An MDL Estimate of the Significance of Rules

John G. Cleary, jcleary@cs.waikato.ac.nz
Shane Legg, sleg001@advlab.math.auckland.ac.nz
Ian H. Witten ihw@cs.waikato.ac.nz

Department of Computer Science,
University of Waikato,
Private Bag 3015,
Hamilton, New Zealand

ph: +64-7-838-4378
fax: +64-7-838-4155

Abstract

This paper proposes a new method for measuring the performance of models—whether decision trees or sets of rules—inferred by machine learning methods. Inspired by the minimum description length (MDL) philosophy and theoretically rooted in information theory, the new method measures the complexity of test data with respect to the model. It has been evaluated on rule sets produced by several different machine learning schemes on a large number of standard data sets. When compared with the usual percentage correct measure, it is shown to agree with it in restricted cases. However, in other more general cases taken from real data sets—for example, when rule sets make multiple or no predictions—it disagrees substantially. It is argued that the MDL measure is more reasonable in these cases. and represents a better way of assessing the significance of a rule set's performance. The question of the complexity of the rule set itself is not addressed in the paper.

Keywords: Machine learning, MDL, compression, evaluation of models, adaptive complexity

Area of Interest: Minimum Encoding Length Inference Methods

1. Introduction

So many different learning algorithms have been proposed that evaluating and comparing them presents a significant challenge. This paper restricts attention to "supervised" learning schemes which operate in the following standard way. They are given *training data* to be learned, which is divided into separate *instances*. Each instance comprises a fixed set of *attributes* and a *classification*. Each attribute has a range of values that it can assume—typically a finite enumerated set, an integer, or a real number. After learning from the training data, a scheme is presented with a set of *test data*. This is just like the original input data except that now the classification must be computed from the remaining attributes.

The result of learning is a model or *theory* of the data. Given an instance from the training or test data, the theory generates a classification which it attributes to that instance. This paper contributes to a common framework which is being developed for comparing and evaluating the various supervised learning schemes, the ultimate aim being to estimate how well a theory will perform when presented with a new set of test data.

The theories that we examine in this paper can be viewed as mappings from the attribute space to a subset of the set of possible classifications. That is, a theory T maps an instance $\overline{A}$ of attribute values

$$T: \overline{A} \mapsto \{c_1, \ldots, c_n\}$$

where the values $\{c_1, \ldots, c_n\}$ are possible classifications. Many theories (for example, decision trees) always have n=1, that is, they always make a single prediction. The framework developed here allows theories where for some instances no prediction is made(n=0) or multiple predictions are made (n>1). For the purposes of this paper, we do not allow a theory to attach probabilities to its predictions. Although, the MDL algorithm that evaluates the theory will attach probabilities to the predictions, after they have been generated.

The easiest way to estimate performance is to observe how well the theory does on the original training set and assume that it will do just as well on test sets. This has the obvious disadvantage that a theory which merely replicates the data instances with one rule per instance does very well when evaluated against the training set but is likely to perform poorly on a new data set because it has not generalised the original data. Such a theory is said to be *overfitted* to the training set. This situation is normally avoided by measuring performance on a test data set which is different from the training data used to form the rules.

There are other problems with measuring a theory's performance. The obvious metric is the frequency of correct classifications. But even such a simple measure raises practical problems. Theories may generate no classification for a particular instance, or multiple classifications; or the instance may have missing data values. Different schemes resolve these ambiguities in different ways, making it difficult to replicate results reported in the literature and rendering comparisons between schemes dubious, if not meaningless. Another difficulty is encountered with highly-skewed data sets. Consider, for example, a situation in which one classification occurs 99% of the time. A trivial theory which always predicts that classification has a 99% success rate—apparently a very good result. The frequency of correct classifications is not necessarily a useful overall measure of performance.

The "minimum description length" (MDL) measure, proposed by Rissanen and others, is an alternative way of assessing inferred theories (Rissanen 1985, 1989). The MDL principle deems the best theory for a set of data to be one which requires the smallest amount of information to specify both the theory and the original data given the theory. Information is measured in bits, assuming an efficient coding scheme for theory and data. In order to apply the minimum description length principle to the output of machine learning schemes we have had to come up with general ways of computing complexity, which are described below.

This paper shows the performance of rule sets generated by different learning schemes can be measured in a uniform way, and describes a system for doing so. Whereas most previous work (eg. Quinlan and Rivest, 1989; Wallace and Patrick, 1993) concentrates on the question of evaluating the complexity of theories, this paper examines how to evaluate the complexity of a data set given a theory. This is normally dismissed as a fairly trivial part of the problem, but in fact we argue that it involves some crucial issues that are not generally considered. We back up this claim by examining the results of several commonly-used machine learning schemes on actual datasets—both the standard datasets and ones encountered in a project on the agricultural applications of machine learning (Garner *et al.*, 1995)—and showing that the new metric does capture subtleties that are hidden by the usual percent correct measure.

