

A comparison of feature selection algorithms in the context of rough classifiers

Suresh K. Choubey* Jitender S. Deogun† Vijay V. Raghavan‡ Hayri Sever§

Abstract

In this paper, we study the feature selection problem and 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 from a given feature set and progressively remove features, while controlling the amount of degradation in classification quality, but differ in the heuristic used for pruning the search space of features. Our experimental results confirm the analytical results on the complexity of algorithms as well as on controlled degradation of upper classification. The algorithms presented can be used with any methods of deriving a classifier where the quality of classification is monotonically decreasing function while feature set is reduced, though we have adopted the *upper classifier* in our study. The upper classifier has some important features that makes it suitable for database mining applications. In particular, we have shown that the 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.

Keywords- Rough sets, feature selection, upper classifier, database mining.

1 Introduction

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 learn-

ing and classifying objects, reducing the cost of classification (e.g., eliminating redundant tests in medical diagnosis), and improving the quality of classification [1]. 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 [2, 3], and dynamic programming [4] 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 [5, 6]; entropy or classification accuracy in pattern recognition and machine learning [?, 7, 8]; classification quality based on variations of MZ metric in information retrieval systems [9]. In such procedures, irrelevant features are either eliminated or assigned small coefficients.

We present four feature selection algorithms of polynomial time complexity. The objective is to find a small subset of features that are 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 classification. Controlled threshold value is inspired by Salzberg's CSS algorithm [10]. Even though our feature selection algorithms are developed as a pre-processing stage for rough classifiers, they 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, pharmacology, industry, engineering, control systems, social sciences, switching circuits, image processing, etc., [11, 12]. In this article we consider classification meth-

*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

‡CONTACT PERSON raghavan@cacs.usl.edu, The Center for Advanced Computer Studies, University of SW Louisiana, Lafayette, LA 70504, USA

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

ods 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 aspects of very large data that can be listed as redundant, incomplete, noisy, and dynamic data [13]. In the rough set literature, the terms ‘inconsistent’ and ‘nondeterministic’ decision algorithms (or rules) are used interchangeably [14, 15], though they are different concepts. As shown in Deogun et al. [16], inconsistent decision algorithms, under an appropriate representation structure, can be interpreted deterministically as well as nondeterministically. This is an important result, particularly when the background knowledge is *incomplete and dynamic*.

We have proposed in Deogun et al. [17], 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 algorithms find 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. In [17], only one heuristic was reported. In this paper, we present a comprehensive experimental study of four different algorithms.

2 Classification Methods in Rough Set Theory

A decision table is defined as a quadruple $S = (U, Q = CON \cup DEC, V, \rho)$ where: U is the finite set of objects; Q is the union of condition, denoted by CON , and decision attributes, denoted by DEC ; 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 $a \in Q$, $\rho(x, a) = \rho_x(a)$. For given $P \subseteq Q$, let $|P| = m$. For given $P \subseteq Q$, let $|P| = m$. Let us introduce following notations. $\bar{\rho}_x(P)$ denotes extended version of total description function, that is, $\bar{\rho}_x(P) = \langle v_1, v_2, \dots, v_m \rangle$, where $v_i = \rho_x(Pi)$. \tilde{P} denotes the equivalence relation on U defined by the values of P , that is, $\tilde{P} = \{(x, y) : x, y \in U \wedge \bar{\rho}_x(P) = \bar{\rho}_y(P)\}$. U/\tilde{P} denotes the set of blocks of \tilde{P} on U . $[x]_{\tilde{P}} = \{y : \bar{\rho}_x(P) = \bar{\rho}_y(P)\}$ denotes a block of

U/\tilde{P} .

Let $X \subseteq U$ be a concept of interest. A decision algorithm, denoted by $D_A(X)$, is induced from S such that, for a given object x , it yields one of these three answers: a) x is in X , b) x is not in X , c) *unknown*. Let $POS_S(X)$ be the set of objects that is considered to be a part of the concept X by the decision algorithm $D_S(X)$. 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 a set of objects that are not regarded as members of X by $D_S(X)$.

