To appear in the Third International Workshop on
Rough Sets and Soft Computing (RSSC 94)

Useful Feature Subsets and Rough Set Reducts

Ron Kohavi and **Brian Frasca**
Computer Science Dept.
Stanford University
Stanford, CA 94305
{ronnyk,brianf}@CS.Stanford.EDU

Abstract

In supervised classification learning, one attempts to induce a classifier that correctly predicts the label of novel instances. We demonstrate that by choosing a useful subset of features for the indiscernibility relation, an induction algorithm based on simple decision table can have high prediction accuracy on artificial and real-world datasets. We show that useful feature subsets are not necessarily maximal independent sets (relative reducts) with respect to the label, and that, in practical situations, using a subset of the relative core features may lead to superior performance.

1 Introduction

In supervised classification learning, one is given a training set containing labelled instances (examples). Each labelled instance contains a list of feature values (attribute values) and a discrete label value. The induction task is to build a classifier that will correctly predict the label of novel instances. Common classifiers are decision trees, neural networks, and nearest-neighbor methods.

Many induction algorithms do not scale up well in the face of irrelevant features. Aha (1992) reports that "IB3's storage requirement increases exponentially with the number of irrelevant attributes." John, Kohavi, and Pfleger (1994) show that C4.5 (Quinlan 1993), a state-of-the-art decision tree induction algorithm, drastically degrades in performance when an irrelevant feature and a relevant but noisy feature are added to a given dataset.

A rough set (Pawlak 1991) approximates a given concept from below and from above using an indiscernibility relation. Pawlak (1993) points out that one of the most important and fundamental notions to the rough sets philosophy is the need to discover redundancy and dependencies between features.

A subset of features is *useful* with respect to a given dataset and an induction algorithm if the projection of the dataset on the subset of features leads to accurate predictions by the induced classifier. In this paper, we show that by searching the space of feature subsets, we are able to select a useful subset for a decision table classifier (Pawlak 1987; Hurley 1983), which yields high prediction accuracy on unseen instances.

To find a useful feature subset, we conduct a best-first search in the space of feature subsets, estimating the accuracy of the induced classifier at each state using the Bootstrap estimation method (Efron & Tibshirani 1993).

Rough set theory defines the unique *core* of a dataset to be the set of *indispensable* features (see Section 2). Removal of any feature from the core set changes the positive region with respect to the label. Our experiments show that it is sometimes beneficial to use a subset of features that does not necessarily contain all the features in the core, and hence a subset that is not a *reduct*. This observation deserves careful attention, as there has been a large effort in the rough set community to efficiently find the set of reducts (Skowron & Rauszer 1991; Skowron & Ruaszer 1992; Grzymala-Busse 1992).

The paper is organized as follows. Section 2 briefly introduces the relevant rough set terminology. Section 3 introduces the basic table classifier and the table-majority classifiers, and shows their performance on some datasets. Section 4 introduces the `Holte-II` inducer which is based on a table-majority classifier but searches for a good subset of features. Results are presented in Section 5 with a discussion of their relevance to rough sets in Section 6. Section 7 concludes with a summary and future work.

2 Rough Set Theory

This section describes the basic concepts in rough set theory, viewed from a supervised classification learning perspective. An *information system* is a four-tuple $S = (\mathcal{U}, \mathcal{Q}, \mathcal{V}, f)$, where

$\mathcal{U}$ is the finite universe of objects.

$\mathcal{Q}$ is the finite set of features, or attributes.

$\mathcal{V}$ is the set of possible feature values.

f is the information function. Given an object and a feature, f maps it to a value:

$$f : \mathcal{U} \times \mathcal{Q} \mapsto \mathcal{V}$$

Let $\mathcal{P}$ be a subset of $\mathcal{Q}$, that is, $\mathcal{P}$ is a subset of features. The *indiscernibility relation,* denoted by $\text{IND}(\mathcal{P})$, is an equivalence relation defined as

$$\text{IND}(\mathcal{P}) = \left\{ \begin{array}{r} \langle x, y\rangle \in \mathcal{U} \times \mathcal{U} : f(x, a) = f(y, a) \\ \text{for every feature } a \in \mathcal{P}. \end{array} \right\}$$

If $\langle x, y\rangle \in \text{IND}(\mathcal{P})$, then x and y are *indiscernible* with respect to the subset $\mathcal{P}$. $\mathcal{U}/\text{IND}(\mathcal{P})$ denotes the set of equivalence classes (the partition) of $\text{IND}(\mathcal{P})$. Each element in $\mathcal{U}/\text{IND}(\mathcal{P})$ is thus a set of objects that are indiscernible with respect to $\mathcal{P}$.