The next section discusses how the notion of complexity can be quantified, and introduces the key distinction between static and adaptive estimation methods. It also briefly discusses measurement of the complexity of theories, which is not further addressed in this paper. Following that, we introduce two different ways of measuring the complexity of data with respect to a theory. One takes account of the frequency of different items, and the other does not. Next we use these metrics to examine the significance of actual rule sets generated by machine learning schemes on a wide variety of different datasets. In a certain special case it is possible to relate the new measures to the standard "percentage correct" measure theoretically; this confirms the correctness of the results. Finally we summarise our findings and draw conclusions from the work.

2. Complexity

According to the minimum description length principle, the "best" theory is the one that represents the data in the simplest, most economical, way. The complexity of a data set relative to a theory is defined as the length of the shortest program necessary to reconstruct it from that theory. The total information required to represent the data is the amount of information necessary to specify the theory, plus the information necessary to specify the data given the theory. This can be written

$$\mathrm{I}(D) = \mathrm{I}(T) + \mathrm{I}(D|T)$$

where $\mathrm{I}(X)$ is the information necessary to specify X, and $\mathrm{I}(X|Y)$ the amount of information necessary to specify X when Y is known. This formulation encapsulates a trade off between a complex, over-fitted, theory where $\mathrm{I}(T)$ is large and $\mathrm{I}(D|T)$ small, and a simple, over-general, one where $\mathrm{I}(T)$ is small and $\mathrm{I}(D|T)$ large.

Our purpose here is to compare different theories on the same data. The measure above includes information that is irrelevant for this purpose. The regenerated data constitutes not only the classification that is assigned by the theory, but also all other attributes of the test instances. One might view the attributes being presented as a question posed to the theory, and the returned classification as the answer to this question. In these terms, the complexity of the questions is being included along with that of the answers. It can be eliminated by splitting the complexity measure into the information required to specify the attributes in the test instances, and that required to specify the class given both the theory and the attributes. That is,

$$I(D|T) = I(Q) + I(D|T,Q),$$

where $I(Q)$ is the complexity of the attributes (the questions) and $I(D|T,Q)$ is the complexity of the data given both the theory and the questions. The total complexity of interest is then:

$$I(D) = I(T) + I(Q) + I(D|T,Q).$$

$I(Q)$ does not depend on the theory T and can be suppressed for comparison purposes. To compare different theories it is sufficient to compute the sum $I(T) + I(D|T,Q)$, in other words, the complexity of the theory plus the complexity of the answers. We show below one way of estimating these two quantities; first, however, we discuss the general philosophy that underlies our approach.

Static versus adaptive complexity estimation

The complexity $I(T)$ of a theory is defined as the smallest program needed to generate it, and the complexity $I(D|T)$ of a set of data relative to a theory is the smallest program needed to generate that data, given the theory. In the theory of complexity, a standard machine is assumed on which these programs are to be executed. The particular machine chosen is deemed to be unimportant because in the limit for large theories and data sets it makes no difference—any machine can be interpreted on any other using a program of fixed size. Even if a particular machine is assumed, complexity theory does not prescribe how to compute these measures: indeed, the question of whether any particular program is in fact the smallest is undecidable in general. This makes it necessary to devise a methodology for estimating the complexities.

Let us consider, by way of example, the complexity of a theory T. One way to estimate $I(T)$ is to specify T by a sequence of decisions that build the theory up. Representing T as a sequence of binary digits, a decision would be made for each digit about whether it was a 0 or 1. If the probability of each decision's outcome were known, a complexity equal to the entropy of the outcomes could be attained. For example, if it were known *a priori* that zeros would occur three times as often as ones in the theory's binary representation, the sequence could be encoded at $-0.25 \log_2 0.25 - 0.75 \log_2 0.25 = 0.81$ bits per binary digit, while if they occurred equally often the theory could be encoded no more efficiently than one bit per binary digit. Using the technique of arithmetic coding, this entropy can be approximated arbitrarily closely (Witten *et al.*, 1987).

Summing the entropy of each outcome only gives a good estimate of the entropy of the sequence if the individual bits are chosen independently. In practice, this will not be the case and a more complex model would be used to encode the theory. However, the real disadvantage of this coding regime is much more serious. If the *a priori* probabilities are not chosen correctly—that is, if they do not reflect the frequencies with which elements of the sequence actually occur—the coding will be inefficient. And the inefficiency will take the form of a constant overhead *per element of the sequence*, so that there is no limit to the inefficiency with which large theories are coded. Contrast this with the choice of standard machine in the alternative, program-oriented, view of complexity, which can only affect the complexity measure by an additive constant.

The problem with the method we have described is that it uses a *static* model to encode the theory, and any deficiency in the model results in inefficient coding of each element of the sequence. An alternative is to use *adaptive* coding techniques instead. The idea is that, as a sequence is processed, statistics are gathered about the decisions that are made. Each prediction about what will occur next is informed by the accumulated statistics so far. Thus if it is thought *a priori* that zeros will occur three times as often as ones, but this turns out to be incorrect, the adaptive model will not suffer the same penalty as a static one.

Adaptive techniques have been used very successfully in many applications. For lossless compression of text, all the best methods are adaptive (Bell *et al.*, 1990). An important property of adaptive techniques is that they are very robust. In the limit of long messages, they can guarantee to achieve the entropy of the sequence plus a term that is logarithmic in message size (Bell *et al.*, 1990; Cleary and Witten, 1984). In contrast, the use of predetermined statistics can cause the data to be expanded by an unbounded constant factor. For any particular problem, adaptive algorithms can be constructed in many different ways for computing the complexity, depending on what statistics are accumulated and how they are used to estimate predictions. The main advantages of adaptive techniques are their robustness and flexibility: these allow them to be applied easily to different problems.

