Resolving rule conflicts with Double induction

Tony Lindgren and Henrik Boström

Department of Computer and Systems Sciences,
Stockholm University and Royal Institute of Technology,
Forum 100,
164 40 Kista, Sweden
tony,henke@dsv.su.se
http://www.dsv.su.se

Abstract. When applying an unordered set of classification rules, the rules may assign more than one class to a particular example. Previous methods of resolving such conflicts between rules include using the most frequent class in the conflicting rules (as done in CN2) and to use Naive Bayes to calculate the most probable class. An alternative way of solving this problem is presented: by generating new rules from the examples covered by the conflicting rules. These newly induced rules are then used for classification. Experiments on a number of domains show that this method significantly improves the accuracy compared to the CN2 approach and Naive Bayes.

1 Introduction

Two major induction strategies are used in top-down rule induction: Divide-and-Conquer [9] or Separate-and-Conquer (SAC) [5]. The former strategy does not generate overlapping rules and hence no conflict between rules can occur, while the latter strategy can give rise to overlapping rules. The rules can be ordered to avoid conflicts (which results in decision lists) [10] or they may be used without any ordering [3, 2]. In the latter case one has to deal with conflicts that occur when two or more rules cover the same example but assign different classes to it.

This work addresses the problem of handling conflicting rules and how to solve the conflicts in a more effective way than using existing methods. Previous methods that deals with conflicts among rules either returns the most frequent class covered by the conflicting rules (as done in CN2 [2]) here referred to as *Frequency-based classification*, or uses Naive Bayes [8] to calculate the most probable class.

We propose another way of resolving such conflicts: by applying rule induction on the examples covered by the conflicting rules, in order to generate a new set of rules that can be used instead of the conflicting rules. The motivation for this idea is that by focusing on the examples that lie in the region of interest (i.e. where the example to be classified resides), one is likely to obtain rules that better separates classes in this region, than when has to consider all

examples, since examples outside this region may dominate the rule induction process, making the separation of classes in this region of marginal importance. This novel method is called *Double induction*.

The paper is organized as follows. In section 2, we first review Frequency-based classification and Naive Bayes to resolve rule conflicts, and then describe Double induction. In section 3, the three different methods are compared empirically and the results are presented. Finally, in section 4, we discuss the results and give pointers to future work.

2 Ways of Resolving Classification Conflicts

In this section, we first recall two previous methods for resolving classification conflicts among overlapping rules and then introduce the novel method, Double induction.

2.1 Frequency-based Classification

The system CN2 [2] resolves classification conflicts between rules in the following way. Given the examples in Fig. 1, the class frequencies of the rules that covers the example to be classified (marked with '?') is calculated:

$$C(+) = covers(R_1, +) + covers(R_2, +) + covers(R_3, +) = 32$$

and

$$C(-) = covers(R_1, -) + covers(R_2, -) + covers(R_3, -) = 33$$

where $covers(R_n, C)$ gives the number of examples of class C that are covered by Rule R_n. This means that CN2 would classify the example as belonging to the negative class (-). More generally:

$$FrequencyBasedClassification = argmax_{Class_i \in Classes} \sum_{j=1}^{|CovRules|} covers(R_j, C_i)$$

where $CovRules$ is the set of rules that cover the example to be classified, and $covers$ is the function defined above.

2.2 Naive Bayes classification

Bayes theorem is as follows:

$$P(C|R_1 \wedge \ldots \wedge R_n) = P(C)\frac{R_1 \wedge \ldots \wedge R_n|C)}{P(R_1 \wedge \ldots \wedge R_n)}$$

where C is a class label for the example to be classified and $R_1 \wedge \ldots \wedge R_n$ is the rules that cover the example. As usual, since $P(R_1 \wedge \ldots \wedge R_n)$ does not affect

