Exploiting Upper Approximation in the Rough Set Methodology

Jitender S. Deogun
The Department of Computer Science
University of Nebraska
Lincoln, NE 68588, USA
e-mail: deogun@cse.unl.edu

Vijay V. Raghavan[†] **and Hayri Sever**[‡]
The Center for Advanced Computer Studies
University of Southwestern Louisiana
Lafayette, LA 70504, USA
e-mail: raghavan@cacs.usl.edu[†]
e-mail: hsr@swamp.cacs.usl.edu[‡]

Abstract

In this paper, we investigate enhancements to an upper classifier – a decision algorithm generated by an upper classification method, which is one of the classification methods in rough set theory. Specifically, we consider two enhancements. First, we present a stepwise 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 prove that the stepwise 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 an acceptable degradation in the quality of an upper classifier. Second, to make an upper classifier adaptive, we associate it with some kind of frequency information, which we call incremental information. An extended decision table is used to represent an adaptive upper classifier. It is also used for interpreting an upper classifier either deterministically or nondeterministically.

Keywords- Rough sets, feature selection, adaptive classifier.

Introduction

Feature selection is the problem of choosing a small subset of features that is necessary and sufficient to describe a target concept(s). The importance of feature selection in a broader sense is due to the potential it offers for speeding up the processes of both concept learning and object classification, 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 time complexity of J is low. Greedy approaches like stepwise 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 (Devicver & 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.

Instead of adapting near-optimal solutions for feature selection problem from other disciplines, we take advantage of the fact that the quality of an upper classifier worsens as the feature set is pruned down. Note that an upper classifier is a decision algorithm (or rules) generated by the upper classification method for a given data (or training) set. We present a stepwise backward selection algorithm to find a small subset of features that is sufficient and necessary to define target concepts with respect to a given threshold. The threshold value indicates how much degradation one is willing to allow in the quality of an upper classifier. Controlled threshold value is inspired by Salzberg's CSS algorithm (Salzberg 1992). Even though our feature selection algorithm is developed as a pre-processing stage for rough classifiers, it can certainly be integrated to any other data analysis technique.

The theory of rough sets in either algebraic or probabilistic approximation spaces has been used for a number of real life applications; namely, in medicine, phar-

macology, industry, engineering, control systems, social sciences, switching circuits, image processing, etc., (Raghavan & Sever 1995). In this article we consider classification methods only in algebraic approximation spaces, which do not require any preliminary or additional information about data as opposed to rough sets in probabilistic approximation spaces. We use upper classifiers and decision tables to address some problematic aspects of handling very large, redundant, incomplete, noisy, and dynamic data (Matheus, Chan, & Piatetsky-Shapiro 1993).

The *dynamic* characteristic of the data requires to design an incremental method and separate the summary and the result of a method from one to another. For example, in the context of rough classification, the strength of a possible decision rule (Ziarko 1991) is a part of the summary of the decision algorithm. Similarly, a further refinement of antecedent parts of rules in a decision algorithm is also a part of the summary, if the decision algorithm is *persistent* in the system and the background knowledge from which the decision algorithm has been induced is dynamic. In the rough set literature, the terms 'inconsistent' and 'nondeterministic' decision algorithms (or rules) are used interchangeably (Slowinski & Stefanowiski 1995; Pawlak 1986), though they are different concepts. As shown later, inconsistent decision algorithms, under an appropriate representation structure, can be interpreted deterministically as well as nondeterministically. This is an important distinction, particularly when the background knowledge is *incomplete and dynamic.*

Classification Methods in Rough Set Theory

A decision table can be viewed as an application of rough set theory such that each object is described by a set of attributes. It is defined as a quadruple $S = (U, Q = CON \cup DEC, V, \rho)$ where: U is the finite set of objects; Q is the the union of condition, denoted by CON, and decision attributes, denoted by DEC; V is the union of domains of attributes in Q; and $\rho : UXQ \Rightarrow V$ is a total description function. For all $x \in U$ and $a \in Q$, $\rho(x,a) = \rho_x(a)$. For given $P \subseteq Q$, let $|P| = r$. We introduce following notations.

(i) $\bar{\rho}_x\ (P)$ denotes extended version of total description function, that is,

$\bar{\rho}_x\ (P) =< v_1, v_2, \ldots, v_r >$, where $v_i = \rho_x(P_i)$.

(ii) $\widetilde{P}$ denotes the equivalence relation on U defined by the values of P, that is,

$\widetilde{P} = \{(x,y) :\ x,y \in U \wedge \bar{\rho}_x\ (P) = \bar{\rho}_y\ (P)\}$.