Theory Complexity

As part of our overall system an adaptive modeller to compute the complexity of general Prolog programs was developed. However, the output from supervised learning schemes are very stereotyped: for example, many of them can be represented by decision trees. The result is that the general complexities computed are significantly greater than the apparent complexity of the theory—so much so that the resultant values for I(T) appears not to be useful for comparing different theories. We are currently re-implementing a system that recognises special subtypes of theories (decision trees, rules using a single attribute, sets of rules, trees of rules, and so on) and computes the adaptive complexity from these.

3 Data Complexity

We now describe a sequence of more refined adaptive algorithms for computing I($D|T,Q$). The way that these algorithms are used in our experiments later is that first a learning technique is applied to some training set of instances. The result of this is a theory T. Then a sequence of test instances (Q) and their associated classifications (D) are adaptively encoded. It is the total size of the resulting encoding that is the value we use for I($D|T,Q$) and which in turn forms the measure of the significance of the theory.

The classification of the members of the data set D are each specified in turn. For each one, the theory generates a set of predictions (say m of them) out of a universe of possible classifications (say n of them); our task is to encode which classification actually occurs.

This can be specified in two steps. The first determines whether or not the actual class appears as one of the predictions. If so, it is only necessary to specify which of the m predictions it is. The theory provides no way to distinguish the different predictions, so this requires $\log_2 m$ bits. If the actual class is not one of the predictions, $\log_2(n-m)$ bits are needed to specify it. It remains only to determine how to specify whether or not the actual class is in the prediction set. A simple adaptive approach is to count how often the actual class has been predicted (say c_p) and how often it was not in the prediction set (say c_n). The probability of the class being predicted can be estimated as $p = (c_p+1)/(c_p+c_n+2)$: this is Laplace's law of succession (the 1 and 2 in numerator and denominator allow for the case where c_p or c_n is 0 because the class has not yet occurred). The number of bits needed for encoding is $-\log_2 p$.

This measure yields some qualitatively correct results. For example, suppose the theory is perfectly correct and makes a single correct prediction for each instance in the test data. Then p will approach 1 (and $-\log_2 p$ will approach 0). Because there is only a single prediction, no bits are needed to specify the class ($\log_2 m = \log_2 1 = 0$), and so in the limit the data requires zero bits per instance to specify it. Now consider a situation where the theory always specifies two classifications. On average, one bit will be required per instance: zero bits for selecting whether there will be a correct prediction, and one to identify it from the two predictions.

Nevertheless, the estimator is unsatisfactory for three reasons. If there are no predictions (m=0), the class cannot possibly be in the prediction set and yet it is predicted with non-zero probability—thus wasting output bits. Similarly, if m=n it is impossible for the class to occur, yet it is predicted with non-zero probability. A third, more subtle, problem arises when considering the maximum average complexity per instance. Ignoring the predictions and specifying the class requires $\log_2 n$ bits per instance, and adaptive coding should never do worse than this. However, suppose the behaviour of the predictions alternates. First, $n-1$ classes are predicted and the actual class is among them. Then just one class is predicted, which is incorrect. If these two cases continue to alternate, the probability of a class being predicted correctly will tend to 1/2. In both cases the actual class is selected from among $n-1$ alternatives, so the complexity per instance is $1 + \log_2(n-1)$ bits, which exceeds $\log_2 n$ whenever n>2. In this case, the adaptive technique is strictly worse than ignoring the theory altogether.

Constant Weighted Complexity

These problems can be solved with a more sophisticated probability estimation technique. This uses two weights α and β. The former applies to the case when some prediction is correct, the latter when they are all incorrect. The probability p is computed as $p = \alpha m/(\alpha m+\beta(n-m))$. This deals correctly with the cases where nothing is predicted (m=0) because then p=0 and no bits are wasted predicting that the class is in the prediction set. Similarly, when all possible classes are predicted (m=n), p=1.

It is not obvious how best to update α and β as evaluation proceeds. The procedure used is that as each instance is evaluated, α is replaced by $\alpha + 1/m$ if the actual class occurred amongst the m predictions, otherwise β is replaced by $\beta + 1/(n-m)$. Note that division by zero can never occur, for if m=0 the actual class can never be predicted so α will not be incremented; similarly if m=n the actual class must be in the prediction set so β will not be incremented. This procedure passes two important tests for reasonableness.

1. In the special case where the theory always generates the same number of predictions (m is constant), then the probability p is, in the limit, the probability that the theory will make a correct prediction. Moreover, the term αm counts the number of times that the theory predicted correctly, while $\beta(n-m)$ counts of the number of times it predicted incorrectly.
2. In the limit, the complexity per instance is less then $\log_2 n$ bits per instance no matter what sequence of correct and incorrect predictions are made by the theory.

These two properties hold no matter what initial values are used for α and β. Choosing the "best" value is one form of the zero-frequency problem (Witten and Bell, 1991) which has no optimum solution unless a prior for α and β is assumed. The current scheme initialises α and β to $1/n$, the smallest amount by which either can be incremented. This complexity calculation is summarised in Table 1.