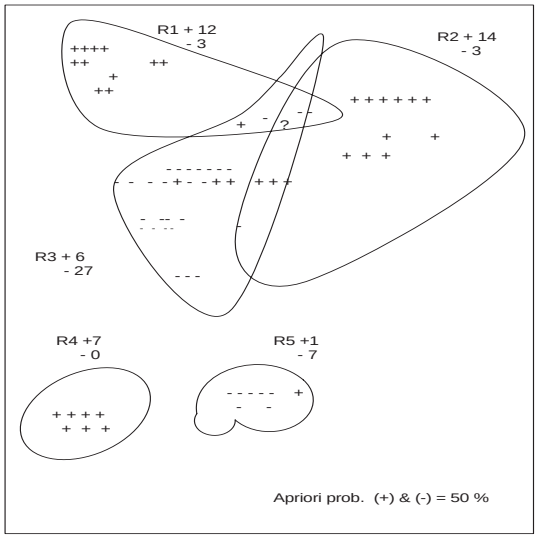

Fig. 1. Three rules covering an example to be classified (marked with ?). The training examples are labeled with their respective classes (+ and -).

the relative order of different hypotheses according to probability, it is ignored. Assuming (naively) that $P(R_1 \wedge \ldots \wedge R_n|C) = P(R_1|C) * \ldots * P(R_n|C)$. This is the difference between Bayes rule and the Naive Bayes classifier, e.g. that the latter assumes that each pice of evidence is conditionally independent in relation to other evidence (rules) given the hypothesis (classes), whereas the former does not. The maximum a posteriori probable hypothesis (MAP) for Naive Bayes is:

$$h_{MAP} = argmax_{Class_i \in Classes} P(Class_i) \prod_{Rj \in Rules}^{|Rules|} P(R_j|Class_i)$$

where *Rules* is the set of rules that cover the example to be classified.

Note that if a rule does not cover any examples of a class this would eliminate that class, as the probability would be zero. To avoid this we use Laplace-m correction described in [6].

If we again consider the example shown in Fig. 1 we get:

$$P(+|R_1 \wedge R_2 \wedge R_3) = P(+) * P(R_1|+) * P(R_2|+) * P(R_3|+) =$$
$$40/80 * 12/40 * 14/40 * 6/40 = 0.0079$$

$$P(-|R_1 \wedge R_2 \wedge R_3) = P(-) * P(R_1|-) * P(R_2|-) * P(R_3|-) =$$
$$40/80 * 3/40 * 3/40 * 27/40 = 0.0019$$

This means that Naive Bayes classification results in that the example with unknown class label is classified as belonging to the positive (+) class.

2.3 Double induction

The idea of Double induction is to induce unordered rules based on the examples that are covered by the rules in conflict. By doing this we obtain a completely fresh set of rules that are tailor made to separate the examples in this subset of the whole domain. By concentrating on a small subspace of the example space there is a higher chance to find rules that separate the classes better. The training set is divided in two halves at random to create a grow set (to induce rules) and a prune set (to prune rules). This division is likely to further reduce the chance that we will find the same rules as before. The Double induction algorithm is given in Fig. 2.

Input: R = rules from the first induction round, e = example to be classified,
E = training examples

collect all rules, r ∈ R that cover e, in R_e
if conflict(R_e)
  collect all training examples covered by R_e in R_E
  induce new rules with using R_E as input and get Rules as output, SAC(R_E, Rules)
  collect all rules, r_{double} ∈ Rules that cover e, in $Rules_e$
  if conflict($Rules_e$)
    resolve the conflict with naive bayes, naiveBayes($Rules_e$, Class)
    classify e with Class
  get the majority class of $Rules_e$, majority($Rules_e$, Class)
  classify e with Class
get the majority class of R_e, majority(R_e, Class)
classify e with Class

Fig. 2. Pseudo code for Double induction algorithm.

It is worth noting is that when solving conflicts with Naive Bayes on the newly induced rules, the apriori probability is computed from the examples covered by the previously conflicting rules. This means that the apriori probability reflects the probability distribution of this subspace (contained by the rules in conflict), rather than the apriori probability of the whole doamain.

