Trading off coverage for accuracy in forecasts: Applications to clinical data analysis

Michael J Pazzani, Patrick Murphy, Kamal Ali, and David Schulenburg
Department of Information and Computer Science
University of California
Irvine, CA 92717
{pazzani, pmurphy, ali, schulenb}@ics.uci.edu

1. Introduction

Machine learning algorithms that perform classification tasks create concept descriptions (e.g., rules, decision trees, or weights on a neural net) guided by a set of classified training examples. The accuracy of the resulting concept description can be evaluated empirically by asking a classifier to use the concept description to classify a set of test examples. One goal of this evaluation is to determine the accuracy of the learning algorithm if it were applied in a "real" application. In such an application, it is assumed that a classifier is produced on a set of training cases and a decision is made automatically on each new case based upon a forecast of the classification of the case. However, in some applications, it is advantageous not to produce a classification on every example. In particular, when a machine learning program is being used to assist a person to perform some task, it might be desirable to have the machine automatically make some decisions while allowing others to be made by the person. The general idea is that in some cases, a user of a machine learning system might be willing to allow the learning system to produce no classification on some examples. This may be acceptable if it means that when the system makes a prediction, the prediction is more accurate.

Giving a classifier the ability to decide whether to make a prediction can be important in clinical settings where there are varying amounts of relevant data available on each case. When there is incomplete or inadequate data, a system that makes no prediction may provide more useful information than a system that makes its best "guess" on every case.

In this paper, we will explore how to modify a variety of machine learning algorithms to trade off coverage for accuracy. The key to these modifications is that learning components of many algorithms already have some internal measure of the quality of their concept description. These measures are typically used by the learner to favor one hypothesis over another, or to decide whether to prune a hypothesis. By giving the classification component access to these measures, we can allow the classifier to decide to classify only those examples for which its prediction is more likely to be correct. Here, we will show how the following algorithms can be modified to trade off coverage for accuracy:

- A neural network trained with back propagation (Rumelhart, Hinton, & Williams, 1986).
- A Bayesian classifier (Duda & Hart, 1973).
- A decision tree learner (ID3, Quinlan, 1986).
- A relational learner (FOCL, Pazzani & Kibler, 1991).
- A probabilistic relational learner (HYDRA, Ali & Pazzani, 1993).

We have evaluated the effects of these modifications on a variety of medical data sets available at the machine learning archive at the University of California, Irvine, including diagnosis of breast cancer and the sparse clinical data provided by the workshop organizers.

2. Methods and Initial Results

Neural networks that use a logistic activation function (Rumelhart, Hinton, & Williams, 1986) have units that produce an output that ranges from 0 to 1. We can use the value of the activation on the unit with the maximum activation as a measure of the likelihood that a test example is classified correctly. If a classifier only makes predictions when this maximum value is greater than a given threshold, it will not produce a classification of every test example. We conducted a series of experiments to determine whether the accuracy improves when the classifier is allowed to refuse to classify some examples. Here, we will present results obtained used the breast cancer database, training on 2/3 of the data and testing on the remaining 1/3. Figure 1 graphs the average (N = 20) coverage and accuracy as a function of the minimum activation required of any output unit in order to make a classification. The graph clearly shows that by restricting the situations in which a classification is made, the accuracy of the classifier can be improved.

A simple Bayesian classifier (Duda & Hart, 1973; Langley, 1993) estimates the probability that a test instance is a member of each class and the test instance is assigned to the class with the highest probability. We can use the same method we used with back propagation to set a minimum threshold required to make a prediction. Figure 2 shows the accuracy and coverage of the Bayesian classifier as a function of the natural logarithm of the minimum probability required to make a prediction.