The complexity measure just described will be referred to as the "constant weight" complexity of the data and written $I_c(D|T,Q)$. However, there is a problem that has arisen when dealing with data where one class has a frequency close to 1—which, in our experience, often occurs in practical applications of machine learning. The measure takes no account of frequency information about the individual classes themselves. A simple system that codes the data adaptively by accumulating the frequency of each class (without using the theory predictions) will often compress the data more than by using the theory as above. This suggests that both the frequency of the classes and the theory predictions should be taken into account.

Frequency Weighted complexity

First consider how to adaptively code the data using just the frequency of the classes (but not the theory). Let the number of times that class i has occurred so far during the scan of the data be c_i. Let the total number of instances be

$$N = \sum_i c_i .$$

The probability of the i'th class is then estimated as $p_i = (c_i + 1)/(N + n)$. The addition of the 1 and n deals with the zero-frequency case when a class has never been seen before but must be predicted with a non-zero probability. In the limit, the complexity of this scheme will be at worst $\log_2 n$ bits per instance (this worst case will occur when all the classes have equal probability $1/n$). The complexity computed in this way is independent of the theory and is written $I_f(D|Q)$.

To incorporate both predictions and frequency information, the two-step procedure described earlier is used—that is, weights α and β are used to predict whether the actual class is in the predicted set or not, whereupon the class is predicted from within the appropriate set. Let R be the set of classes predicted by the theory. The probability that the actual class is in the predicted set is estimated as:

$$p = \frac{r\alpha}{r\alpha + (1-r)\beta} .$$

Here, r is the probability that a class will be in the predicted set,

$$r = \sum_{i \in R} p_i .$$

Here α and β again function as weighting factors, estimating the probability that the theory predictions will be correct. As each instance is evaluated, α is replaced by $\alpha + p_i / r$ if the actual class occurred amongst the m predictions, otherwise β is replaced by $\beta + p_i / (1-r)$.

This new form of complexity is referred to as the "frequency weighted" complexity and written $I_f(D|T,Q)$. The complexity calculation is summarised in Table 1. It gives reasonable results in special cases. For example, if all the classes occur with equal frequency these formulae reduce to just those used above for the constant weight complexity. Also, in the limit, it cannot exceed the simple frequency based complexity which ignores the theory.

Comparison With Other MDL Techniques

Muggleton (1992) proposed using the MDL principle for evaluating learned rules, following closely the seminal ideas proposed by Rissanen (1985).

The major difference from our work is that Muggleton used proof complexity when computing the complexity of the data. Proof complexity encodes the result of executing a theory in terms of the sequence of choices made by a Prolog interpreter while generating the actual class. In contrast, answer complexity, as described in this paper, first collects all the answers and then determines the complexity of the actual class from the answer set.

As has been noted in (Kovacic,1994; Conklin and Witten 1994; and Srinivasan *et al*, 1992) proof complexity can be a very inefficient way of specifying the class: the average complexity of a proof is always larger than the answer complexity and can be unboundedly larger. Think of the proof tree (ie. the SLD tree): every answer must appear once on some leaf, but may appear many times on different leaves, and there may be failure or infinite branches that contain no answer. To specify the actual answer in a proof tree must always involve more choices on average than to specify it in the set of answers.

4. Significance of Theories

We use the complexity measures to evaluate theories, in other words to assess whether a theory is "significant" or not, or to answer the question "does this theory provide significantly better predictions than the null hypothesis".

Our procedure is to take a learning algorithm and apply it to a training set, generating a theory. The theory is then applied to a different test set and the various complexity measures computed. $I_f(D|Q)$ is a measure of the data complexity without using any theory and can be taken as the complexity of the data with respect to the null hypothesis. $I_f(D|T,Q)$ is a measure of the complexity of the data with respect to the generated theory and if the theory has actually captured some regularity or information from the data should be less than $I_f(D|Q)$. The difference $S_f=I_f(D|Q)-I_f(D|T,Q)$ indicates the extent to which the theory has departed from the null hypothesis and can be used as a measure of the significance of the theory. The larger the difference, the more likely it is that the theory has captured some regularity in the data.

In order to fully determine whether the theory provides a worthwhile improvement over the null hypothesis, it would be necessary to take into account the complexity of the theory itself, and see whether it could be represented in less than S_f bits. This is beyond the scope of the present paper.

Experiments

To assess this significance measure a number of experiments were run. In each one a data set was split randomly into two equal training and test subsets 25 times. Equal splits were made because the complexity measures are not linear on small numbers of instances: keeping the test and training sets equal allows easier comparison of the complexities on the two sets. Several different learning schemes were applied to these splits:

- 1-R (Holte, 1993)
- C4.5 (Quinlan, 1986, 1993), generating
  - a pruned decision tree
  - an unpruned decision tree
  - decision rules;
- FOIL (Quinlan, 1990)
- INDUCT (Gaines, 1991), generating
  - ripple-down rules
  - DNF rules.

In each case, S_f was computed. The data sets used were those used by Holte (1993) and a number of agricultural data sets supplied by the Weka project (Garner *et al.*, 1995).