For any concept $\mathcal{X} \subseteq \mathcal{U}$ and for any subset of features $\mathcal{P}$, the lower approximation, $\underline{\mathcal{P}}$, and the upper approximation $\overline{\mathcal{P}}$ are defined as follows:

$$\begin{aligned} \underline{\mathcal{P}}(\mathcal{X}) &= \cup\{\mathcal{Y} \in \mathcal{U}/\text{IND}(\mathcal{P}) : \mathcal{Y} \subseteq \mathcal{X}\} \\ \overline{\mathcal{P}}(\mathcal{X}) &= \cup\{\mathcal{Y} \in \mathcal{U}/\text{IND}(\mathcal{P}) : \mathcal{Y} \cap \mathcal{X} \neq \emptyset\} \end{aligned}$$

The boundary region for $\mathcal{P}$ and concept $\mathcal{X}$ is defined as follows:

$$\text{BND}_{\mathcal{P}}(\mathcal{X}) = \overline{\mathcal{P}} - \underline{\mathcal{P}}$$

If $\text{BND}_{\mathcal{P}}(\mathcal{X}) = \emptyset$ then $\mathcal{X}$ is *definable* using $\mathcal{P}$; otherwise $\mathcal{X}$ is a *rough set* with respect to $\mathcal{P}$. Figure 1 depicts a rough set and the terms defined above.

The *degree of dependency* of a set of features $\mathcal{P}$ on a set of features $\mathcal{R}$ is denoted by $\gamma_{\mathcal{R}}(\mathcal{P})$ $(0 \leq \gamma_{\mathcal{R}}(\mathcal{P}) \leq 1)$ and is defined as

$$\gamma_{\mathcal{R}}(\mathcal{P}) = \frac{|\text{POS}_{\mathcal{R}}(\mathcal{P})|}{|\mathcal{U}|} \quad \text{, where}$$

$$\text{POS}_{\mathcal{R}}(\mathcal{P}) = \bigcup_{\mathcal{X} \in \mathcal{U}/\text{IND}(\mathcal{P})} \underline{\mathcal{R}}\mathcal{X}$$

$\text{POS}_{\mathcal{R}}(\mathcal{P})$ contains the objects of $\mathcal{U}$ which can be classified as belonging to one of the equivalence classes of $\text{IND}(\mathcal{P})$, using only features from the set $\mathcal{R}$. If $\gamma_{\mathcal{R}}(\mathcal{P}) = 1$, then $\mathcal{R}$ functionally determines $\mathcal{P}$.

$\mathcal{P}$ is an *independent set of features* if there does not exist a strict subset $\mathcal{P}'$ of $\mathcal{P}$ such that $\text{IND}(\mathcal{P}) = \text{IND}(\mathcal{P}')$. A set $\mathcal{R} \subseteq \mathcal{P}$ is a *reduct* of $\mathcal{P}$ if it is independent and $\text{IND}(\mathcal{R}) = \text{IND}(\mathcal{P})$. Each reduct has the property that a feature can not be removed from it without changing the indiscernibility relation. Many reducts for a given set of features $\mathcal{P}$ may exist.

The set of features belonging to the intersection of all reducts of $\mathcal{P}$ is called the *core* of $\mathcal{P}$:

$$\text{core}(\mathcal{P}) = \bigcap_{\mathcal{R} \in \text{Reduct}(\mathcal{P})} \mathcal{R}$$

A feature $a \in \mathcal{P}$ is *indispensable* if $\text{IND}(\mathcal{P}) \neq \text{IND}(\mathcal{P} \setminus \{a\})$. The core of $\mathcal{P}$ is the union of all the indispensable features in $\mathcal{P}$.

The indispensable features, reducts, and core can be similarly defined relative to the output feature, or label; the same facts hold in this case. For example, the relative core is the union of all indispensable features with respect to the label, and it is the intersection of all relative reducts. The reader is referred to Pawlak's book on rough sets (Pawlak 1991, Chapter 3) for the precise definitions. Throughout this paper, the terms *core* and *reduct* will refer to the relative core and relative reduct with respect to the label.

3 Classifiers and Inducers

A *classifier* maps an unlabelled instance to a class label using some internally stored structure. Given a test set, we define the (estimated) accuracy of the classifier to be the ratio of the number of correctly classified instances to the number of instances. An *inducer* generates a classifier from a training set. The (estimated) *accuracy* of an inducer given a training set and a test set is the accuracy of the classifier induced from the training set, when run on the test set.

