Recognizing Reliability of Discovered Knowledge*

Petr Berka

Laboratory of Intelligent Systems, Prague University of Economics,
Ekonomická 957, 148 00 Prague, CR
phone: (+420 2) 24095465,
fax: (+420 2) 7934749
E-mail: berka@vse.cz

Abstract. When using discovered knowledge for decision making (e.g. classification in the case of machine learning), the question of reliability becomes very important. Unlike global view on the algorithms (evaluation of overall accuracy on some testing data) or unlike multistrategy learning (voting of more classifiers), we propose "local" evaluation for each example using one classifier. The basic idea is to learn to classify the correct decisions made by the classifier. This is done by creating new class attribute "match" and by running the learning algorithm on the same input attributes. We call this (second) step verification.
Some first preliminary experimental results of this method used with C4.5 and CN4 are reported. These results show that: (1) if the classification accuracy is very high, it makes no sence to perform the verification step (since the verification step will create only the majority rule), (2) in multiple-class and/or noisy domains the verification accuracy can be significantly higher then the classification accuracy.

1 Introduction

The declared goal of automated knowledge acquisition from data (KDD) is to obtain *nontrivial of implicit, previously unknown, and potentially useful information from data* [9]. But in many situations, our motivation is not only to possess new knowledge but also to use it for decision making. In this situations, the realibility of obtained knowledge becomes very important.

Problem solvers (classifiers) built using machine learning (ML) techniques are usually evaluated in the terms of overall performance (accuracy, error rate) on given data sets. The training data are used in the knowledge discovery (learning) step and testing data are used to obtain some global characteristics of the classifier. It is a question, if this is satisfactory to evaluate the reliability of the class assignment of a single example.

Recently, increasing attention is in the ML community payed to the global behaviour of classifiers. So e.g. Domingos and Pazzani studied the naive bayesian

* Partially supported by the grant MSMT no. VS96008.

classifiers [5], or Kohavi and Wolpert studied the missclassifiacion rates of classifiers in general [7]. Some research was done also on looking for the best classifier, either by running more classifiers on given data [12] or by using some "meta-knowledge" which can help to select the best classifier according to some characteristics of the data. Let us mention here e.g. the Machine Learning Toolbox [6] or work related to the STATLOG project [11, 2]. Nevertheless, the method of voting of more classifiers when classifying single example remains the only used method which deals with some kind of local evaluation of classification results.

In human decision making, the expert usually accompanies her decisions by some statements about the realibility of the decision. So she may say e.g. "the class is x, this was an easy problem" (because for similar cases, she was always right), or "it's hard to say, I think that the class is y" (because for similar cases, her decision were sometimes wrong). Such insight can help the user (client) to better understand and accept the expert's decision.

In this paper we propose a simple method, which in some sense evaluates the reliability of classification of one particular example. We try to learn similar "metaknowledge" as shown in expert's explanation above by learning to recognize correctly classified examples.

2 Method

We propose a general method for evaluation of the reliability of classification, which can be easily incorporated in any ML algorithm. The scheme of the method is shown in Fig. 1.

The *classification step* is performed in the usual way. We use training data to learn the knowledge and testing data to test it (to obtain the standard evaluation of the knowledge in terms of classification accuracy). During classifiaction of both training and testing data, we obtain for each example the predicted class value. By comparing this predicted value with the real one found in data (we assume supervised learning), we can introduce new (binary!) attribute "match". The value of this attribute is "yes" if the classification was correct, or "no" if the example was misclassified.

We use this new attribute as the class attribute in the subsequent *verification step*. Here we run the machine learning algorithm on the same training and testing data for this new class. The resulting verification accuracy can be interpreted as "reliability" of the "classification knowledge" learned in the first step.

Our expectations about the behaviour of the proposed method can be formulated as following hypotheses:

H1: If the classification accuracy is very high, it makes no sence to perform the verification step.

H2: In multiple-class domains and/or in noisy domains the verification accuracy can be significantly higher then the classification accuracy.