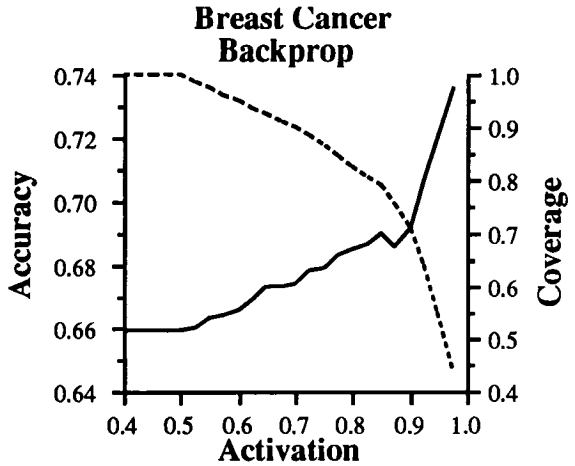

Figure 1. Accuracy (solid line) and Coverage (Dotted line) of back propagation as a function of the minimum activation required to make a prediction.

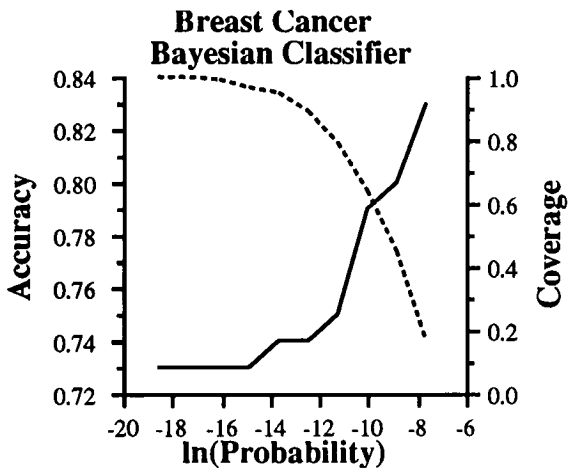

Figure 2 Accuracy (solid line) and Coverage (Dotted line) of a Bayesian classifier.

Unlike neural networks and Bayesian classifiers, ID3 does not have a continuous output that is maximized to assign a class label. Instead, a leaf in the decision tree is used to assign a class label to an example. We can use the distribution of examples at a leaf to estimate the reliability of a leaf. In particular, we use a Laplace estimate of the probability of a class given a leaf. If there are N_i examples of class i at a leaf and k classes, then the probability that an example at the leaf is a member of class i is given by:

$$p(class = i) = \frac{N_i+1}{k+\sum_j^k N_j}$$

For example, if there are 3 examples of class A, 1 example of class B, and 0 examples of class C at a leaf, the Laplace estimate of the probability that an example classified at that leaf should be assigned to class A is P(class = A) = (3 + 1)/(4+3) = 4/7. The Laplace estimate has the effect that the difference between class probabilities is decreased when there is less evidence (in the form of fewer instances at the leaf) to determine the probabilities. The decision tree classification procedure can easily be modified not to make a prediction on an example when the example is classified by a leaf whose majority class has a probability below a given threshold. Figure 3 graphs the accuracy and coverage of ID3 modified in this manner on the breast cancer data set.

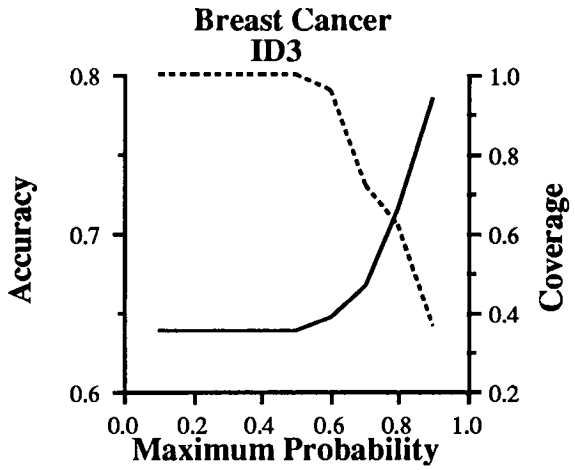

Figure 3 Accuracy (solid line) and Coverage (Dotted line) of a decision tree learner

FOCL (Kibler & Pazzani, 1991) is a relational learning program that extends FOIL (Quinlan, 1990) by including a compatible explanation-based learning system that uses information gain to guide the operationalization process. FOCL was designed to deal with binary classification problems. It learns a concept description (i.e., a set of clauses) to cover the positive examples. It uses the closed-world assumption to classify test examples. If an example satisfies any clause, it is classified as positive. Otherwise, the example is classified as negative. We have an approach to trading off coverage for accuracy in FOCL by learning multiple concept descriptions to cover the positive examples.