(iii) $[x]_{\widetilde{P}} = \{y\ :\ \bar{\rho}_x\ (P) = \bar{\rho}_y\ (P)\}$ denotes a block of $\widetilde{P}$.

(iv) $U/\widetilde{P}$ denotes a set of blocks of $\widetilde{P}$.

It is usual to call $[x]_{\widetilde{CON}}$ an elementary set and $[x]_{\widetilde{DEC}}$ a concept of interest. For notational simplicity we denote a concept $[x]_{\widetilde{DEC}}$ by X. 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, or

c) *unknown.*

In the following, we define corresponding sets of X in S for each answer. Let $POS_S(X)$ be a set of objects each of which is considered as a member of the concept X by the decision algorithm $D_S(X)$. Let $BND_S(X)$ be a set of objects for which $D_S(X)$ gives the answer of *unknown.* Finally, let $NEG_S(X)$ be a set of objects that are not regarded as members of X by $D_S(X)$.

The approximation accuracy of *a decision algorithm* $D_S(X)$ is defined as the ratio of objects in $POS_S(X)$ that are correctly approximated to be in X to all objects in $POS_S(X)$, which can be formulated as $AD = |POS_S(X) \cap X| / |POS_S(X)|$. We also introduce a second accuracy measure that is called the approximation accuracy of *a concept* X in S. It is defined as the ratio of objects in X that are correctly approximated as $POS_S(X)$ to all objects in X, which can be formulated as $AC = |POS_S(X) \cap X| / |X|$. To get the overall picture, we propose to consider the normalized size of intersection between $POS_S(X)$ and X. This intuitive idea, 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). We simply take $s_1 = s_2 = 0.5$. 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: $POS^l_S(X) = \{x \in U : [x]_{\widetilde{CON}} \subseteq X\}$.

2. The upper bound approximation: $POS^u_S(X) = \{x \in U : [x]_{\widetilde{CON}} \bigcap X \neq \emptyset\}$.

3. The elementary set approximation: $POS^e_S(X) = \bigcup_{\frac{|R_i \cap X|}{|R_i|} \geq \theta} R_i$, where θ denotes a threshold value ranging in $(0.5, 1]$ and $R_i \in U/\widetilde{CON}$.

For all approximation methods stated above, the boundary region $BND_S(X)$ is equal to $POS^u_S(X) - POS_S(X)$.

A classification problem is described as generating a decision algorithm from S, $D(S)$, that relates elements of $U/\widetilde{CON}$ to that of $U/\widetilde{DEC}$. If $D(S)$ 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 nondeterministically. Let $U/\widetilde{DEC} = \{X_1, X_2, \ldots, X_n\}$. Since $POS(S) = \{POS_S(X_1), POS_S(X_2), \ldots, POS_S(X_n)\}$, the extension of an approximation method to its counterpart in classification problem is straightforward. Similarly, the classification quality $\varphi(S)$ is equal to $\frac{1}{|U|}\sum_{i=1}^{n} |X_i|\, \mu_S(X_i)$.

Our motivation is two fold. First, from the point of database 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 the part of the data where decisions can be made with certainty. On the other hand elementary set classifier provide us a fine and consistent approximation of target concepts on the expense of more demand on disk space when the data is incomplete or noisy. Second, one of the characteristics of database mining is that data is dynamic. Neither lower nor elementary set classification methods provide basis for adaptive classifiers since they weed out some portion of background knowledge. Whereas the upper classification method assumes such a decision is a matter of how its decision algorithm is interpreted; that is, an upper classifier bears inconsistency given in a background knowledge. This feature of upper classification method enable us to develop truely adaptive classifiers. Additionally an upper classifier could be just as well interpreted if it were produced by the classification method using only lower bounds or elementary sets.

Stepwise Backward Feature Selection

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 UXQ'. We say that $CON - P$ is *θ-superfluous* in S iff $\varphi(S/P) = \varphi(S)(1-\theta)$, where $0 \leq \theta \leq 1$. Similarly, we say that P is a *θ-reduct* of CON iff $CON - P$ is *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, the feature selection problem can be re-expressed as finding a θ-reduct of CON in S.