A *constant* classifier is probably the simplest classifier possible. Its internal structure is one class label, which it predicts independent of the input instance. A *majority* inducer computes the most frequent class in the training set, and generates the appropriate constant classifier. The accuracy of a majority inducer is usually called the *baseline accuracy* of the dataset.

A *table* classifier has a table of labelled instances as its internal structure. Given an instance, it searches for all matching instances in the table. If no matching instances are found, *unknown* is returned; otherwise, the majority class of the matching instances is returned (there may be multiple matching instances with conflicting labels). Unknown values are assumed to be a possible feature value. A *table* inducer simply passes the training set to the table classifier for its internal structure.

A *table-majority* classifier is similar to a table classifier, except that when an instance is not found in the table, the majority class of the table is returned. A *table-majority* inducer simply passes the training set to the table-majority classifier for its internal structure.

Table 1 shows some datasets with their corresponding training set and test set sizes (a single test set was used for each dataset), and the accuracy estimates for the inducers described above. The datasets are from the UC Irvine repository (Murphy & Aha 1994), except for parity5+5 which is an artificial dataset where the concept is the parity

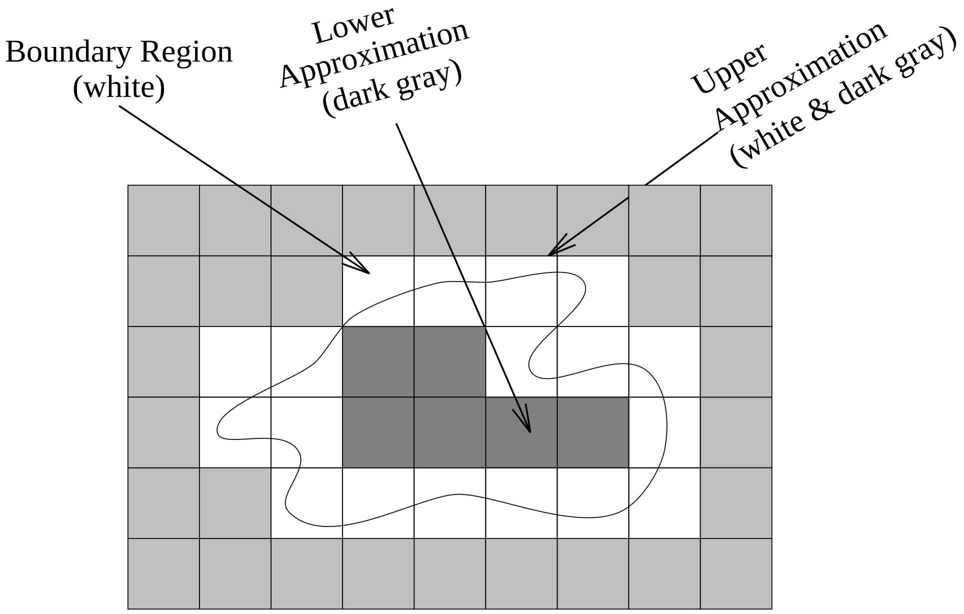

Figure 1: Depiction of a rough set

Dataset	Number of attributes	Training set size	Test set size	Majority Accuracy	Table Accuracy	Table-Maj Accuracy
Monk 1	6	124	432	50.0%	28.7%	64.4%
Monk 2	6	169	432	67.1%	39.1%	81.9%
Monk 2-local	17	169	432	67.1%	39.1%	81.9%
Monk 3	6	122	432	47.2%	26.9%	60.6%
Parity 5+5	10	100	1024	50.0%	9.8%	54.4%
Tic-Tac Toe	9	638	958	65.3%	66.6%	87.8%
Credit	15	490	200	55.0%	0.0%	55.0%
Breast cancer	9	191	95	73.7%	6.3%	74.7%
Chess	36	2130	1066	53.2%	0.0%	53.2%
Glass	9	142	72	30.6%	0.0%	30.6%
Glass2	9	108	55	58.2%	0.0%	58.2%
Heart-disease	13	202	101	62.4%	0.0%	62.4%
Hepatitis	19	103	52	86.5%	0.0%	86.5%
Horse-colic	22	300	68	60.3%	2.9%	60.3%
Hypothyroid	25	2108	1055	94.8%	2.7%	95.2%
Iris	4	100	50	30.0%	0.0%	30.0%
Labor	16	40	17	64.7%	0.0%	64.7%
Lymphography	18	98	50	60.0%	0.0%	60.0%
Mushroom	22	5416	2708	51.1%	0.0%	51.1%
Sick-euthyroid	25	2108	1055	90.4%	4.1%	90.9%
Soybean	35	31	16	31.1%	0.0%	31.1%
Vote	16	290	145	64.8%	30.3%	80.0%
vote1	15	290	145	64.8%	30.3%	80.0%

Table 1: Datasets and accuracies for the three basic inducers

of five bits, with five irrelevant bits. Monk 2-local is a variant of Monk 2 (Thrun etal. 1991) where each feature value is made into an indicator variable. Vote1 has the "physician-fee-freeze" feature deleted, something that is commonly done to make the problem harder. For all datasets that did not have a test set, we randomly chose one third of the instances for a test set.

