification accuracies of UC and other algorithms on real data sets using Upperbound Experiments.

data set	*Accuracy in %*				
	UC	*UC+BFS*	*UC+AHS*	*UC+HHS*	*UC+KBS*
Glass	64.3	36.5	53.4	50	36.5
Breast cancer	96.4	91.8	93.1	95	91.8
Parity 5+10	55.5	100	52.9	55.9	100
Iris	92.4	94.3	94.3	94.3	94.3
Monk 1	86.1	100	70.4	70.4	100
Monk 2	74.8	67.8	67.8	67.8	67.8
Monk 3	90	93.5	93.8	93.8	93.5
Vote	91.1	95.4	95.7	93.4	95.4
Soybean-s	96.9	61.1	100	100	100
XOR	78.1	100	100	100	100
Mushroom	99.5	99.7	99.6	99.8	99.7
Hepatitis	75.9	77.8	81.5	78.7	77.8
Duodenal ulcer	58.9	58.9	53.6	33.9	53.6

Table 8: Comparison of classification accuracies of UC and other algorithms on real data sets using Predictive Experiments.

data set	*UC+BFS*		*UC+AHS*		*UC+HHS*		*UC+KBS*	
	Attr. Subset	*%Re.*	Attr. Subset	*%Re.*	Attr. Subset	*%Re.*	Attr. Subset	*%Re.*
Glass	0,1	78	4,5	78	5,6	78	0,1	78
Breast	0,2,3,4	56	3,4,5,6	56	5,6,7,8	56	0,2,3,4	56
Parity	0,1,2,3,4	67	1-5,7-14	13	1-2,4-14	13	0,1,2,3,4	67
Iris	2,3	50	2,3	50	2,3	50	2,3	50
Monk 1	0,1	67	4	83	4	83	0,1	67
Monk 2	0,1,2,3,5	17	1,2,3,4,5	17	1,2,3,4,5	17	0,1,2,3,5	17
Monk 3	1,3,4	50	1,3,4	50	1,3,4	50	1,3,4	50
Vote	0-3,9-11	56	2,3,6,7,9		2,3,10-12		0-3,9-11	56
			11,14,15	50	14,15	56		
Soy-s	0,2,3	91	20,21	94	21,28	94	0,2,3	91
XOR	3,8	87	3,8	87	3,8	87	3,8	87
Hepatitis	0,2,4,5,7-9	63	8,9,10,11,13	74	12,15,17,18	79	0,2,4,5,7-9	63
Duodenal	3,4,5,6		1,2,4,5		4,5,6,7		1,2,4,5	
Ulcer	7,9,10	36	7,9,10	36	9,10,11	36	7,9,10	36

Table 9: Comparison of reduction in attribute sets for heuristics for Upperbound Experiments.

is one if they are different and zero otherwise, and the difference between two quantitative values is normalized into the interval [0,1]. Thus, no domain knowledge on data sets is exploited, except the type of attributes, e.g., quantitative or non-quantitative.

We first consider results from Table 7 and Table 8. Except for Glass, Monks, and Hepatitis data sets, the performance obtained in Predictive Experiments approach those in the case of Upperbound Experiments. This suggests that for Glass, Monks, and Hepatitis data sets, the training sets are not well designed. Since reduced training sets of XOR and Parity data sets were complete, in the sense that they contained all combinations of corresponding relevant attributes, $BFS+UC$ and $KBS+UC$ performed much better than UC did. For XOR data set, $AHS+UC$ and $HHS+UC$ also performed much better than UC did. The only data sets for which $BFS+UC$ performed worse than UC were small soybean, and glass data sets. $AHS+UC$, and $HHS+UC$ performed worse than UC on glass, monk1, and monk2 data sets. $KBS+UC$ performed poorly only in the case of glass data set. In the case of Upperbound Experiments, only monk1 data set performed relatively worse in the case of all the heuristics.