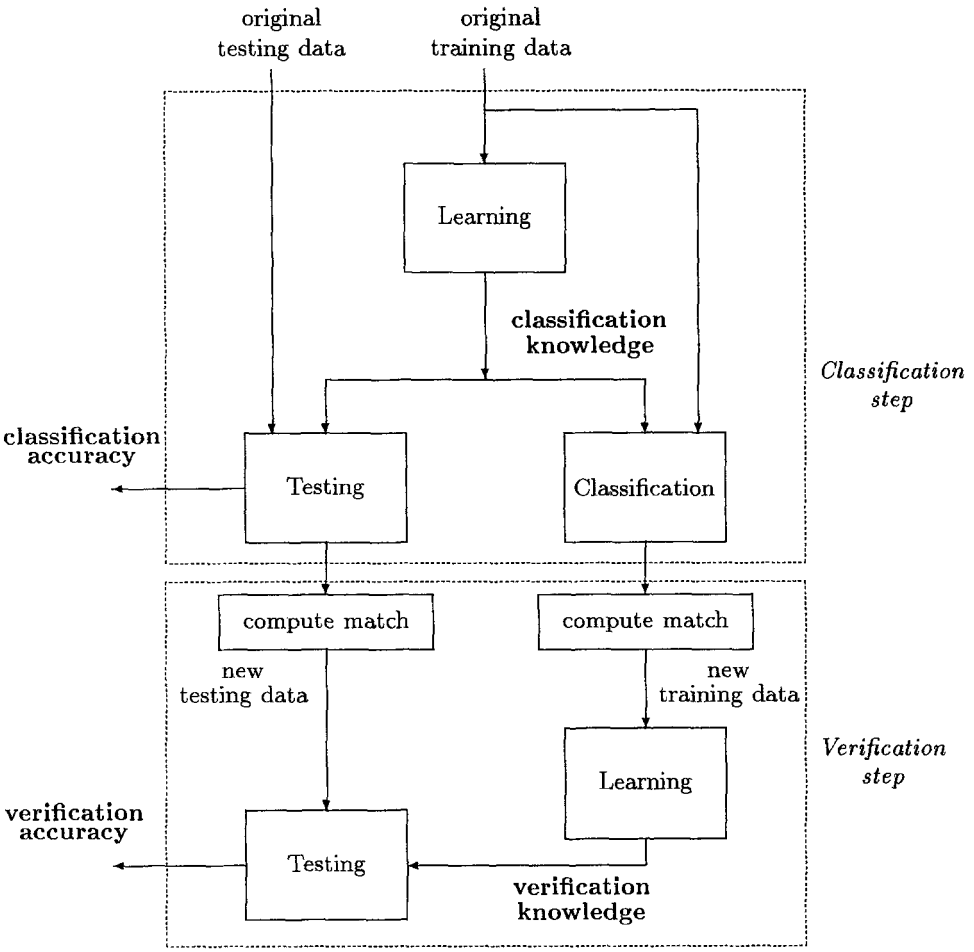

Fig. 1. Scheme of the method

H3: If the classifier gives not only the (yes/no) class assignment, but also some additional information (probability as in Bayesian classifier, or weight as in KEX [1]), this information can be significant for the verification.

We will further concentrate on the first two hypotheses. If the classification accuracy is very high, the verification knowledge will consist only of default rule "everything is fine". In this case, the verification accuracy is exact the same as the classification accuracy and the verification step brings no benefit.

If the classifiaction accuracy is low (this can be for the multiple-class or noisy domains), there is a chance (because we transformed multiple-class problem into binary-class problem) that the verification accuracy will be higher. Here, the verification step can be important.

If the classification accuracy is higher than verification accuracy, we can forget about the verification step.

Both "classification knowledge" and "verification knowledge" should be used in processing of new, unseen examples. We classify the example and verify the "correctness" of this classification. If the verification accuracy is high (as shown before, we can assume, that verification accuracy is at least equal to classification accuracy), we can more believe that the classification of an example is correct, iff the verification confirmes this decision.