While the performance of the table-majority inducer is considerably above the baseline (majority) in some cases, it is clear that the table is not helping much, especially when there are many features or continuous features (*e.g.*, chess, credit, labor, lymphography, soybean).

4 `Holte-II`: Table-Majority with Feature Subset Selection

Finding a good subset of features for an inducer is a hard problem. Some authors in the rough set community have suggested using the degree of dependency on the label (γ) for this selection process (Ziarko 1991; Modrzejewski 1993), or other measures such as normalized entropy (Pawlak, Wong, & Ziarko 1988). In statistics, many measures have been investigated (Boyce, Farhi, & Weischedel 1974; Miller 1990; Neter, Wasserman, & Kutner 1990), and others have been investigated in the pattern recognition community (Devijver & Kittler 1982; Ben-Bassat 1982). The main problem with such measures is that they ignore the utility of the features for a given inducer. An alternative method uses a wrapper embedding the induction algorithm (John, Kohavi, & Pfleger 1994). In the wrapper model, a search for a good feature subset is conducted using the inducer itself as a black box; the future prediction accuracy is estimated using methods such as cross validation (Breiman *et al.* 1984; Weiss & Kulikowski 1991) or Bootstrap (Efron & Tibshirani 1993).

The `Holte-II` inducer[1] is an inducer that produces a table-majority classifier based on a subset of the original set of features. Given an instance, the classifier behaves like a table-majority classifier, except that only the subset of features is used for matching.

Given a dataset with m features, there are 2^m possible subsets of features. We face two problems when trying to find a good subset. The first is how to estimate the accuracy of each subset, and the second is which subsets to examine. For our experimental results, we evaluated each subset by using Efron's .632-bootstrap estimator (Efron & Tibshirani 1993). In order to avoid searching the full space, we conducted a best-first search (Ginsberg 1993), stopping after a predetermined number of non-improving node expansions. Figure 2 shows the search through the feature subsets in the IRIS dataset. The number in brackets denotes the order the nodes are visited. The bootstrap estimate is given with one standard deviation of the accuracy after the $+/-$ sign. The estimated real accuracy (on the unseen test set) is also noted, but not used during the search, nor at any time during the induction process.

The .632-Bootstrap method creates b bootstrap samples of n instances each, where b is an external parameter and n is the number of instances in the original training set. The instances in each bootstrap are independently sampled from the original dataset, thus duplicates will usually appear in the bootstrap samples. In fact, it is easy to show that about $0.632n$ instances from the training set will end up in each bootstrap sample (hence the name). Let $\mathrm{acc}_u(B_i)$ be the accuracy on unseen data, *i.e.*, the accuracy of the inducer when trained on the bootstrap sample B_i and tested on the unseen instances (about $0.368n$); let $\mathrm{acc}_r(B_i)$ be the resubstitution estimate, that is, the accuracy of the inducer when trained *and* tested on the bootstrap sample. The .632-bootstrap estimate of the accuracy is

$$\widehat{\mathrm{Acc}} = \frac{\sum_{i=1}^{b} 0.632 \cdot \mathrm{acc}_u(B_i) + 0.368 \cdot \mathrm{acc}_r(B_i)}{b}$$

5 Experimental Results

We ran a best-first search from the initial node, representing the empty subset of features, until 10 node expansions did not show any improvement of more than 0.1%.[2] The number of bootstrap samples was set to 20; the subset shown is the set of features that were chosen by `Holte-II`, starting with feature number zero. Table 2 summarizes our results.

On all datasets that do not have continuous features, that is, Monk 1-3, parity, tic-tac-toe, breast-cancer, chess, mushroom, vote, and vote1, `Holte-II` has an average accuracy of 93.6%, much better than C4.5's average accuracy of 82.2%. If we ignore Monk 1, Monk 2, and parity—datasets that C4.5 does very badly on—the average accuracy for `Holte-II` is 91.2% and 88.5% for C4.5.