Considering Table 9 and Table 10, on XOR data set, all algorithms found the smallest reduct of the attributes. On the other hand, on Parity data sets, BFS and KBS algorithms found smallest reduct of attributes. The average percentage of the reduction in features of data sets is 68.83% in the cases of $BFS+UC$ and $AHS+UC$, 60.25% in the case of $AHS+UC$ and 59.07% in the case of $HHS+UC$ in the case of Predictive Experiments. The average percentage of the reduction in features of data sets is 62% in the cases of $BFS+UC$ and $AHS+UC$, 59.27% in the case of $AHS+UC$ and 60.27% in the case of $HHS+UC$

data set	*UC+BFS*		*UC+AHS*		*UC+HHS*		*UC+KBS*	
	Att. Subset	*%Red.*	Att. Subset	*%Red.*	Att. Subset	*%Red.*	Att. Subset	*%Red.*
Glass	0	89	5,6	78	5,6	78	0	89
Breast	0,2	89	3,5,6	67	5,6,7,8	56	0,2	89
Parity5+10	0,1,2,3,4	67	2-9,11-14	20	2,3,5,6		0,1,2,3,4	67
					7,9-14	27		
Iris	2	75	2	75	2	75	2	75
Monk 1	0,1,4	50	1,2,3,4,5	17	1,2,3,4,5	17	0,1,4	50
Monk 2	1,2,3,4,5	17	1,2,3,4,5	17	1,2,3,4,5	17	1,2,3,4,5	17
Monk 3	0,1,3,4	33	1,3, 4,5	33	1,3,4,5	33	0,1,3,4	33
Vote	1,2,3,6,8	69	3,7,8,9,10	69	3,13,14,15	75	1,2,3,6,8	69
Soybean-s	0,2,3,5	89	20,21	94	26,27,34	91	0,2,3,5	89
XOR	3,8	87	3,8	87	3,8	87	3,8	87
Mushroom	2,4,9		4,9,10,14	82	8-10,15		2,4,9	
	10,13	77			19-21	68	10,13	77
Hepatitis	0,2,4	84	8,10,13	84	15,17,18	84	0,2,4	84
Duodenal Ulcer	5,6,7	73	1,2,3	73	6,8,9,11	64	1,2,4	73

Table 10: Comparison of reduction in attribute sets for heuristics for Predictive Experiments.

in the case of Upperbound Experiments. When Parity and XOR were ignored, the average percentage of the reduction in features of data sets is 68.20%. These results indicate that all the proposed algorithms find a small subset of features that are sufficient to define target (or unknown) concepts.

Table 11 and Table 12 give an indication of how fast the feature selection process is, for different feature selection algorithms. These feature selection methods take time proportional to the number of the nodes they have to visit in order to obtain the smallest possible subset of attributes. Generally speaking, the number of nodes visited is a function of training data sets, number of initial attributes, and the feature selection method used. For example, if the initial number of attributes are fewer in number, then $UC+AHS$ and $UC+HHS$ are faster for all the data sets except the parity, monk1, monk2, and monk3. We also notice that, the algorithms $UC+BFS$ and $UC+KBS$ have performed equally well, in terms of accuracy and the percentage reduction in the size of the feature set, for all the data sets. However, $UC+KBS$ is more efficient as can be seen by the number of nodes visited by this algorithm compared to that visited by the algorithm $UC+BFS$. Hence, if initial set of features is large, then $UC+KBS$ can be recommended as a suitable choice. However, if feature set is small, then either $UC+ABS$ or $UC+HHS$ can be used depending on the data set used.

data set	*Number of nodes visited*			
	UC+BFS	*UC+AHS*	*UC+HHS*	*UC+KBS*
Glass	45	26	21	25
Breast cancer	44	30	25	26
Parity 5+10	110	149	198	60
Iris	10	6	7	6
Monk 1	18	26	31	13
Monk 2	11	26	31	11
Monk 3	15	22	27	10
Vote	126	89	45	71
Soybean-s	624	325	62	324
XOR	119	69	39	65
Mushroom	243	149	123	131
Hepatitis	187	105	32	99
Duodenal ulcer	41	63	43	35

Table 11: Comparison of number of nodes visited by the algorithms on real data sets for Predictive Experiments.