Learning multiple concept descriptions is related to the concept of averaging multiple models (Kwok & Carter, 1989). In particular, we get FOCL to learn a set of different concept descriptions (where each concept description is a set of clauses that cover the positive training examples). Averaging multiple models would require that a majority of the concept descriptions are satisfied to call an example positive. We can use the number of concept descriptions that are satisfied as a measure of the reliability of a classification. For example, if 11 concept descriptions are learned, an example might be considered positive if more than 7 concept descriptions are satisfied for that example. Similarly, an example might be considered negative if fewer than 4 concept descriptions are satisfied. When an intermediate number of concept descriptions are satisfied, no prediction is made.

To get FOCL to learn a variety of different concept descriptions, the approach to selecting a literal to add to a clause is modified. Normally, FOCL computes the information gain of every possible literal and chooses the one with the maximum information gain. To get FOCL to learn different concept descriptions, we modify the algorithm to save the N literals (e.g., the top 11) that have the highest information gain. Next, it stocastically selects a literal from these N literals (Kononenko & Kovacic, 1992). The probability that any literal is chosen is proportional to its information gain. For example, if one literal has information gain of 50 and another 10, then the literal with information gain of 50 would be 5 times more likely to be added to the clause under construction. Using this procedure, FOCL learns a different concept description each time it is run. Figure 4 shows the average accuracy and coverage as a function of the minimum number of concept descriptions required to be satisfied to call an example positive. [1]

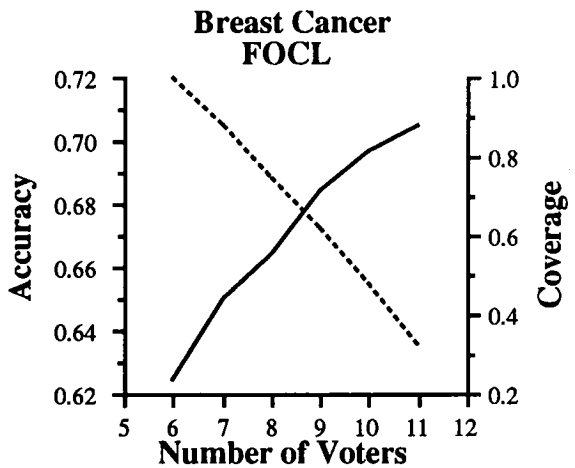

Figure 4 Accuracy (solid line) and Coverage (Dotted line) of FOCL

A variety of learning algorithms attach weights or likelihoods (e.g., CN2– Clark & Niblett, 1989) to learned rules. Here, we describe how we have modified one such system, HYDRA, to allow it to decide whether to classify a test example. HYDRA (Ali & Pazzani, 1993) is an algorithm for learning first-order Horn clause concept descriptions from noisy relational data. Unlike FOCL, HYDRA does not use the closed world assumption to classify test examples. Rather, it learns a set of clauses for each class. This is done by running an induction algorithm once for each class, treating the examples of that class as positive examples and the examples of all other classes as negative examples. HYDRA attaches a measure of the classification reliability of that clause. Currently, we use the degree of logical sufficiency (*ls*) (Duda *et al.*, 1979) that the clause has for its associated class as a measure of reliability. LS is defined as follows:

$$ls_{ij} = \frac{p(clause_{ij}(t) = true | t \in class_i)}{p(clause_{ij}(t) = true | t \notin class_i)}$$

where i denotes the class number, j denotes the clause number and t is a random variable representing an example. Thus, the concept descriptions learned by HYDRA for two classes look something like this:

$a(X,Y)$ and $b(Y,Z) \rightarrow class_1(X,Z)$	LS = 3.0
$d(X,X)$ and $e(Y,X) \rightarrow class_1(X,Z)$	LS = 4.8
$\neg a(X,Y)$ and $\neg e(W,Z) \rightarrow class_2(X,Z)$	LS = 2.1

In order to classify a test example, HYDRA examines the clauses for each class that are satisfied by that test example and chooses the clause with the highest LS value. Thus, it produces a representative clause for each class. Then, the test example is classified to the class associated with the representative clause with the highest LS value. If no clause of any class is satisfied by the test example, HYDRA guesses the most frequent class.