Consider again the the scenario shown in Fig. 1. Given the examples covered by R_1, R_2 and R_3, Separate-and-Conquer, may come up with the four new rules shown Fig. 3. The unlabeled example is then classified using these newly induced rules.

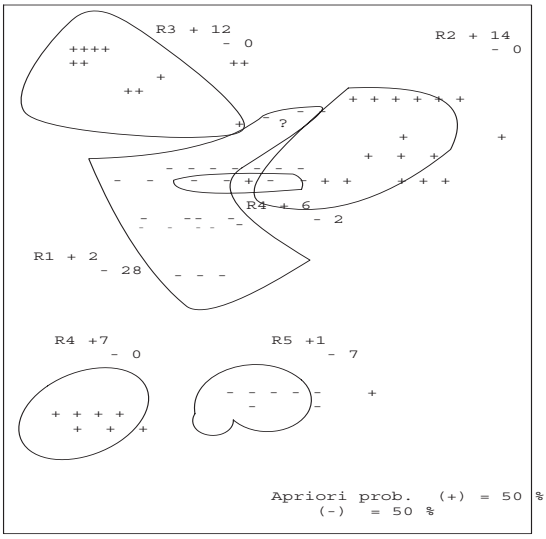

Fig. 3. Four rules covering an unlabeled example (marked with ?). The training examples are labeled with their respective classes (+ and -).

In our hypothetical scenario, the example is covered by R_1 resulting in that the negative class is assiged to the example.

In our experiments we use two different sets of examples as input in the second rule induction round: one which have no duplicates of examples and one with the concatenation of all the covered examples of every rule (a multi-set), in which some examples may be present more than once. The latter is in a sence a weighting scheme of the examples in the example set. One reason for using such a weighting scheme is that it has been empirically demonstrated that the examples in the intersection is more important (and hence should have more weight) than other examples when doing the classification, see [7]. Note that this type of weighting is implicitly used in both Frequency-based classification and Naive Bayes.

3 Empirical Evaluation

Double induction has been implemented in the Rule Discovery System (RDS) [1], which is a rule based machine learning system that supports various rule induction techniques, e.g., separate-and-conquer, divide-and-conquer, and ensemble methods like bagging and boosting.

Double induction has been implemented both with and without weighting of the examples according to the coverage of previous learned rules. Both of these variants are compared to Frequency-Based classification and Naive Bayes.

3.1 Experimental Setting

The employed search strategy was Separate-and-Conquer together with information gain to greedily choose what condition to add when refining a rule. One half of the training set was used as grow set and the second half was used as pruning set. The rules were pruned using Incremental reduced error pruning (IREP) [4]. These settings was used in both induction rounds, i.e. both when inducing the initial set of rules and when resolving rule conflicts.

The experiments were performed using 10 fold cross-validation.

All datasets used were taken from the UCI Machine Learning Repository except the King-Rook-King-Illegal (KRKI) database which comes from the Machine Learning group at the University of York. In Table 1, the domains used in the experiment are shown, as well as their main characteristics.

Table 1. The domains used in the experiment

Domain	Classes	Class distribution	Examples
The Glass	2	24.1, 75.9	112
Balance Scale	3	8, 46, 46	625
Breast Cancer	2	29.7, 70.3	286
Liver-disorders	2	42, 58	345
Car Evaluation	4	4, 4, 22, 70	1728
Dermatology	6	5.5, 13.3, 14.2, 16.7, 19.7, 30.6	366
Congressional Voting	2	45.2, 54.8	435
Ionosphere	2	36, 64	351
Lymphography	4	1.4, 2.7, 41.2, 54.7	148
New thyroid	3	69.8, 16.3, 13.9	215
Primary tumor	22	24.8, 5.9, 2.7, 4.1, 11.5, 0.3, 4.1, 1.8, 0, 0.6, 8.3, 4.7, 2.1, 7.1, 0.6, 0.3, 2.9, 8.6, 1.8, 0.6, 0.3, 7.1	339
Sonar	2	53, 47	208
Nursery	5	33.3, 0.0, 2.5, 32.9, 31.2	12960
Shuttle Landing Control	2	47.8, 52.2	278
KRKI	2	34, 66	1000