Relating complexity to percent correct

To assess the reasonableness of S_f as a significance measure we relate it to the standard percentage correct measure. The usual way that the latter is used is to compare the percentage of correct predictions made by the theory against the percentage correct if just the most likely class had been predicted. At first sight it is not clear what relationship there might be between the two measures. However, in some special cases it is possible to analytically compute the expected S_f as a function of the percentage correct.

Consider a data set where there are exactly two classes, and assume that the theory always predicts exactly one class. Let the number of instances of the most likely class be i and the corresponding probability be $p=i/N$. Let the number of instances correctly predicted by the theory be j, with corresponding probability $q=j/N$—in our experimental design q is the percentage correct measure. We now compute S_f as a function of p and q.

It is easily shown that

$$I_f(D|Q)=\log_2\frac{(N+1)!}{i!(N-i)!}$$

and

$$I_f(D|T,Q) = \log_2 \frac{(N+1)!}{j!(N-j)!}.$$

Applying Stirling's approximation to these expressions, simplifying, and neglecting terms of $O(N^{-1})$ and less gives:

$$S_f = N(E(p) - E(q))$$
$$-\tfrac{1}{2}\left[\log_2 \frac{p}{q} + \log_2 \frac{1-p}{1-q}\right]$$

where

$$E(p) = -p\log_2 p - (1-p)\log_2(1-p).$$

Note that S_f is 0 whenever $p=q$. Also the second term, which is constant in *N,* is small on most actual examples.

E(q) is a decreasing function of q (for $q \geq 1/2$) so S_f increases monotonically as q increases beyond the default probability p. Thus the percentage correct and S_f measures agree on the ordering of theories in such cases.

Results

Figure 1a shows the result of one experiment. S_f and percentage correct are plotted together for the dataset CH taken from (Holte, 1993), in which the class is binary. The learning schemes C4.5-pruned, C4.5-unpruned, C4-rule and Induct-ripple all produce just one prediction for each instance in this case. The theoretical value for S_f derived above is also shown and is seen to lie very close to the data points for these schemes—the expanded view in Figure 1b emphasises this point.

The schemes FOIL and Induct-DNF depart markedly from single predictions. The former has between 40% – 80% multiply classified instances and approximately 10% unclassified instances; the latter has between 3% – 8% multiply classified instances and 0% – 1% unclassified instances. Their points are scattered well off the theoretical curve.

Interestingly, while the percentage correct figures for the Induct-ripple scheme lie clearly outside the range for the majority of the schemes, the value for S_f is not correspondingly reduced. In the presence of multiple and unclassified instances the predictions of percentage correct and S_f may depart markedly.

This point is emphasised by the results in Figure 2. This shows the plot of percentage correct versus S_f on dataset "wcr", one dataset from an agricultural problem involving cow-culling (McQueen *et al.*, 1995). The data consists of some 2000 instances of data about a single farmer's cows over a period of five years. As well as data about the cows milk production, breeding and other attributes, it also records which cows were culled from the herd and which died from other causes. The goal is to determine what rules the farmer used for deciding which cows to cull.

There are several problems with this data. The frequency of culling is very low—only 6.1% of the cows were culled. The frequency of cows which died was also low—1.6%. These instances were essentially random with respect to the given data. In work using the standard percentage correct measure to evaluate the results, we have found that standard learning schemes seem to be unable to develop any significant theories to explain the data.

Although strictly speaking the conditions for the analytic curve derived above do not hold in this case, for there are more than two classes, nevertheless the curve has been plotted using the probability of the most frequent class for p. Some of the schemes (C4.5-rule, and C4.5-pruned) lie close to it, while others depart markedly.

Of all the schemes only Induct-DNF has a consistently positive S_f, and thus seems to have captured some of the structure of the problem. Closer investigation shows that it has a high level of multiply classified instances (60% - 90%). Even more remarkably, this is accompanied by a very low percentage correct score. Using a traditional analysis the fact that these rules had captured a significant part of the data's structure would have been missed entirely.

Preliminary results from other agricultural data sets highlight the importance of rules which make multiple (or no predictions) and consequently of having an evaluation methodology which takes these into account. For example, in a study on the grading of venison the class is essentially a discretisation of a continuous variable. Thus it is often difficult to separate which of two adjacent grades should be chosen. The most successful rules seem to predict two possible classes in these cases.

In another set of experiments, the 16 data sets used by Holte (1993) were tested against seven different learning schemes, including the 1R and C4.5-pruned schemes that Holte used. The values for percentage correct and S_f averaged over the 25 random splits are listed in Table 2, which also includes the frequency of the default class for each data set (the percentage correct values are all shown in bold face). The Sf values generally corroborate Holte's conclusions about the relative performance of 1R and C4.5. That is, 1R generally does well compared to C4.5 except on two data sets CH and SO. It is notable, however, that on the data set GL S_f indicates 1R performing worse than the percentage values would otherwise indicate. There is little difference between pruned and unpruned C4.5 (this is to be expected as generally both these schemes generate rules which make single predictions and so the percentage correct and S_f should give comparable orderings). The major differences between the two measures occur for FOIL on G2 and LY and Induct-DNF on LY. In each of these cases, the S_f value indicates higher significance than percentage correct. This seems to be a consequence of these schemes generating rules with multiple and unclassified instances which may still perform well in these cases.