Algorithm SBS(S, θ)

```
1.  F = CON
2.  Threshold = φ^u(S) * (1 − θ)
3.  for(j = |CON| ; j > 1; j − −)
4.      MinQuality = Threshold
5.      Found = false
6.      for (i = 0; i < j; i + +)
7.          F = F − {c_i}
8.          CurrentQuality = φ^u(S/F)
9.          if (CurrentQuality ≥ MinQuality)
10.             MinQuality = CurrentQuality
11.             KeepAttribute = {c_i}
12.             Found = true
13.         F = F ∪ {c_i}
14.     if(Found == true) F = F − KeepAttribute
15.         else break
16. return F
```

Table 1: The stepwise backward selection algorithm.

In Table 1, Stepwise Backward Selection (SBS) algorithm is defined using C language like notation. At first two steps, we initialize F with CON and find the threshold value for the quality of an upper classifier in S. In the inner loop, we find an attribute $c \in F$ such that its elimination gives the minimum decrease in the quality of the upper classifier in S/F, but the resulting value of the quality of the upper classifier in $S/(F - \{c\})$ is no worse than the given threshold for that of S. Such an attribute is eliminated at each iteration of the outer loop until either there is only one attribute is left or no other attribute can be pruned down with respect to the threshold.

Let l be the size of CON, p be the size of $U/\widetilde{CON}$, and m be the number of objects in U. The computational effort to evaluate the quality of an upper classifier in a given S is equal to $O_{\varphi^u(S)}(lmp)$ (Sever 1995). Then it is easy to see that the time complexity of SBS algorithm is bounded by $O(l^2 O_{\varphi^u(S)}(lmp))$, which is a polynomial time algorithm with the degree of three in l and of one in m and p. To justify that SBS algorithm finds a θ-reduct of CON in S, we need to prove that the quality of an upper classifier worsens as the feature set is pruned down.

Lemma 1 *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 $U/\widetilde{B}$.*

Proof: Assume $P = B \cup B'$. Let $|P| = k$ and $|B| = r$. It is enough to show that for all $[u]_{\widetilde{P}}$, there exists $[y]_{\widetilde{B}}$ such that $[u]_{\widetilde{P}} \subseteq [y]_{\widetilde{B}}$. Let $[u]_{\widetilde{P}}$ be an element of $U/\widetilde{P}$. Since $U/\widetilde{B}$ is a partition of U, there must be a $[y]_{\widetilde{B}}$ that contains u. Let $\bar{\rho}_u(P) =< v_0, v_1, \ldots, v_k >$ Then,

$$\begin{aligned}
[u]_{\widetilde{P}} &= \{x : x \in U \wedge \bar{\rho}_x(P) =< v_0, v_1, \ldots, v_k >\} \\
&= \{y : y \in U \wedge \bar{\rho}_y(B) =< v_0, v_1, \ldots, v_r >\} \cap \\
&\quad \{z : z \in U \wedge \bar{\rho}_z(B') =< v_{r+1}, v_{r+2}, \ldots, v_k >\} \\
&\leq \{y : y \in U \wedge \bar{\rho}_y(B) =< v_0, v_1, \ldots, v_r >\} \\
&= [y]_{\widetilde{B}} \ \square
\end{aligned}$$

Theorem 1 *Let $X \subseteq U$ and $\mu^u_S(X)$ be the quality of upper approximation of X in S.*

$$\forall (P, B) \subseteq CON[P \supseteq B \Longrightarrow \mu^u_{S/P}(X) \geq \mu^u_{S/B}(X)]$$

Proof: Assume $P \supseteq B$. We know that $POS^u_{S/P}(X) = \bigcup_{[x]_{\widetilde{P}} \cap X \neq \emptyset} [x]_{\widetilde{P}}$ and $POS^u_{S/B}(X) = \bigcup_{[y]_{\widetilde{B}} \cap X \neq \emptyset} [y]_{\widetilde{B}}$, by definition. It is easy to see that if $[x]_{\widetilde{P}}$ is included in $POS^u_{S/P}(X)$ then there exists a $[y]_{\widetilde{B}} \supseteq [x]_{\widetilde{P}}$ that must also be included in $POS^u_{S/B}(X)$ because $U/\widetilde{P}$ is refinement of $U/\widetilde{B}$ by Lemma 1. Then $POS^u_{S/P}(X) \subseteq POS^u_{S/B}(X)$. Thus,

$$\begin{aligned}
\mu^u_{S/P}(X) &= 2\frac{|X|}{|X| + \left|POS^u_{S/P}(X)\right|} \\
&\geq 2\frac{|X|}{|X| + \left|POS^u_{S/B}(X)\right|} \\
&= \mu^u_{S/B}(X) \ \square
\end{aligned}$$

The immediate consequence of Theorem 1 is given in the following.