We can modify HYDRA to decide whether or not to classify an example by examining the ratio of the LS value of the satisfied clause with the highest LS value and the LS value of the satisfied clause of a different class with the next highest LS (or 1.0 if no clause from another class is satisfied). When this ratio is below a certain value, no prediction is made. Figure 5 graphs the average accuracy and coverage of HYDRA as a function of the natural logarithm of the minimum LS ratio required to make a prediction.

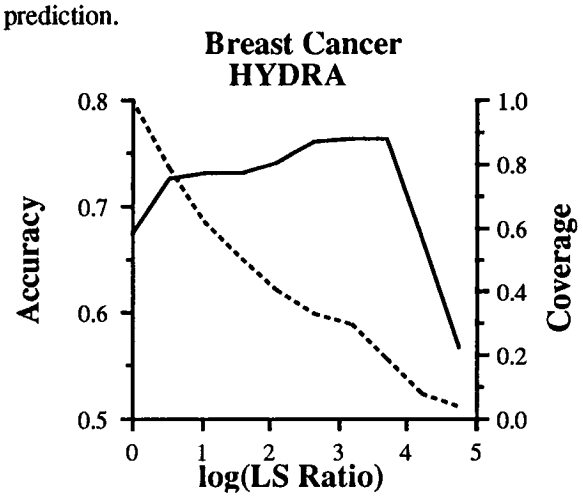

Figure 5 Accuracy (solid line) and Coverage (Dotted line) of HYDRA

3. Applications to the diabetes data sets

The most common problems addressed by machine learning systems are classification problems in which each example is described by a fixed set of attributes[2] and each

1. The maximum number of concepts descriptions that may be satisfied to classify an example as negative is given by 11 minus the minimum number of concept descriptions required to be satisfied to call an example positive.

2. Although most approaches can deal with missing attribute values, they typically assume that only a small fraction of the attributes are missing.

example belongs to one of a small number of categories. The diabetes patient data does not meet either of these criteria. Rather, there are a series of time-stamped events that describe either the patient's blood glucose measurements, or events (such as meals, exercise, or insulin) that may influence the blood glucose measurement. To apply classifier learning methods to this database, we recoded the data into a form in which there was a class to be predicted and a fixed set of attributes.

We decided to create a classification problem in which the value to be predicted was the blood glucose measurement. The mean blood glucose measurement of each patient was determined and the goal was to predict, for each measurement, whether the value was above or below the patient's mean. This was chosen for practical reasons, including our lack of medical expertise. A two class problem insures that all of the learning algorithms we have tested will be able to run on the problem. In contrast, if we had chosen to predict a numeric quantity (and wanted to investigate trading off coverage for accuracy), none of the algorithms would have been directly applicable.

Once the decision of the class to be predicted was made, we were faced with the task of deciding on the attributes and values used to describe each example. Some of the attributes describe the time and type of the glucose measurement (e.g., pre-breakfast). For the remaining attributes, we used a windowing approach in which the last "significant" event of each type is recorded, together with information on the time elapsed between the occurrence of that event and the current blood glucose measurement. For significant events, we chose the last regular insulin dose, the last NPH insulin dose, and the last glucose measurement. These significant events were chosen because they commonly occur in many of the larger patient files, and they are recorded frequently.

To summarize, each example is described by 12 attributes. The first attribute is the one that is to be predicted:

- Current glucose measurement: (above or below).
- CGT: Current glucose time: (in hours) numeric.
- CGM: Current glucose meal: (unspecified, breakfast, lunch, super, or snack).
- CGP: Current glucose period: (unspecified, pre, or post).
- LGV: Last glucose measurement: numeric.
- ELGV: Elapsed time since last glucose measurement: (in hours) numeric.
- LGM: Last glucose meal: (unspecified, breakfast, lunch, super, or snack).
- LGP: Last glucose period: (unspecified, pre, or post).
- ENPH: Elapsed time since last NPH insulin: (in hours) numeric.
- NPH: Last NPH dose: numeric.
- EREG: Elapsed time since last regular insulin: (in hours) numeric.
- LREG: Last regular dose: numeric.