5.2 Comparison of Performance of Upper and Lower Classifiers on Duodenal Ulcer data set

The data on highly selective vagotomy for duodenal ulcer consists of 77 patients. Each patient has been described using 11 characteristics (attributes). The data has been described in detail in Pawlak, Slowinski, and Slowinski (1986). Based on these attributes, the patients have been classified into four classes of excellent, very good, satisfactory, and unsatisfactory.

In order to use the same basis for comparison of performance of upper and lower classifiers, we have conducted our experiments in following phases. First, we analyzed the data set and results given in Pawlak, Slowinski, and Slowinski (1986), by using the lower approximation (LC) method. They have derived three reduced feature sets, denoted by case E, G, and H respectively. They have studied the percentage accuracy of classification for each of these feature sets using lower classifier. We then, computed $1-\gamma_S^l(U/\widetilde{DEC})$ for each of these cases of E, G, and H as $12\%, 9\%$, and 9% respectively. We decided to use 9% for the value of θ, as it was obtained in two out of three cases, and it will allow fewer misclassification when the judgement on classification will be made.

Using θ equal to 9%, we conducted the detailed experiment on this data set. We derived reduced feature set and computed the accuracy of upper classifiers for each of the algorithms, described earlier. Again, for the purposes of computing the percentage of accuracy, an inconsistent decision algorithm is interpreted deterministically (see section 6). The result is tabulated in Table 13 and indicates a higher accuracy of classification for upper classifiers compared to lower classifiers for all the cases (E, G, and H) of duodenal ulcer data. The fact that upper classifier gave better accuracy than lower classifier for all the cases can be explained as follows: In case of lower approximation, the patients were classified as unknown when they were in boundary region (in case of doubt). In the upper approximation, even

data set	*Number of nodes visited*			
	UC+BFS	*UC+AHS*	*UC+HHS*	*UC+KBS*
Glass	44	26	21	26
Breast cancer	39	33	25	25
Parity 5+10	110	184	212	60
Iris	9	8	8	6
Monk 1	20	12	9	12
Monk 2	11	26	31	11
Monk 3	18	17	28	13
Vote	115	120	117	67
Soybean-s	627	325	62	323
XOR	119	69	16	65
Hepatitis	169	116	47	91
Duodenal ulcer	69	45	92	27

Table 12: Comparison of number of nodes visited by the algorithms on real data sets for Upperbound Experiments.

Algorithms	*% Accuracy*	*Reduced Feature Set*	*% Reduction*	*# of Nodes Visited*
LC with E	88	4,5,6,9,10	55	Not Available
LC with G	91	3,4,5,6,9	55	Not Available
LC with H	91	3,4,6,9,10	55	Not Available
UC+BFS	96	2,4,5,7,9,10	45	51
UC+AFS	96	4,5,6,9,10	55	52
UC+HHS	96	4,5,8,9,10,11	45	62
UC+KBS	96	3,4,5,6,9	55	32

Table 13: Comparison of Accuracies, Reduced Feature Set and Number of Nodes Visited using various algorithms for Upper Classifiers.

in the case of boundary region we classify the patient as belonging to the concept. In some applications, it may be considered risky to have unknown cases as belonging to the concept. However, we can adjust the parameters s_1 and s_2 in the expression for $\mu_S(X)$. At present, we are using 0.5 for both s_1 and s_2.