3 Experiments

We tested the proposed method using Quinlan's C4.5 [10] and Bruha's CN4 [3], a large extension of Clark's and Nibblet's CN2 [4]. The first system creates decision trees, the second system creates decision rules (in unordered mode) and decision lists (in ordered mode). To perform our experiments, we modified both systems to:

(1) perform batch consultation from a file, and
(2) create the attribute `match`.

We used some data from the UCI repository [8]. The results for C4.5 (for pruned trees on testing data) are summarised in the Table. 1. For each data set we show (for both classification and verification step) the size of the pruned tree, the percentage of errors on testing data, and the estimate of the percentage of errors on unseen data.

	Classification			Verification		
data	size	Errors	Estimate	size	Errors	Estimate
1 Monk 1	14	35.2%	35.7%	1	35.2%	24.0%
2 Monk 2	31	59.2%	35.1%	52	45.1%	21.5%
3 Monk 3	12	6.1%	23.8%	1	6.1%	10.5%
4 Heart	28	13.1%	26.0%	5	12.2%	14.3%
5 Austral. credit	12	14.6%	16.9%	1	14.6%	9.6%
6 Indian diabetes	27	23.2%	30.6%	131	27.3%	22.6%
7 Soya	72	18.9%	26.6%	16	15.7%	11.7%
8 LED 0% noise	19	0.0%	10.5%	1	0.0%	1.1%
9 LED 10% noise	41	24.0%	41.7%	35	18.0%	20.0%
10 LED 20% noise	67	39.0%	60.1%	61	24.3%	27.7%
11 LED 30% noise	117	47.0%	69.9%	67	24.7%	25.4%

Table 1. Results of experiments for C4.5

The results for CN4 are summarised in the Table. 2. We show for both ordered and unordered mode the number of rules and the accuracy on testing data.

	Ordered mode				Unordered mode			
	Classification		Verification		Classification		Verification	
data	size	Accuracy	size	Accuracy	size	Accuracy	size	Accuracy
1 Monk 1	8	100.0%	1	100.0%	24	100.0%	1	100.0%
2 Monk 2	43	69.7%	10	69.7%	63	53.5%	16	53.5%
3 Monk 3	19	90.0%	1	90.0%	24	85.2%	1	85.2%
4 Heart	20	65.9%	1	65.9%	35	53.3%	1	53.3%
5 Austral. credit	56	76.9%	10	76.9%	53	80.8%	22	80.8%
6 Indian diabetes	85	71.1%	70	70.1%	104	72.7%	53	72.2%
7 Soya	43	70.2%	3	71.0%	33	71.8%	26	72.6%
8 LED 0% noise	10	100.0%	1	100.0%	11	100.0%	1	100.0%
9 LED 10% noise	25	81.3%	19	87.7%	37	82.7%	23	82.7%
10 LED 20% noise	35	68.3%	17	77.0%	52	74.7%	17	74.7%
11 LED 30% noise	39	62.7%	24	75.3%	56	71.0%	6	72.0%

Table 2. Results of experiments for CN4

We run both systems with default parameters, only in some cases, we changed the pruning confidence level for C4.5.

The verification step very often results in the default rule (of size 1); this gives the same verification accuracy as the classifiaction accuracy. In all cases (with exception of C4.5 results for the Pima indian diabetes data) the knowledge obtained in the verification step was significantly smaller then the knowledge obtained in the classification step. In the case of the Pima indian diabetes data, the verification accuracy was lower then the classification accuracy for both systems (so some overfitting occured in this case). On the contrary, best results (greater verification accuracy then classification accuracy) were obtained for the noisy LED domains. For C4.5, the estimated number of errors on unseen data was significantly lower for the verification step for every dataset.

The plot of verification accuracy on testing data vs. classification accuracy on testing data for C4.5 is shown in Fig. 2. The numbers used in the graph correspond to the numbers of datasets in the Table. 1. Same graf for CN4 in ordered mode is shown in Fig. 3. Since for CN4 in unordered mode the verification accuracy does not significantly differ from the classification accuracy for all used datasets, we don't plot the results.