The approximation accuracy of a *decision algorithm* $D_S(X)$, 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 accuracy measure that is called the approximation accuracy 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, 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: $POS_S^l(X) = \{x \in U : [x]_{\tilde{P}} \subseteq X\}$.
2. The upper bound approximation: $POS_S^u(X) = \{x \in U : [x]_{\tilde{P}} \cap X \neq \emptyset\}$.
3. The elementary set approximation: $POS_S^e(X) = \bigcup_{\frac{|R_i \cap X|}{|R_i|} \geq \theta} R_i$, 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 $POS_S^u(X) - POS_S^l(X)$.

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*. Let $U/\widetilde{DEC} = \{X_1, X_2, \dots, X_n\}$. 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 $\frac{1}{|U|} \sum_{i=1}^n |X_i| \mu_S(X_i)$.

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 certain data. On the other hand elementary set classifier provides 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 database 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. In contrast, the upper classification method assumes 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 a background knowledge. Thus, an inconsistent decision algorithm might be interpreted deterministically or nondeterministically. This feature of upper classification method enable us to develop a truly adaptive classifiers. Additionally an upper classifier, a decision algorithms produced by the upper classification method, could be just as well interpreted as if it were produced by the classification method using only lower bounds or elementary sets¹. We will elaborate this point in the Section 6.

3 Statement of the 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

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

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 .

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)*, 4.) *K-level Best SBS (KBS)*. 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 $\varphi_{S/F}^u(U/\widetilde{DEC})$. State of the root node is, defined by CON and $\varphi_S^u(U/\widetilde{DEC})$. Note that the superscript u is used to indicate that $POS_S(X_i)$ is defined as $POS_S^u(X_i)$, for each $X_i \in U/\widetilde{DEC}$. The feasibility condition for a current node to be expanded is imposed as

$$\varphi_{S/F}^u(U/\widetilde{DEC}) \geq \varphi_S^u(U/\widetilde{DEC}) * (1 - \theta).$$

In *BFS* 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 [17, 18]. In *BFS* algorithm, we initialize the current node to the root node in the beginning. Here the state of root node consists of the given set of features and 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. Let F be the computational effort to compute theta-superfluous in given S , then, *BFS* was shown to be of order $O(l^2)$, complexity [18], where $l = |CON|$.

For the *HHS* algorithm, in the beginning, 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.

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.

KBS algorithm divides the search space from root node level (or level-1) to leaf node level (or level- $|CON|$) into $\lceil \frac{K}{|CON|} \rceil$ 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.

5 Experimental Results

A set of experiments that use traditional machine learning data sets 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) [19], 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 parity ‘exclusive or’ with thirteen irrelevant bits. 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 1 shows the data sets used with their number of attributes, training and test set sizes. Table 2 shows the percentage of accuracy for different data sets and different methods using *different* data sets for test and training and *same* data sets for test and training. The test sets used in the first group of experiments are used both as training and test sets in the second group. We call the first group of tests as Predictive Experiments and the second group of

tests as Upperbound Experiments. Upperbound Experiments are seen as providing upper bound on the accuracy achievable by the Predictive Experiments. This table gives the accuracies of the upper classifier without and with the feature selection algorithms. Table 3 and Table 4 present the feature subsets and percentage of reduction in feature sets for different data sets using different feature selection algorithms. Table 3 refers to the results of Predictive Experiments whereas Table 4 refers to the results of Upperbound Experiments. Table 5 shows the number of nodes visited for different data sets using different methods. Table 5 has the result for both groups of experiments. Tables 2 through 5 appear in the last page, due to their size. 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. When the description of a given object does not match to known concepts we use 5NNR classification scheme with Euclidean distance function to determine closest known concept. The difference between two values of an attribute are computed as suggested in *Relief* algorithm [1]; 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 $[0,1]$.

We first consider results from Table 2. 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

<i>data set</i>	<i>Size</i>		<i>No. of Attributes</i>
	<i>Training set</i>	<i>Test set</i>	
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

Table 1: Test and Training sets for different various data sets.