6. Summary and Conclusions

An adaptive MDL measure of the performance of rule sets has been proposed. It has been argued that it:

- provides an unbiased measure of different learning schemes performance by evaluating theories independent of the original software;
- is theoretically well founded in complexity theory;
- deals in a principled way with theories that sometimes make multiple or no predictions;
- provides a measure of the significance of a theory
- agrees in simple cases with the "percentage correct" measure;
- deals in a satisfactory way with data where one class has a probability close to 1.0.

It has been applied to several large agricultural data sets, and found to provide an intuitively reasonable account of the performance of different rule sets on this data. The work underlines the fact that it is important for rule sets to be able to make ambiguous predictions in some cases: evaluation metrics must deal with this gracefully and correctly.

The work would be further enhanced by a complexity measure for theories, in order to detect overfitting and provide estimates of performance on unseen test data. An attempt to compute theory complexity by using general Prolog rules failed in practice because actual rules tend to be very specialised. In future work we intend to implement a more specialised measure of rule complexity, assess its ability to measure overfitting, and use it to predict performance on unseen data.

Acknowledgments

We would like to thank all members of the WEKA team, particularly Stephen Garner and Luke Guyton, for help in implementation and generating the results.

References

Bell, T.C., Cleary, J.G. & Witten, I.H. (1990) *Text compression.* Prentice Hall, Englewood Cliffs, NJ.

Cleary, J.G. and Witten, I.H. (1984) "A comparison of enumerative and adaptive codes." *IEEE Trans Information Theory IT-30*(2): 306–315; March.

Conklin D., and Witten I.H(1994) "*Complexity-Based Induction*," Machine Learning, vol. 16.

Gaines, B.R. (1991) "The tradeoff between knowledge and data in knowledge acquisition." In *Knowledge discovery in databases*, edited by G. Piatetsky-Shapiro and W.J. Frawley. AAAI Press, Menlo Park, CA, pp. 491–505.

Garner, S.R., Cunningham, S.J., Holmes, G., Nevill-Manning, C.G. and Witten, I.H. (1995) "Machine learning in practice: experience with agricultural databases." *Workshop on Applications of Machine Learning*, Tahoe City, California; July.

Holte, R.C. (1993) "Very simple classification rules perform well on most commonly-used datasets." *Machine Learning 11*, 63–91.

Kovacic, M.(1994) "*MDL-Heuristics in ILP Revisited*," Workshop on Applications of Descriptional Complexity to Inductive, Statistical, and Visual Inference.

McQueen, R.J., Garner, S.R., Nevill-Manning, C.G. and Witten, I.H. (1995) "Applying machine learning to agricultural data." *J Computing and Electronics in Agriculture*, *12*(4), 275–293.

Muggleton, S., Srinivasan, A., and Bain, M. (1992) "Compression, Significance and Accuracy" Proc Machine Learning Conference, pp. 338–347.

Quinlan, J.R. (1986) "Induction of decision trees." *Machine Learning 5*, 239–266.

Quinlan, J.R. and Rivest, R.L. (1989) "Inferring decision trees using the minimum description length principle." *Information and Computation 80*: 227–248.

Quinlan, J.R. (1990) "Learning logical definitions from relations." *Machine Learning 1*, 81–106.

Quinlan, J.R. (1993) *C4.5: programs for machine learning*. San Mateo, CA.

Rissanen, J. (1985) "Minimum description length principle," *Encyclopedia of the Statistical Sciences, Vol. 5*, edited by S. Kotz and N.L. Johnson. Wiley, NY, 523–527.

Rissanen, J. (1989) *Stochastic Complexity in Statistical Inquiry*. World Scientific, Singapore.

Srinivasan A., Muggleton S., and Bain M.(1992) "*Distinguishing Exceptions from Noise in Non-Monotonic Learning*," Proc. of the 2nd Int. Workshop on Inductive Logic Programming, Tokyo.

Wallace, C.S. and Patrick, J.D. (1993) "Coming decision trees." *Machine Learning 11*: 7–22.

Witten, I.H., Neal R., and Cleary, J.G. (1987) "Arithmetic coding for data compression." *Communications of the Association for Computing Machinery 30* (6) 520–540, June.

Witten, I.H. and Bell, T.C. (1991) "The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression." *IEEE Trans Info Theory 37*(4): 1085–1094.

	Probability of Actual Class	Updates
$I_c(D\|Q)$	1/n	
$I_c(D\|T,Q)$ if in predicted set R if not in R Initialization	$\frac{\alpha m}{\alpha m+\beta(n-m)} \times \frac{1}{m}$ $\frac{\beta(n-m)}{\alpha m+\beta(n-m)} \times \frac{1}{n-m}$ $\alpha \leftarrow 1/n \quad \beta \leftarrow 1/n$	$\alpha \leftarrow \alpha + 1/m$ $\beta \leftarrow \beta + 1/(n-m)$
$I_f(D\|Q)$	$c_i+1/N+n$	
$I_f(D\|T,Q)$ if in predicted set R if not in R Initialization	$\frac{\alpha r}{\alpha r+\beta(1-r)} \times \frac{p_i}{r}$ $\frac{\beta(1-r)}{\alpha r+\beta(1-r)} \times \frac{p_i}{1-r}$ $\alpha \leftarrow 1/n \quad \beta \leftarrow 1/n$	$\alpha \leftarrow \alpha + p_i/r$ $\beta \leftarrow \beta + \frac{p_i}{1-r}$