4 Conclusions and further work

The paper describes a simple method for evaluation of the reliability of classification result for a particular example. We didn't overcome the problem with using "global" characteristics (overall accuracy) on a single example. We only transformed the standard task of learning classes into the task of learning correct decisions.

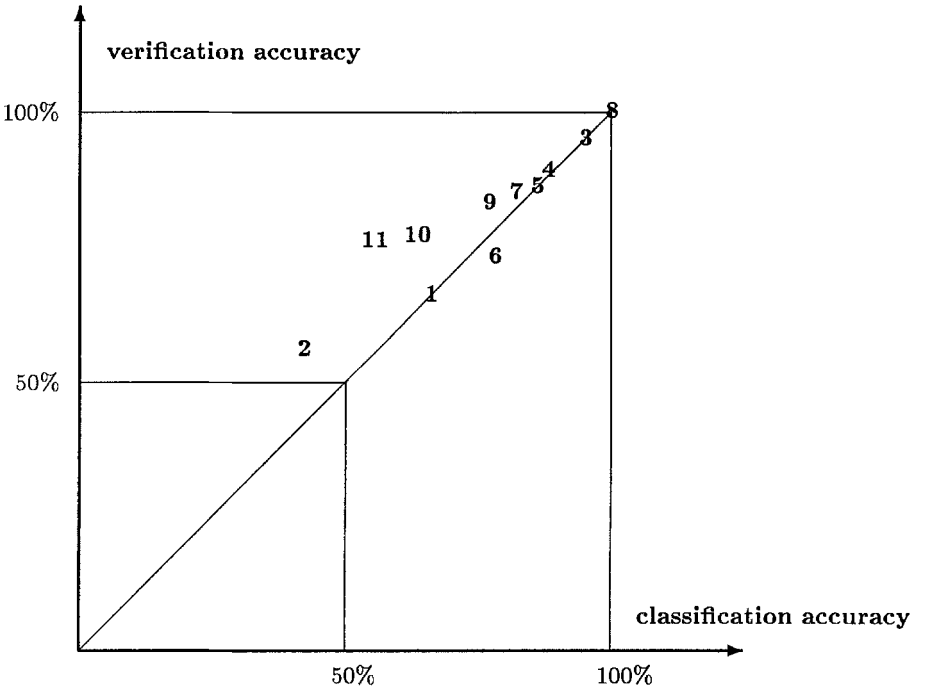

Fig. 2. Verification vs. classification accuracy for C4.5

Our preliminary results show, that the verification step often brings no better results than the classification step. Sometimes, the results of verification step can be even worse (see the Indian diabetes data). Nevertheless, since the verification accuracy can be taken as at least the same as the classification accuracy, we can make (in more cases) more reliable statements about the class assignment done by the classifier. Ofcourse, more experiments should be done with different ML algorithms.

What was not done yet are experiments related with the hypothesis H3. One straightforward way how to use the probability (or weight) is not to classify examples, where the probability (weight) is "near the border" (e.g. probability is near 0.5 for binary class domains). The relationship between probability (or weight) of assigned class and the "correctness" of the classification should be studied in our next work.

We feel, that no general conlusions can be made on the basis of our experiments. But we think, that the topic touched in our paper has a great practical importance (if the ML classifiers should be used to **really** classify new examples) and that it is worth further study.