It is not obvious to the authors that the last 11 attributes provide enough information to predict reliably the first attribute. All of the algorithms we have tried on data converted to this format do better than chance at making the prediction. However, none of the algorithms approach 100% accuracy.

We concentrated our experimental work on two patients with a large amount of data that was frequently collected (data-20 and data-27). For each of the data sets, over 450 examples were constructed. We ran three algorithms, (the neural network trained with back propagation using 10 hidden units), the decision tree learner (ID3), and the rule learner (FOCL) on 300 examples, and measured the accuracy and coverage of the algorithms on 150 examples (averaged over 20 trials). Figures 6, 7 and 8 show the results for back propagation, ID3, and FOCL, respectively, for data-27. Table 1 shows an example of a rule learned by FOCL on this problem. The results with data 20 are similar.

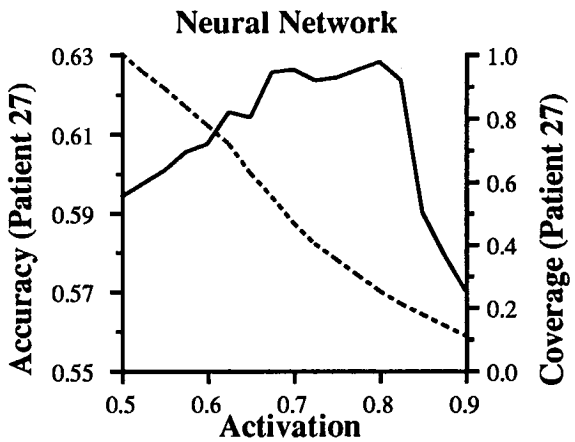

Figure 6 Accuracy (solid line) and Coverage (Dotted line) of back propagation on data-27.

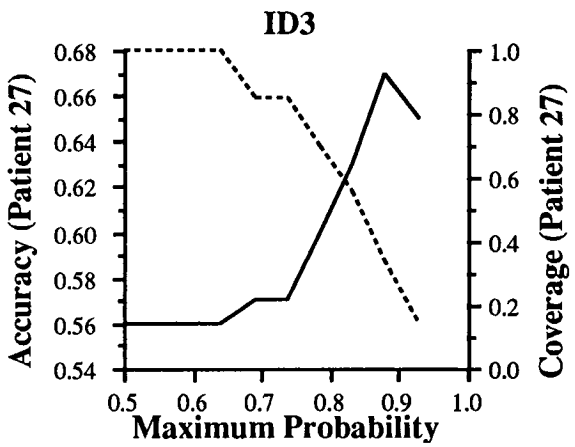

Figure 7 Accuracy (solid line) and Coverage (Dotted line) of ID3 on data-27.

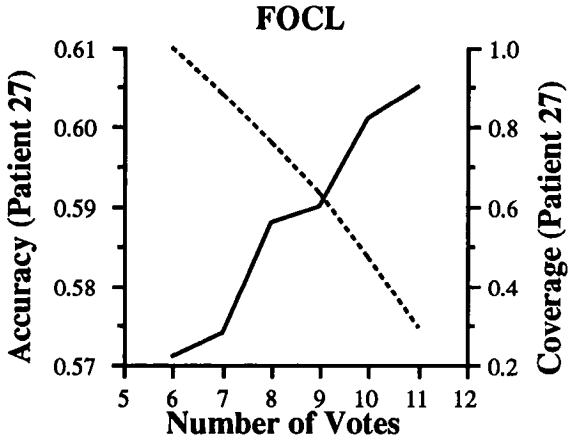

Figure 8 Accuracy and Coverage of FOCL.

4. Conclusions

This paper presents a purely empirical study. We have shown that a number of learning algorithms can be modified to deal with the situation where the user does not need the classifier to classify all examples but needs higher accuracy when a classification is made. Experimental results verified that a trade off between coverage and accuracy can be achieved. People are faster and more accurate at classifying typical instances than atypical ones (Smith & Medin, 1981) and we believe the modifications we have made to the various algorithms tend to make predictions only on the more typical examples.