We also computed the percentage reduction in the size of feature set, and the number of nodes visited. This result is also given in Table 13. As we see in this table, the pruned set of attributes is same for lower approximation (Case E) and the upper approximation using AFS algorithm as given in Table 13. The percentage reduction in feature set is also found to be the same when we used $UC + AFS$ and $UC + KBS$ algorithm as that of lower classifier (cases E, G, and H). Also $UC + KBS$ has again given best performance for the number of nodes visited, as we have seen in the case of other data sets. Hence it is the fastest algorithm for this data set also. For data set on duodenal ulcer, $UC + KBS$ has given best performance out of all the algorithms with respect to accuracy of classification, feature set

reduction, and the computational time.

6 Extended Decision Table

The classification methods are data driven methods, and hence, it is unrealistic, in most cases, to expect that the decision rules obtained from a snapshot/part of a database will stand up no matter how the database changes over time. Therefore, one usually associates some frequency information with the decision rules to make them incremental. We call such information *incremental information*. They are not related with the contents of a decision table but provide information about the accuracy of each rule or the likelihood of its occurrence. Let CON_SAT be an event consisting of objects that satisfy the condition part of a decision rule in a *base decision table*[2], and let $RULE_SAT$ be an event consisting of objects being classified correctly by the same decision rule. Then incremental information of a decision rule is composed of the sizes of CON_SAT and $RULE_SAT$.

To incorporate incremental information into the decision table, we introduce the notion of Extended Decision Table (EDT) in which each row corresponds to a decision rule. We use EDT to represent a decision algorithm that is induced such that the antecedent part of each rule corresponds to only one elementary set in the base decision table. Details of EDT are omitted for lack of space and can be found in Deogun, Raghavan, and Sever (1994). The important point to note, however, is that EDT can be useful in three places.

First, EDT enables us compute the *accuracy measure of a decision rule*. Second, EDT is adaptive because any data entry into (or update on) its base decision table is easily propagated to it. Observe that the feedback channel we consider here is between EDT and base decision table. Third, EDT enables us to interpret inconsistent decision algorithms either deterministically or nondeterministically as explained here. A deterministic interpretation of an inconsistent EDT would be *to always select the row whose accuracy measure (is proportional to RULE_SAT) is the maximum among the repeating rows for a given* $x \in U$. On the other hand, a nondeterministic interpretation of inconsistent EDT would be *to select a row randomly when there is a conflict in the value of the consequent part among the rows for a given* $x \in U$. The random choice can be *biased* in proportion to the relative values of the approximation accuracy of the rows or other criteria (e.g. in proportion to the number of training samples associated with various values of the consequent).

7 Conclusion

We have proposed and experimentally evaluated four feature selection algorithms that find a θ-reduct of a given feature set in polynomial time. We have used learning data sets and a real data set on duodenal ulcer. Based on our experiments, we show that we can find θ-reduct by only visiting a small portion of the state space. In one group of experiments, the same data is used to both train and test a classification system, in order to determine an

[2]A base decision table is the one from which the decision algorithm is obtained.

upper bound on the performance of that system. It has been shown that results of predictive experiments for most data sets are very close to these ideal conditions. These algorithms can be used with any method of deriving a classifier where the quality of classification is a monotonically decreasing function while feature set is reduced. In our comparison of the performance of upper classifiers with those of lower classifiers, where only duodenal ulcer data set was used, we found upper classifiers to perform better than lower classifiers. It is also shown that an upper classifier can be summarized at a desired level of abstraction by using extended decision tables. We also point out that an inconsistent decision algorithm can be interpreted as if it were a consistent decision algorithm.

References

Bollmann, P. & Cherniavsky, V. S. (1981). Measurement theoretical investigation of the mz-metric, *Information Retrieval Research* (Oddy, R. N., Robertson, S. E., VanRijsbergen, C. J. & Williams, R. W., eds.), Boston:Butterworths, 256-267.

Chang, C. Y. (1973). Dynamic programming as applied to feature subset selection in a pattern recognition system, *IEEE Trans. Syst., Man, Cybern.*, Vol. SMC-3, 166-171.