Holte's 1R program (Holte 1993) built one-rules, that is, rules that test a single attribute, and was shown to perform reasonably well on some commonly used datasets. Table 3 compares `Holte-II` with 1R on the same datasets used in Holte's paper. While the test sets were not the same, the small

[1] The name was inspired by Holte's paper (1993), but the algorithm bears no resemblance to Holte's 1R algorithm.

[2] For hypothyroid and sick-euthyroid the stopping criteria was 3 non-improving expansions due to the large number of instances.

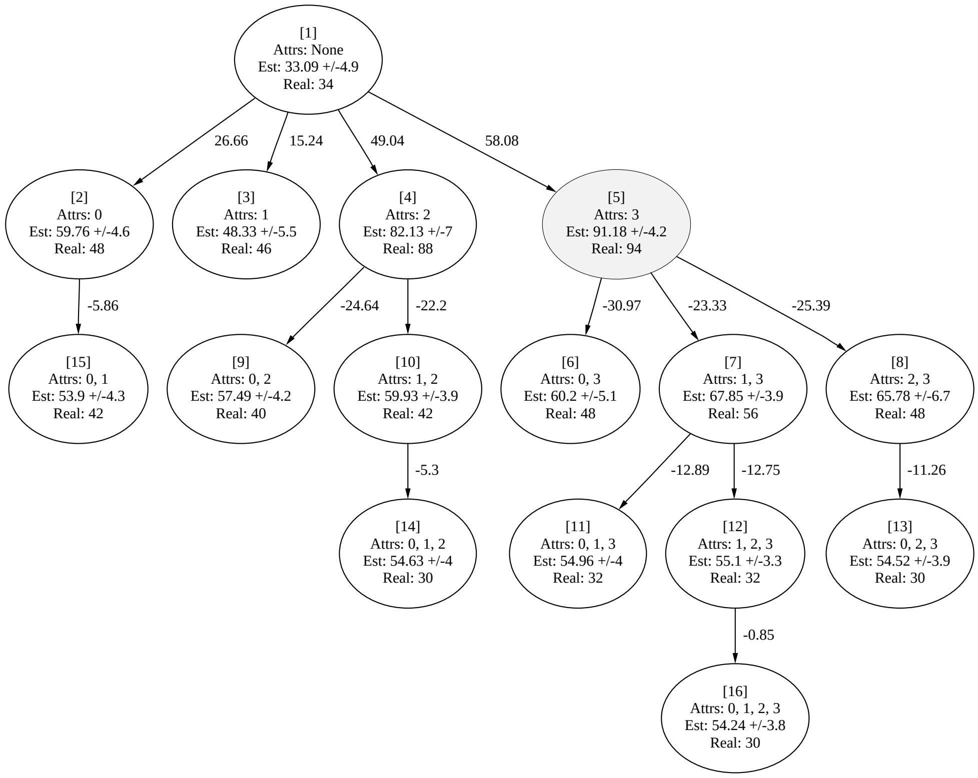

Figure 2: Best-first search through the space of feature subsets in IRIS

Dataset	Maj. Acc.	Table-Maj Acc.	C4.5 Acc.	`Holte-II` Acc.	Subset
Monk 1	50.0%	64.4%	75.7%	100.0%	0, 1, 4
Monk 2	67.1%	81.9%	65.0%	81.9%	0, 1, 2, 3, 4, 5
Monk 2-local	67.1%	81.9%	70.4%	100.0%	0, 3, 6, 8, 11, 15
Monk 3	47.2%	60.6%	97.2%	97.2%	1, 4
Parity 5+5	50.0%	54.4%	50.0%	100.0%	1, 2, 3, 5, 7
Tic-Tac Toe	65.3%	87.8%	88.2%	92.0%	0, 2, 3, 4, 5, 6, 8
Credit	55.0%	55.0%	80.0%	82.5%	3, 8, 9, 12
Breast-cancer	73.7%	74.7%	74.7%	74.7%	0, 1, 2, 5, 6, 7, 8
Chess	53.2%	53.2%	99.5%	97.7%	0, 5, 9, 13, 14, 20, 27, 31, 32, 34
Glass	30.6%	30.6%	63.9%	37.5%	2, 5
Glass2	58.2%	58.2%	72.7%	60.0%	0, 7
Heart-disease	62.4%	62.4%	74.3%	77.2%	1, 8, 11
Hepatitis	86.5%	86.5%	80.8%	82.7%	1, 2, 5, 7, 11, 18
Horse-colic	60.3%	60.3%	80.9%	67.7%	0, 6, 13, 14, 16, 21
Hypothyroid	94.8%	95.2%	99.2%	96.4%	4, 10, 12, 13, 22
Iris	30.0%	30.0%	94.0%	94.0%	3
Labor	64.7%	64.7%	82.4%	82.4%	6, 10
Lymphography	60.0%	60.0%	76.0%	76.0%	8, 10, 12, 16
Mushroom	31.1%	31.1%	100.0%	100.0%	2, 4, 10, 14, 19
Sick-euthyroid	90.4%	90.9%	97.7%	93.7%	2, 4, 6, 14, 16
Soybean-small	31.1%	31.1%	100.0%	100.0%	20, 21
Vote	64.8%	80.0%	95.2%	94.5%	2, 3, 9, 15
Vote1	64.8%	80.0%	88.3%	91.9%	0, 2, 3, 7, 8, 9