We have applied the learning algorithm to a variety of data sets in experiments not reported here and have consistently found that the accuracy of the classifier increases when effective methods are found to estimate the reliability of predictions made by the classifier. Although we believe the techniques are general purpose, we also believe that each application will have to decide how to trade off accuracy for coverage.

```
ABOVE if ENPH ≥ 12.0 & LGV ≥ 131 & ENPH < 24.0
      & CGM ≥ SUPPER
ABOVE if LGV ≥ 132 & LREG < 6.5 & CGT ≥ 23.0
ABOVE if ELGV ≥ 6.5 & LGV < 130 & LGV ≥ 121
ABOVE if ELGV < 56.0 & LGV < 83 & ENPH ≥ 24.0
ABOVE if LGV ≥ 163 & CGP = UNSPECIFIED
      &  LGV < 181
ABOVE if LGV ≥ 131 & LGV < 147 & CGM = LUNCH
ABOVE if ENPH ≥ 12.0 & LGV ≥ 131 & CGT ≥ 8.0
      & LGV < 142
ABOVE if ELGV ≥ 6.5 & LGV ≥ 191 & ELGV < 10.5
ABOVE if ELGV ≥ 6.5 &  CGT ≥ 8.0 & LGV < 90
ABOVE if LGV ≥ 96 & LGV < 118 & ELGV ≥ 4.5
      & ENPH ≥ 14.5
ABOVE if LGV ≥ 128 & ENPH ≥ 5.0 & ELGV < 5.5
      & LGV < 147
ABOVE if ENPH ≥ 5.0 & LGV ≥ 189 & ELGV < 4.0
ABOVE if LREG ≥ 7.5 & ENPH < 11.5 & LGV < 147
ABOVE if LGV ≥ 128 &  ELGV ≥ 33.5 & CGT < 7.5
```

Table 1. An example of the rules learned by FOCL from 200 examples of blood glucose measurement from data-27.

We consider the results of applying the algorithms to the diabetes data to be at best only partially successful. With more medical knowledge, we might have been able to define a better classification problem and a better set of attributes. However, we do not believe the ultimate solution to analyzing clinical data is to force the problem to be a classification problem. Rather, precise definitions of the desired outputs and available inputs may lead to the identification of a new task and should lead to the creation of new learning algorithms.

Acknowledgements

The research reported here was supported in part by NSF Grant INT-9201842, ARPA Grant F49620-92-J-0430, and AFOSR AASERT grant F49620-93-1-0569.

Ali, K. & Pazzani, M. (1993). *HYDRA: A noise-tolerant relational concept learning algorithm.* The International Joint Conference on Artificial Intelligence, Chambery, France.

Clark, P. & Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning, 3*, 261-284.

Duda, R. & Hart, P. (1973). *Pattern classification and scene analysis.* New York: John Wiley & Sons.

Duda, R., Gaschnig, J. & Hart, P. (1979). Model design in the Prospector consultant system for mineral exploration. In D. Michie (Ed.), *Expert Systems in the Micro-electronic Age.* Edinburgh, UK. Edinburgh University Press.

Kononenko, I & Kovacic, M. (1992). Learning as optimization: stochastic generation of multiple knowledge. In *Proceedings of the Ninth International Workshop on Machine Learning (ML92).* Aberdeen, Scotland. Morgan Kaufmann.

Kwok, S. & Carter, C. (1989). Multiple decision trees. *Uncertainty in Artificial Intelligence, 4*, 327-335.

Langley, P. (1993). *Induction of Recursive Bayesian Classifiers.* ECML-93. Vienna, Austria.

Pazzani, M. & Kibler, D. (1991). The utility of knowledge in inductive learning. *Machine Learning, 9, 1*, 57-94.

Quinlan, J.R. (1986). Induction of decision trees. *Machine Learning, 1*, 81–106.

Quinlan, J.R. (1990). Learning logical definitions from relations. *Machine Learning, 5*, 239–266.

Rumelhart, D., Hinton, G., & Williams, R. (1986). Learning internal representations by error propagation. In D. Rumelhart and J. McClelland (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations,* (pp 318-362). Cambridge, MA: MIT Press.

Smith, E. & Medin, D. (1981). *Categories and Concepts.* Cambridge, MA. Harvard University Press.