Where n is the number of classes

c_i is the number of occurrences of class i

R is the set of classes predicted by the rule set

$m = |R|$

$N = \sum c_i$

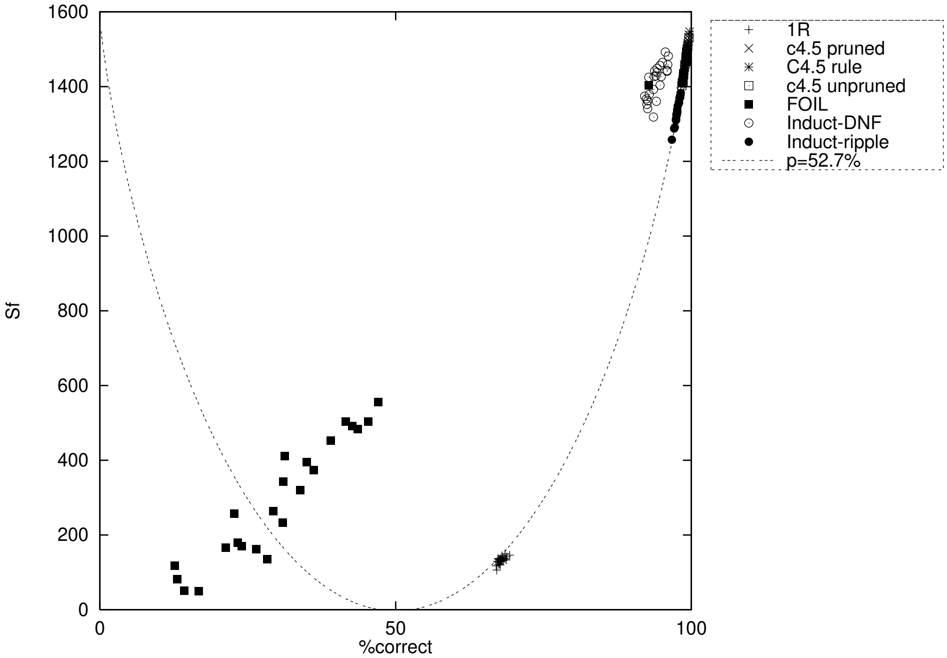
$p_i = c_i + 1/N + n$

$r = \sum_{i \in R} p_i$

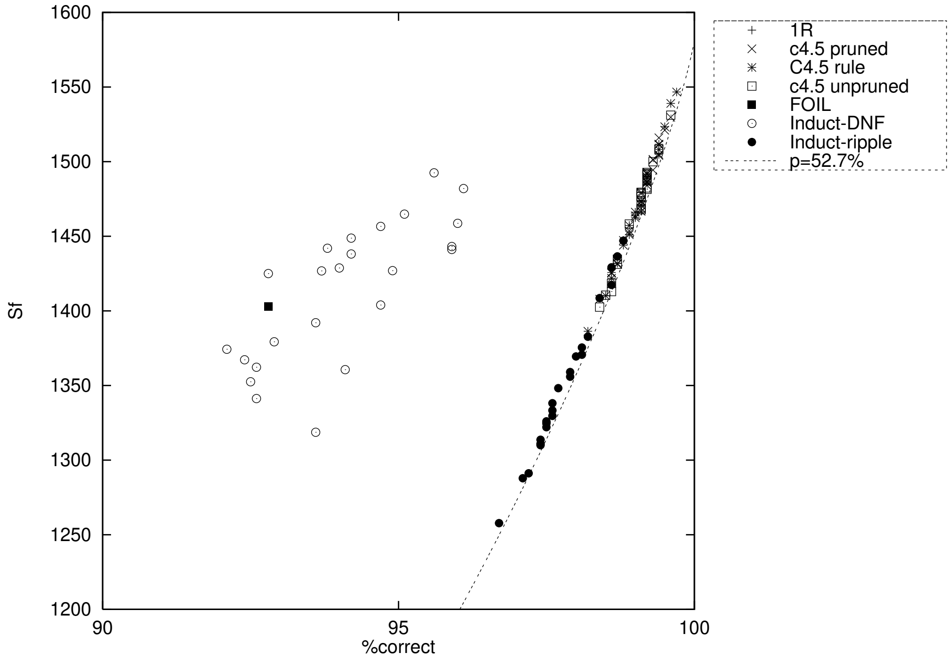
Table 1. Algorithms for Computing $I(D|T,Q)$.

Table 2. Average Percentage Correct and S_f for Datasets from (Holte, 1993).