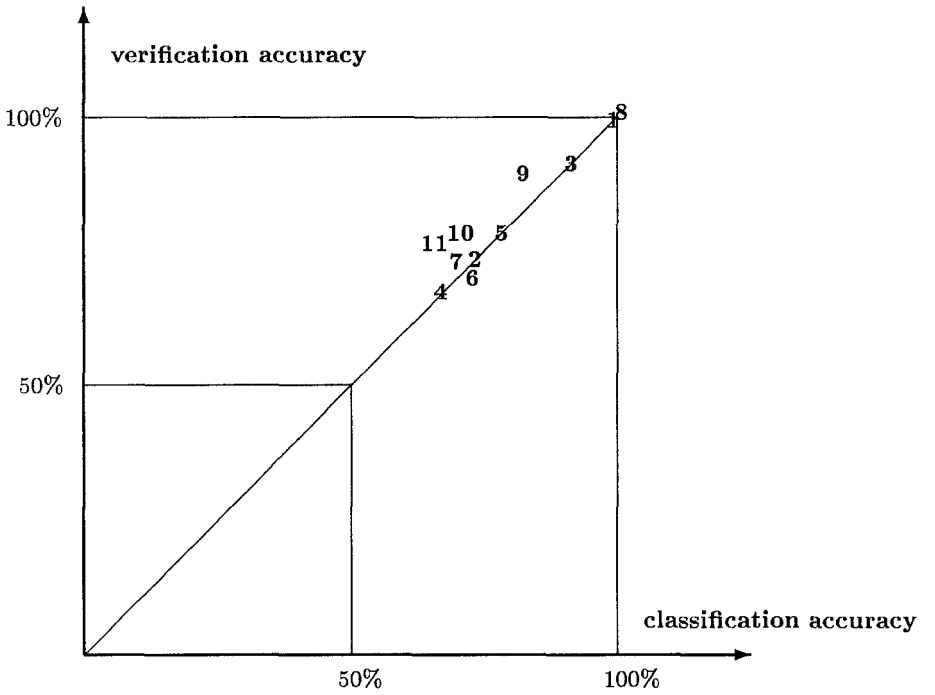

Fig. 3. Verification vs. classification accuracy for CN4, ordered mode

Acknowledgments

We'd like to thank Ivan Bruha, University of Hamilton, Canada for the possibility to use and modify his system CN4.

References

1. Berka,P., Iványek,J.: Automated Knowledge Acquisition for PROSPECTOR-like Expert Systems. In: (Bergadano, deRaedt eds.) Proc. ECML'94, Springer 1994, 339-342.
2. Brazdil,P., Gama,J., Henery,B.: Characterizing the Applicability of Classification Algorithms Using Meta-Level Learning. In: (Bergadano, deRaedt eds.) Proc. ECML'94, Springer 1994, 83-102.
3. Bruha,I., Kočková,S.: A covering learning algorithm for cost-sensitive and noisy environments. In: Proc. of ECML'93 Workshop on Learning Robots, 1993.
4. Clark,P., Niblett,T: The CN2 induction algorithm. Machine Learning 3 (1989), 261-283.
5. Domingos,P., Pazzani,M.: Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. In: (Saitta ed.) Proc. ICML'96, Morgan Kaufman, 1996, 105-112.

6. Kodratoff,Y., Sleeman,D., Uszynski,M., Causse,K., Craw,S.: Building a Machine Learning Toolbox. In: (Steels, Lepape eds.) Enhancing the Knowledge-Engineering Process - Contributions from Esprit. Elsevier, 1992, 81-108.
7. Kohavi,R., Wolpert,D.H.: Bias plus Variance Decomposition for Zero-One Loss Functions. In: (Saitta ed.) Proc. ICML'96, Morgan Kaufman, 1996, 275-283.
8. Murphy,P.M., Aha,D.W.: UCI Repository of Machine Learning Databases. Irvine, University of California, Dept. of Information and Computer Science.
9. Piatetsky-Shapiro,G., Matheus,C., Smyth,P.: KDD-93: Progress and Challenges in Knowledge Discovery in Databases, 1993.
10. Quinlan,J.R.: C4.5: Programs for Machine Learning. Morgan Kaufman, 1993.
11. Taylor,C., Michie,D., Spiegelhalter,D.: Machine Learning, Neural and Statistical Classification. Ellis Horwood, 1994.
12. Tsumoto,S., Tanaka,H.: Automated selection of Rule Induction Methods based on Recursive Iteration of Resampling Methods and Multiple Statistical Testing. In: Proc. KDD-95, 1995, 312-317.