Corollary 1 $\forall (P, B) \subseteq CON[P \supseteq B \Longrightarrow \varphi^u(S/P) \geq \varphi^u(S/B)] \ \square$

According to corollary 1, pruning the feature set down does not yield a better classification quality. This fact guarantees that SBS algorithm satisfies 'sufficient' condition of the feature selection problem. It is also easy to see that there is no θ-superfluous attribute in returned set of relevant attributes, F, because the last repetition of outer loop (i.e., steps 3-15) in SBS algorithm verifies that no $c \in F$ can be extracted from F and yet we have $\varphi^u(S/(F - \{c\})) \geq \varphi^u(S/F) * (1 - \theta)$.

Corollary 1 also enables us to use a *branch and bound* algorithm to find the smallest θ-reduct of relevant features. Since the detailed description of the branch and bound algorithm for feature selection is given in Narendra & Fukunaga 1977, we only state the corresponding information to incorporate our feature selection criterion into that algorithm. For a given decision table S and threshold value θ, a subset of condition attributes, denoted by F, and the quality of an upper classifier in S/F constitute the state of a node in a search space. The state of a root node is, then, defined by CON and $\varphi^u(S)$. A subset F is said to be *feasible* if the feature selection criterion, defined as $\varphi^u(S/F) \geq \varphi^u(S) * (1 - \theta)$, is satisfied. Note that the objective function is defined as optimizing the feature selection criterion (i.e., finding a feasible subset such that its size is minimum among the other feasible subsets).

Empirical Evaluation

To see how SBS algorithm affects the performance of an upper classifier, we design and run an experiment that uses traditional machine learning data sets. In the next two subsections, we discuss the issues related to designing the experiment and interpret the experimental results.

Design of the Experiment

The data sets are from the UC Irvine's machine learning database repository (Murphy & Aha), except for parity 5+10 and XOR, which 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 'exclusive or' with thirteen irrelevant bits. For all data sets that do not have a test set, we randomly choose two third of the objects from each class of the corresponding data set for a test set. We induce two upper classifiers for each data set; one is from a non-reduced training set and the other one is from a reduced training set. 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. No domain knowledge on data sets is exploited, except the type of attributes, e.g., quantitative or non-quantitative. Given a test set, the accuracy of an upper classifier is defined as the ratio of the number of correctly classified objects to the number of objects in the test set. 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 closest known concept. The difference between two values of an attribute are

data set	*Size*		
	Attr.	*Training*	*Test*
1. Glass	9	66	148
2. Breast cancer	9	211	488
3. Parity 5+10	15	226	524
4. Iris	4	45	105
5. Monk 1	6	124	432
6. Monk 2	6	169	432
7. Monk 3	6	122	432
8. Vote	16	132	303
9. Soybean-small	35	15	32
10. XOR	15	226	524
11. Mushroom	22	2439	5685
12. Hepatitis	19	47	108

Table 2: Part I of Table 3.

data	*Accuracy*		*Subset*	*Reduc-*
set	*UC*	*SBS+UC*		tion
1.	64.3%	78.4%	0,1	77.8%
2.	96.4%	91.8%	0,2	77.8%
3.	55.5%	100.0%	0,1,2,3,4	66.7%
4.	92.4%	94.3%	2	75.0%
5.	86.1%	100.0%	0,1,4	50.0%
6.	74.8%	67.8%	1,2,3,4,5	16.7%
7.	90.0%	93.5%	0,1,3,4	33.3%
8.	91.1%	95.4%	1,2,3,6,7	68.8%
9.	100%	65.3%	0,2,3	91.43%
10.	78.1%	100.0%	3,8	86.7%
11.	99.5%	99.7%	2,4,9,10,13	72.27%
12	75.9%	77.8%	0,2,4	84.21%

Table 3: Comparison of classification accuracies of UC and UC+SBS on real data sets.

computed as suggested in *Relief* algorithm (Kira & Rendell 1992); that is, the difference between two non-quantitative values is one if they are different and zero otherwise, and the difference between two quantitative values is normalized into the interval [0,1].

The Results of the Experiment

Table 3 shows some data sets with their number of attributes, training and test sizes,

the accuracies of the Upper Classifier (UC) without and with the feature selection algorithm, SBS, the subset of features that SBS algorithm have returned when applied to the corresponding training sets, and the percentage of reductions in feature sets.