Table 2: Comparison of `Holte-II` and C4.5

difference of 0.3% in the average of the two C4.5 columns—one representing the accuracy on Holte's samples and the other representing accuracy on our samples—show that the differences are not significant.

The `Holte-II*` column represents an estimate of the upper bound possible with `Holte-II` type algorithms. The upper bound is computed by using the accuracy on the *test set* as an estimate of performance. Holte's 1R* was similarly trained. Note that this upper bound is very optimistic in some cases, but it is only an approximate upper bound, since we are still conducting a best-first search, and are not guaranteed to find the optimal feature subset.

On the real datasets taken from Holte's paper, C4.5 has a 3.5% higher accuracy. The average accuracy for `Holte-II` is 82.7%, and 86.2% for C4.5. If we ignore the two glass datasets on which `Holte-II` does poorly, the difference shrinks to 1.3%. Thus even on data with continuous features that have not been discretized, `Holte-II` does reasonably close to C4.5. Moreover, the upper-bound given by `Holte-II*` was about 1% higher than C4.5. These results compare favorably with those of Holte, where the "1R" program was 5.7% lower than C4.5, and 3.1% lower if two out of the 16 databases were ignored.

6 Reducts and Cores

An optimal classifier must use all features in one of the reducts, which must include all features in the core. Classifiers induced from data, however, are not optimal as they have no access to the underlying distribution. Induction algorithms generating such classifiers may benefit from the omission of core features, resulting in feature subsets that are not reducts. Similarly, such induction algorithms may benefit from the inclusion of features that would be superfluous to an optimal classifier.

Example 1 (Buggy Inducer) *Let $\mathcal{S}$ be an information system where the only reduct (and hence the core) contains features $\{2, 3, 4, 5\}$. To achieve the highest possible prediction accuracy, an optimal classifier must use exactly these features.*

Consider an inducer `Buggy-ind` *that induces a classifier for future predictions. Due to a bug, if feature number one is not given, or if the number of features is greater than three, the inducer produces a classifier that labels instances randomly; otherwise, a more sensible classifier is produced.*

If the dataset containing the reduct features $\{2, 3, 4, 5\}$ is given to `Buggy-ind`, *the classifier predictions will be random; however, if a subset such as $\{1, 2, 3\}$ is given, the classifier produced might have an accuracy better than random.*

While the example above might sound silly, finding minimal structures—a goal clearly stated in many induction algorithms—is NP-hard (Wong & Ziarko 1985; Hyafil & Rivest 1976; Blum & Rivest 1992). Induction algorithms usually resort to non-optimal hill-climbing techniques and may thus degrade in performance when given too many features. More generally, the biases that algorithms have may be inappropriate for a given feature subset, while more appropriate for others.

For practical induction algorithms, it may therefore be beneficial to use a feature subset that is not a reduct, or even one that does not contain all features in the core. Our experimental results indicate that this is the case. In some datasets, the feature subsets chosen (by both `Holte-II` and `Holte-II*`) are *not* reducts. In tic-tac-toe, for example, the best subset creates an inconsistent table, *i.e.*, it contains four conflicting instances, while the original training set is consistent. The advantage of forming an inconsistent table comes from the fact that the space of possible instances shrinks by a factor of 9 (each square is an X, an O, or a blank), while the number of instances shrinks by a smaller factor. The remaining instances thus form a better covering of the projected space. Creating a table without two features lowers the possible accuracy from 100 to 99.6%, but the denser space more than offsets for this loss.

7 Summary and Future Work

Generalization without a bias is impossible (Schaffer 1994). `Holte-II` is biased to select a feature subset maximizing the Bootstrap accuracy estimate. Whenever the estimates are good, `Holte-II` should choose a feature subset leading to high accuracy. However, when the Bootstrap estimates are inappropriate (Efron & Tibshirani 1993, Section 7.4), such as when the training set is not a representative of the true distribution, the selected feature subset might be inappropriate. Our results show that such a bias is indeed appropriate for the datasets used in our experiments.