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, monks1, and monks2 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 Tables 3 and 4, 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 $HHS + UC$ 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 5 gives 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.

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*.

To incorporate incremental information into the decision table, we have introduced 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 they can be found in[16]. The important thing we would like to point out 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. Third, EDT enable us to interpret inconsistent decision algorithms either deterministically or nondeterministically. 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.

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. Based on our experiments, we are confident that we can find θ -reduct with very little or almost no degradation in classification quality. In one group of experiments, the same data is used to both train and test a classification system, in order to determine an upper bound on the performance of that system. It has been shown that

results of predictive experiments 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 monotonically decreasing function while feature set is reduced. 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

- [1] K. Kira and L. Rendell, "The feature selection problem: Traditional methods and a new algorithm," in *Proceedings of AAAI-92*, pp. 129–134, AAAI Press, 1992.
- [2] M. James, *Classification Algorithms*. John Wiley & Sons, 1985.
- [3] M. Modrzejewski, "Feature selection using rough sets theory," in *Machine Learning: Proceedings of ECML-93* (P. B. Brazdil, ed.), pp. 213–226, Springer-Verlag, 1993.
- [4] C. Y. Chang, "Dynamic programming as applied to feature subset selection in a pattern recognition system," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, pp. 166–171, 1973.
- [5] R. A. Devijver and J. Kittler, *Pattern Recognition: A statistical approach*. London: Prentice Hall, 1982.
- [6] A. J. Miller, *Subset Selection in Regression*. Chapman and Hall, 1990.
- [7] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.
- [8] U. M. Fayyad and K. B. Irani, "The attribute selection problem in decision tree generation," in *Proceedings of AAAI-92*, pp. 104–110, AAAI Press, 1992.
- [9] P. Bollmann and V. S. Cherniavsky, "Measurement-theoretical investigation of the mz-metric," in *Information Retrieval Research* (R. N. Oddy, S. E. Robertson, C. J. van Rusbergen, and R. W. Williams, eds.), pp. 256–267, Boston: Butterworths, 1981.
- [10] S. Salzberg, "Improving classification methods via feature selection," Tech. Rep. JHU-92/12, Johns Hopkins University, Department of Computer Science, Jun 1992. (revised April 1993).
- [11] R. Slowinski, ed., *Intelligent Decision Support-Handbook of Advances and Applications of the Rough Set Theory*. Boston, MA: Kluwer Academic Publishers, 1992.
- [12] V. V. Raghavan and H. Sever, "The state of rough sets for database mining applications," in *Proceedings of 23rd Computer Science Conference Workshop on Rough Sets and Database Mining* (T. Y. Lin, ed.), (San Jose State University, San Jose, CA), pp. 1–11, mar 1995.
- [13] C. J. Matheus, P. K. Chan, and G. Piatetsky-Shapiro, "Systems for knowledge discovery in databases," *IEEE Trans. on Knowledge and Data Engineering*, vol. 5, no. 6, pp. 903–912, 1993.
- [14] R. Slowinski and J. Stefanowski, "Rough classification with valued closeness relation," in *Proceedings of the International Workshop on Rough Sets and Knowledge Discovery*, (San Jose, CA), 1995.
- [15] Z. Pawlak, "On learning- a rough set approach," in *Lecture Notes*, vol. 208, pp. 197–227, Springer Verlag, 1986.
- [16] J. S. Deogun, V. V. Raghavan, and H. Sever, "Rough set based classification methods and extended decision tables," in *Proceedings of the International Workshop on Rough Sets and Soft Computing*, (San Jose, California), pp. 302–309, 1994.
- [17] J. S. Deogun, V. V. Raghavan, and H. Sever, "Exploiting upper approximations in the rough set methodology," in *The First International Conference on Knowledge Discovery and Data Mining* (U. Fayyad and R. Uthurusamy, eds.), (Montreal, Quebec, Canada), pp. 69–74, aug 1995.
- [18] H. Sever, *Knowledge Structuring for Database Mining and Text Retrieval Using Past Optimal Queries*. PhD thesis, The University of Southwestern Louisiana, 1995.
- [19] P. M. Murphy and D. W. Aha, "UCI repository of machine learning databases." For information contact m-lrepository@ics.uci.edu.