Scheme	BC	CH	G2	GL	HD	HE	HO	HY	IR	LA	LY	MU	SE	SO	V1	VO
Baseline accuracy	**70.3**	**52.7**	**53.4**	**35.5**	**54.5**	**79.4**	**63.0**	**95.2**	**33.3**	**64.9**	**54.7**	**51.8**	**90.7**	**36.2**	**61.4**	**61.4**
1R	**68.4**	**67.9**	**72.0**	**49.6**	**70.5**	**71.4**	**81.1**	**89.5**	**93.1**	**45.6**	**71.4**	**98.5**	**64.5**	**78.5**	**85.7**	**93.7**
	-4.1	132.2	9.0	14.3	23.6	1.3	50.6	181.3	85.3	-0.8	13.9	3603.9	119.2	19.4	91.0	153.4
C4.5 pruned	**70.6**	**99.2**	**74.2**	**63.7**	**73.0**	**69.7**	**78.2**	**91.0**	**94.1**	**65.7**	**74.8**	**100.0**	**75.4**	**95.8**	**84.4**	**92.9**
	-3.5	1484.0	12.0	56.0	31.0	1.7	47.8	251.4	89.2	1.9	18.8	4053.4	246.6	36.7	99.3	149.9
C4.5 rule	**68.6**	**99.0**	**75.8**	**62.6**	**75.8**	**81.0**	**81.2**	**99.1**	**94.1**	**82.4**	**74.4**	**99.9**	**97.4**	**95.3**	**89.2**	**94.7**
	-3.2	1469.9	14.4	48.4	35.5	0.5	45.2	318.7	89.5	6.3	17.4	3994.3	433.5	36.0	99.3	142.6
C4.5 unpruned	**66.9**	**99.0**	**74.4**	**63.3**	**72.3**	**60.8**	**44.7**	**88.1**	**94.1**	**53.3**	**74.4**	**100.0**	**73.8**	**95.8**	**81.1**	**91.1**
	-1.4	1468.9	12.4	54.6	29.5	1.8	6.4	222.4	89.4	0.3	18.2	4053.4	206.6	36.7	88.4	142.0
FOIL	**54.3**	**31.5**	**63.5**	**47.5**	**64.1**	**66.9**	**60.9**	**97.9**	**90.9**	**64.9**	**63.3**	**99.6**	**95.0**	**95.5**	**76.9**	**87.6**
	2.7	316.6	20.8	62.0	47.0	6.1	39.0	298.7	88.9	6.4	26.0	4037.4	398.7	39.8	94.4	142.2
Induct-DNF	**51.1**	**94.1**	**63.5**	**43.2**	**60.5**	**67.4**	**69.8**	**95.1**	**84.6**	**69.6**	**65.8**	**99.9**	**84.2**	**95.7**	**81.8**	**88.4**
	-1.6	1413.6	16.0	57.8	39.7	4.7	55.2	312.4	86.2	5.0	29.3	4051.0	364.0	39.9	114.9	152.2
Induct-ripple	**65.1**	**97.8**	**73.2**	**61.7**	**70.7**	**78.1**	**80.0**	**98.6**	**93.8**	**80.0**	**76.0**	**100.0**	**96.3**	**97.1**	**87.9**	**94.0**
	-1.3	1350.8	11.6	46.7	23.0	0.2	39.8	266.9	88.1	4.7	22.1	4048.6	338.6	38.0	90.8	136.6



a)



b)

Figure 1. Comparison of Percentage Correct and S_f for "CH" Dataset.

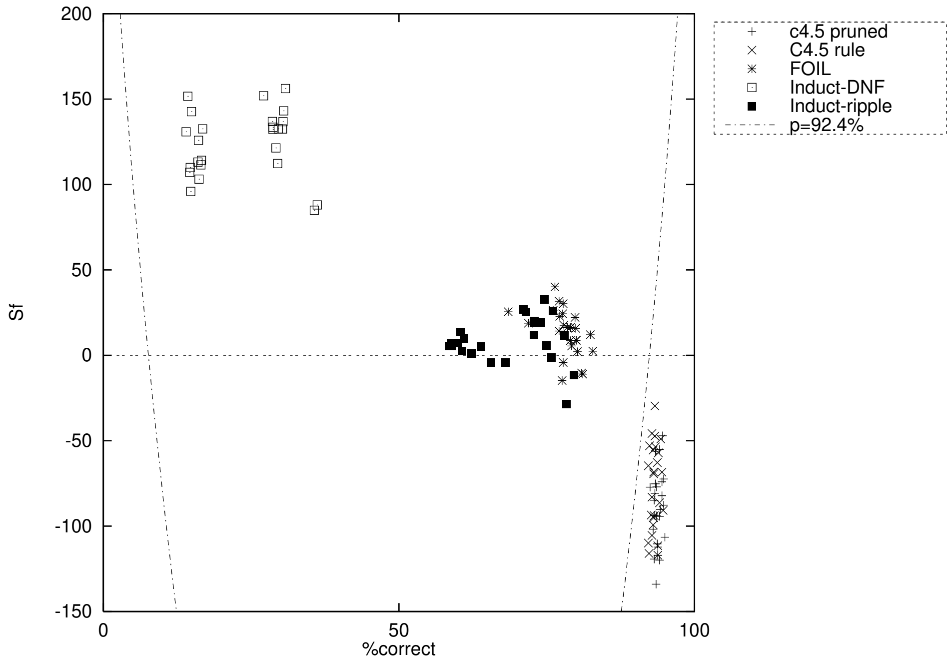


Figure 2. Comparison of Percentage Correct and S_f for "wcr" Dataset.