We have shown that a table-majority inducer can perform well on artificial and real-world databases, if a useful set of features can be found. The `Holte-II` inducer outperforms C4.5 on the tested datasets that had only discrete features. Surprisingly, even on datasets with continuous features, the difference between `Holte-II` and C4.5 is only 3.5%, and this difference shrinks to 1.3% if we ignore the glass datasets that have only continuous features.

The fact that `Holte-II*` does very well compared to `Holte-II` may indicate that there is room for improving the accuracy estimator. We have used 20 bootstrap samples, while the usual recommended number is 50 to 200. An analysis of

Dataset	Holte's samples			Our samples		
	1R Acc.	1R* Acc.	C4.5 Acc.	C4.5 Acc.	`Holte-II` Acc.	`Holte-II*` Acc.
Breast cancer	68.7%	72.5%	72.0%	74.7%	74.7%	74.7%
Chess	67.6%	69.2%	99.2%	99.5%	97.7%	97.8%
Glass	53.8%	56.4%	63.2%	63.9%	37.5%	41.7%
Glass2	72.9%	77.0%	74.3%	72.7%	60.0%	63.6%
Heart-disease	73.4%	78.0%	73.6%	74.3%	77.2%	82.2%
Hepatitis	76.3%	85.1%	81.2%	80.8%	82.7%	94.2%
Horse-colic	81.0%	81.2%	83.6%	80.9%	67.7%	88.2%
Hypothyroid	97.2%	97.2%	99.1%	99.2%	96.4%	96.9%
Iris	93.5%	95.9%	93.8%	94.0%	94.0%	94.0%
Labor	71.5%	87.4%	77.2%	82.4%	82.4%	94.1%
Lymphography	70.7%	77.3%	77.5%	76.0%	76.0%	80.0%
Mushroom	98.4%	98.4%	100.0%	100.0%	100.0%	100.0%
Sick-euthyroid	95.0%	95.0%	97.7%	97.7%	93.7%	95.1%
Soybean	81.0%	87.0%	97.5%	100.0%	100.0%	100.0%
Vote	95.2%	95.2%	95.6%	95.2%	94.5%	97.2%
vote1	86.8%	87.9%	89.4%	88.3%	89.0%	92.4%

Table 3: Comparison of 1R, 1R*, C4.5, `Holte-II` , and `Holte-II*` on real datasets.

the results shows that the bootstrap estimates do not correlate well enough with the real accuracies. Although only the relative magnitudes of the accuracies are important, there are cases where the estimated ranking of feature subsets are very poor. One way to improve the estimates without increasing the running time considerably is to dynamically decide on the number of bootstrap samples needed. An abstract description of this problem is described in (Kohavi 1994).

For the artificial datasets: Monk 1-3, parity, and tic-tac-toe, the test sets include the space of all possible instances, and therefore the test set accuracy is the actual real accuracy. For the real datasets, our accuracy results are based on a single test set, and a more thorough experiment on multiple test sets is called for. Such an experiment, however, is important mainly for the comparison with Holte's results and for estimating the upper bound of `Holte-II`'s performance. For the purpose of comparing C4.5 against `Holte-II`, the same test set was used, and thus the relative accuracies are important more than the actual absolute values.

While we have shown that useful feature subsets are not reducts, we believe that starting the search from the core features may lead to faster identification of useful subsets, thus narrowing the search considerably. We stress, however, that any attempt to identify core features from a dataset is an induction problem by itself.

For most datasets tested here, the real valued attributes were not very useful (glass is a notable exception). In many real world datasets it is probably true that real values are necessary, and the data must undergo a discretization process (Lenarcik & Piasta 1992; Fayyad & Irani 1993).

Acknowledgments The work in this paper was done using the $\mathcal{MLC}$++ library, partly funded by ONR grant N00014-94-1-0448 and NSF grant IRI-9116399. George John has been of great help in this ongoing research. Rob Holte provided us with excellent comments and useful pointers after seeing the first draft. Tom Dietterich, Pat Langley, Ofer Matan, and Ross Quinlan provided useful comments. Pat Langley referred the first author to the work on rough sets when he saw the work on Oblivious read-Once decision graphs (OODGs). His results on Oblivion (Langley & Sage 1994) sparked the first ideas toward this paper.

References

Aha, D. W. 1992. Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. *International Journal of Man-Machine Studies* 36(1):267–287.