On parity and XOR data sets, SBS algorithm found the smallest reduct of attributes. Since reduced training sets of these data sets were complete, in the sense that they contained all combinations of corresponding relevant attributes, $SBS + UC$ performed much better than UC did. The only data set that $SBS + UC$ relatively performed much worse than UC was small soybean data set. When we continued our experiment on this data set with different θ-reduct that was 20th and 21st features of soybean's training set, we obtained accuracy of 96.1%, which was much better than that of the one given in soybean row in Table 2. Hence, soybean data set is peculiar enough to show us that a θ-reduct of a feature set would not be as good as another one. On all data sets, average accuracy and sample standard deviation of UC and $SBS + UC$ are $83.40 \pm 13.83\%$ and $88.66 \pm 12.87\%$, respectively. On the other hand the average percentage of the reduction in features of data sets, excluding parity and XOR, is 64.7%. These results indicate that SBS algorithm finds a small subset of features that are sufficient to define target (or unknown) concepts.

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*[1], 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 a decision rule. We use EDT to represent a decision algorithm that is induced such that the antecedent part of each rule corresponds only one elementary set in the base decision table. Details of EDT are omitted for lack of space (Deogun, Raghavan, & Sever 1994; Sever 1995). The important thing that we would like to point out in the following paragraph is that EDT has three important advantages.

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 deci-

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

sion table is easily propagated to it. Observe that the feedback channel we consider here is between *EDT* and base decision table. Third, *EDT* enable us to interpret inconsistent decision algorithms either deterministically or nondeterministically as explained below.

A deterministic interpretation of an inconsistent EDT would be *to always select the row whose accuracy measure is the maximum among the conflicting 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 domain conflict 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.

Conclusion

The contributions of our work can be summarized as follows. We introduced a feature selection algorithm that finds a θ-reduct of given feature set in polynomial time. This algorithm can be used in places where the quality of classification is monotonically non-increasing function as the feature set is reduced. We showed that the upper classifier can be summarized at a desired level of abstraction. The incorporation of incremental information into extended decision tables can make decision algorithms capable of evolving over time. This also allows an inconsistent decision algorithm to behave as if it were a consistent decision algorithm.

References

Bollmann, P., and Cherniavsky, V. S. 1981. Measurement-theoretical investigation of the mz-metric. In Oddy, R. N.; Robertson, S. E.; van Rusbergen, C. J.; and Williams, R. W., eds., *Information Retrieval Research.* 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.* SMC-3:166–171.

Deogun, J. S.; Raghavan, V. V.; and Sever, H. 1994. Rough set based classification methods and extended decision tables. In *Proceedings of the International Workshop on Rough Sets and Soft Computing*, 302–309.

Devicver, R. A., and Kittler, J. 1982. *Pattern Recognation: A statistical approach.* London: Prentice Hall.

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

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

James, M. 1985. *Classification Algorithms.* John Wiley & Sons.

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

Matheus, C. J.; Chan, P. K.; and Piatetsky-Shapiro, G. 1993. Systems for knowledge discovery in databases. *IEEE Trans. on Knowledge and Data Engineering* 5(6):903–912.

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

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

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

Narendra, P. M., and Fukunaga, K. 1977. A branch and bound algorithm for feature subset selection. *IEEE Trans. on Computers* c-26(9):917–922.

Pal, S. K., and Chakraborty, B. 1986. Fuzzy set theoretic measure for automatic feature evaluation. *IEEE Trans. Syst., Man, Cybern.* SMC-16(5):754–760.

Pawlak, Z. 1986. On learning- a rough set approach. In *Lecture Notes*, volume 208. Springer Verlag. 197-227.

Raghavan, V. V., and Sever, H. 1995. The state of rough sets for database mining applications. In Lin, T. Y., ed., *Proceedings of 23rd Computer Science Conference Workshop on Rough Sets and Database Mining*, 1–11.

Salzberg, S. 1992. Improving classification methods via feature selection. Technical Report JHU-92/12, Johns Hopkins University, Department of Computer Science. (revised April 1993).

Sever, H. 1995. *Knowledge Structuring for Database Mining and Text Retrieval Using Past Optimal Queries.* Ph.D. Dissertation, The University of Southwestern Louisiana.

Slowinski, R., and Stefanowiski, J. 1995. Rough classification with valued closeness relation. In *Proceedings of the International Workshop on Rough Sets and Knowledge Discovery.*

Ziarko, W. 1991. The discovery, analysis, and representation of data dependencies in databases. In Piatetsky-Shapiro, G., and Frawley, W. J., eds., *Knowledge Discovery in Databases.* Cambridge, MA: AAAI/MIT.