<i>data set</i>	%Accuracy for Predictive Experiments					%Accuracy for Upperbound Experiments				
	<i>UC</i>	<i>UC+BFS</i>	<i>UC+AHS</i>	<i>UC+HHS</i>	<i>UC+KBS</i>	<i>UC</i>	<i>UC+BFS</i>	<i>UC+AHS</i>	<i>UC+HHS</i>	<i>UC+KBS</i>
Glass	64.3	36.5	53.4	50	36.5	100	100	100	100	100
Breast	96.4	91.8	93.1	95	91.8	100	99.6	99.6	99.6	99.6
Parity	55.5	100	52.9	55.9	100	100	100	98.7	98.7	100
Iris	92.4	94.3	94.3	94.3	94.3	95.2	95.2	95.2	95.2	95.2
Monk 1	86.1	100	70.4	70.4	100	100	83.3	75	75	83.3
Monk 2	74.8	67.8	67.8	67.8	67.8	100	84.7	78.9	78.9	84.7
Monk 3	90	93.5	93.8	93.8	93.5	100	100	100	100	100
Vote	91.1	95.4	95.7	93.4	95.4	100	99.3	99.3	99.3	99.3
Soybean-s	96.9	61.1	100	100	100	100	100	100	100	100
XOR	78.1	100	100	100	100	100	100	100	100	100
Mushroom	99.5	99.7	99.6	99.8	99.7	100	100	100	100	100
Hepatitis	75.9	77.8	81.5	78.7	77.8	100	100	100	100	100

Table 2: Comparison of classification accuracies of UC and other heuristics on various data sets.

<i>data set</i>	<i>UC+BFS</i>		<i>UC+AHS</i>		<i>UC+HHS</i>		<i>UC+KBS</i>	
	Attr. Subset	%Red.	Attr. Subset	%Red.	Attr. Subset	%Red.	Attr. 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	2,3,5,6,7,9-14	27	0,1,2,3,4	67
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	2,4,9,10,13	77	4,9,10,14	82	8-10,15,19-21	68	2,4,9,10,13	77
Hepatitis	0,2,4	84	8,10,13	84	15,17,18	84	0,2,4	84

Table 3: Comparison of reduction in attribute sets for heuristics for Predictive Experiments.

<i>data set</i>	<i>UC+BFS</i>		<i>UC+AHS</i>		<i>UC+HHS</i>		<i>UC+KBS</i>	
	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	2,3,6,7,9,11,14,15	50	2,3,10-12,14,15	56	0-3,9-11	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

Table 4: Comparison of reduction in attribute sets for heuristics for Upperbound Experiments.

<i>data set</i>	<i>No. of nodes visited for Predictive Experiments</i>				<i>No. of nodes visited for Upperbound Experiments</i>			
	<i>UC+BFS</i>	<i>UC+AHS</i>	<i>UC+HHS</i>	<i>UC+KBS</i>	<i>UC+BFS</i>	<i>UC+AHS</i>	<i>UC+HHS</i>	<i>UC+KBS</i>
Glass	45	26	21	25	44	26	21	26
Breast cancer	44	30	25	26	39	33	25	25
Parity 5+10	110	149	198	60	110	184	212	60
Iris	10	6	7	6	9	8	8	6
Monk 1	18	26	31	13	20	12	9	12
Monk 2	11	26	31	11	11	26	31	11
Monk 3	15	22	27	10	18	17	28	13
Vote	126	89	45	71	115	120	117	67
Soybean-s	624	325	62	324	627	325	62	323
XOR	119	69	39	65	119	69	16	65
Mushroom	243	149	123	131				
Hepatitis	187	105	32	99	169	116	47	91

Table 5: Comparison of number of nodes visited by the heuristics on various data sets.