Ben-Bassat, M. 1982. Use of distance measures, information measures and error bounds in feature evaluation. In Krishnaiah, P. R., and Kanal, L. N., eds., *Handbook of Statistics*, volume 2. North-Holland Publishing Company. 773–791.

Blum, A. L., and Rivest, R. L. 1992. Training a 3-node neural network is NP-complete. *Neural Networks* 5:117–127.

Boyce, D.; Farhi, A.; and Weischedel, R. 1974. *Optimal Subset Selection*. Springer-Verlag.

Breiman, L.; Friedman, J. H.; Olshen, R. A.; and Stone, C. J. 1984. *Classification and Regression Trees*. Wadsworth International Group.

Devijver, P. A., and Kittler, J. 1982. *Pattern Recognition: A Statistical Approach*. Prentice-Hall International.

Efron, B., and Tibshirani, R. 1993. *An introduction to the bootstrap*. Chapman & Hall.

Fayyad, U. M., and Irani, K. B. 1993. Multi-interval discretization of continuous attributes for classification learning. In Bajcsy, R., ed., *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann.

Ginsberg, M. L. 1993. *Essential of Artificial Intelligence*. Morgan Kaufmann.

Grzymala-Busse, J. W. 1992. LERS—a system for learning from examples based on rough sets. In Slowinski, R., ed., *Intelligent Decision Support*. Kluwer Academic Publishers. 3–18.

Holte, R. C. 1993. Very simple classification rules perform well on most commonly used datasets. *Machine Learning* 11:63–90.

Hurley, R. B. 1983. *Decision Tables in Software Engineering*. Van Nostrand Reinhold data processing series.

Hyafil, L., and Rivest, R. L. 1976. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters* 5(1):15–17.

John, G.; Kohavi, R.; and Pfleger, K. 1994. Irrelevant features and the subset selection problem. In *Machine Learning: Proceedings of the Eleventh International Conference*. Morgan Kaufmann.

Kohavi, R. 1994. Feature subset selection as search with probabilistic estimates. In *AAAI Fall Symposium on Relevance*. To appear.

Langley, P., and Sage, S. 1994. Oblivious decision trees and abstract cases. In *AAAI94 Workshop on Case-Based Reasoning*.

Lenarcik, A., and Piasta, Z. 1992. Discretization of condition attributes. In Slowinski, R., ed., *Intelligent Decision Support*. Kluwer Academic Publishers. 373–389.

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., *Proceedings of the European Conference on Machine Learning*, 213–226.

Murphy, P. M., and Aha, D. W. 1994. UCI repository of machine learning databases. For information contact ml-repository@ics.uci.edu.

Neter, J.; Wasserman, W.; and Kutner, M. H. 1990. *Applied Linear Statistical Models*. Irwin: Homewood, IL, 3rd edition.

Pawlak, Z.; Wong, S.; and Ziarko, W. 1988. Rough sets: Probabilistic versus deterministic approach. *Internation Journal of Man Machine Studies* 29:81–95.

Pawlak, Z. 1987. Decision tables — a rough sets approach. *Bull. of EATCS* 33:85–96.

Pawlak, Z. 1991. *Rough Sets*. Kluwer Academic Publishers.

Pawlak, Z. 1993. Rough sets: present state and the future. *Foundations of Computing and Decision Sciences* 18(3-4):157–166.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. Los Altos, California: Morgan Kaufmann.

Schaffer, C. 1994. A conservation law for generalization performance. In *Machine Learning: Proceedings of the Eleventh International Conference*, 259–265. Morgan Kaufmann.

Skowron, A., and Rauszer, C. 1991. The discernibility matrices and functions in information systems. Technical Report Research Report, ICS WUT, Warsaw.

Skowron, A., and Ruaszer, C. 1992. The discernibility matrices and functions in information systems. In Slowinski, R., ed., *Intelligent Decision Support*. Kluwer Academic Publishers. 331–362.

Thrun etal. 1991. The monk's problems: A performance comparison of different learning algorithms. Technical Report CMU-CS-91-197, Carnegie Mellon University.

Weiss, S. M., and Kulikowski, C. A. 1991. *Computer Systems that Learn*. San Mateo, CA: Morgan Kaufmann.

Wong, S., and Ziarko, W. 1985. On optimal decision rules in decision tables. In *Bulletin of the Polish Academy of Sciences and Mathematics*, 693–696.

Ziarko, W. 1991. The discovery, analysis, and representation of data dependencies in databases. In Piatetsky-Shapiro, G., and Frawley, W., eds., *Knowledge Discovery in Databases*. MIT Press.