Choubey, S. K., Deogun, J. S., Raghavan, V. V. & Sever, H. (1996). A Comparison of Feature Selection Algorithms in the Context of Rough Classifiers, *Proceedings of Fifth IEEE International Conference on Fuzzy Systems*, Vol. 2, (New Orleans, LA), 1122-1128.

Deogun, J. S., Raghavan, V. V. & Sever, H. (1994). Rough set based classification methods and extended decision tables, *Proceedings of the International Workshop on Rough Sets and Soft Computing*, (San Jose, California), 302-309.

Deogun, J. S., Raghavan, V. V. & Sever, H. (1995). Exploiting upper approximations in the rough set methodology, *The First International Conference on Knowledge Discovery and Data Mining*, (U. Fayyad and R. Uthurusamy, eds.), (Montreal, Quebec, Canada), 69-74.

Deogun, J. S., Raghavan, V. V., Sarkar, A., & Sever, H. (1996). Data mining: Research trends, challenges, and applications, in *Rough Sets and Data Mining: Analysis of Imprecise Data* (Lin, T. Y. & Cercone, N., eds.), Boston, MA: Kluwer Academic Publishers, 1-28.

Devijver, R. A. & Kittler, J. (1982) *Pattern Recognation: A statistical approach*, London: Prentice Hall.

Duda, R. O. & Hart, P. E. (1973). *Pattern Classification and Scene Analysis*, John Wiley & Sons.

Fayyad, U. M. & Irani, K. B. (1992). The attribute selection problem in decision tree generation, *Proceedings of AAAI-92*, AAAI Press, 104-110.

Fayyad, U. M. (1996). Data mining and knowledge discovery: Making sense out of data, *IEEE Expert*, no. 10, 20-25.

James, M. (1985). *Classification Algorithms*, New York, NY:John Wiley & Sons.

Kira, K. & Rendell, L. (1992). The feature selection problem: Tradational methods and a new algorithm, *Proceedings of AAAI-92*, AAAI Press, 129-134.

Lin, T. Y. & Cercone, N., eds., (1996). *Rough Sets and Data Mining: Analysis of Imprecise Data*, Boston, MA: Kluwer Academic Publishers.

Matheus, C. J., Chan, P. K. & Piatetsky-Shapiro, G. (1993). Systems for knowledge discovery in databases, *IEEE Trans. on Knowledge and Data Engineering*,vol. 5, no. 6, 903-912.

Miller, A. J. (1990). *Subset Selection in Regression*, Chapman and Hall.

Modrzejewski, M. (1993). Feature selection using rough sets theory, *Machine Learning: Proceedings of ECML-93* (Brazdil, P. B., ed.), Springer-Verlag, 213-226.

Murphy, P. M. & Aha, D. W. UCI repository of machine learning databases, For information contact m-lrepository@ics.uci.edu.

Pal, S. K. & Chakraborty, B. (1986). Fuzzy set theoretic measure for automatic feature evaluation, *IEEE Trans. Syst., Man, Cybern.*, Vol. SMC-16, no. 5, 754-760.

Pawlak, Z. (1982). Rough sets, *International Journal of Computer and Information Sciences*, Vol. 11, 341-356.

Pawlak, Z. (1991). Rough sets: theoretical aspects of reasoning about data. Kluwer.

Pawlak, Z., Grzymala-Busse, J., Slowinski, R., & Ziarko, W. (1995). Rough sets. CACM, 38(11), 89-95.

Pawlak, Z., Slowinski, K. & Slowinski, R. (1986). Rough classification of patients after highly selective vagotomy for duodenal ulcer, *International Journal of Man-Machine Studies*, Vol. 24, 413-433.

Sever, H. (1995). *Knowledge Structuring for Database Mining and Text Retrieval Using Past Optimal Queries*, PhD thesis, The University of Southwestern Louisiana.

Simoudis, E. (1996). Reality check for data mining, *IEEE Expert*, no. 10, 26-33.