3.2 Experimental Results

The rule conflict resolution methods were tested on fifteen domains. In four of these domains, the rules learned were without conflict on test examples. These domains was: Glass, Liver-disorders, Ionosphere and Sonar.

The results from the domains with rule conflicts are shown in Table 3, where the result for each domain has been obtained by ten-fold cross-validation. Exactly the same folds and generated rules are used by the four classification methods.

In Table 2, the first column shows the domain used. The second column gives the accuracy of Double induction with weighted examples. The third column gives the accuracy of Double induction with no weighting. The fourth column shows the accuracy of Frequency-based classification. The fifth column shows the accuracy of Naive Bayes. The sixth column shows the upper-bound of how much accuracy can be improved, i.e. if we have a upper-bound of 30 percent then the accuracy can be improved by at most 30 percent. Note that this reflects the amount of conflicts between the rules induced.

Table 3 gives some numbers that is used for the statistical test. Inside the parentheses, the number of domains in which the method in the leftmost column has a higher respectively lower accuracy compared to the method in the upper row (wins, losses). The value after each parenthesis shows the p-value according to an exact version of McNemar's test. An asterisk (*) is used to signal that the result is statistically significant, using the threshold $p < 0.05$.

It's worth noting that Double induction with weighting is significantly more accurate than both Naive Bayes and Frequency-based classification. It is not significantly more accurate than Double induction without weighting, but still it is better (6 wins and 3 losses).

Double induction without weighting performs significantly better than Frequency-based classification, but not significantly better than Naive Bayes.

There is no significant difference in accuracy between Naive Bayes and Frequency-based classification .

Table 2. Intersecting rules compared with Naive Bayes and Intersecting rules compared with Frequency based classification

Domain	D. ind. w weight	D. ind.	Frequency-b.	Naive B.	Conflicts
Balance scale	83.30	82.43	81.20	82.78	50.8
Breast cancer	75.00	74.60	73.81	75.00	21.4
Car	86.82	87.21	78.04	77.59	34.1
Dermatology	91.69	91.69	91.69	91.08	5.2
C. Votes	95.20	94.95	94.44	93.94	6.6
Lymphography	80.00	80.00	79.26	80.00	26
New thyroid	85.71	87.24	85.71	85.71	5.6
Primary tumor	38.33	38.33	37.50	37.50	49.2
Nursery	85.63	85.42	79.91	84.32	26.0
Shuttle	94.86	94.07	93.68	94.86	7.5
KRKI	98.35	98.24	92.20	98.02	11.2

To minimise the amount of work within Double Induction, the conflicts already seen (for a particilar a fold) are saved, to allow re-use of the generated rules in the second round whenever the same conflict is observed again (within

Table 3. Result of McNemar's test to obtain p-values

	D. ind. w. weight	D. induction	Frequency-b.	Naive Bayes
D. induction	(2, 6), 0.289	(0, 0), 1	(10, 0), 1.953e-3*	(7, 3), 0.344
D. ind. w. weight	(0, 0), 1	(6, 2), 0.289	(9, 0), 3.91e-3*	(7, 0), 1.56e-2*
Naive Bayes	(0, 7), 1.56e-2*	(3, 7), 0.344	(7, 3), 0.344	(0, 0), 1
Frequency-b.	(0, 9), 3.91e-3	(0, 10), 1.953e-3*	(0, 0), 1	(3, 7), 0.344

the same fold). This can greatly speed up the classification process. For the current domains, this reuse of classifications vary from 2 percent to 60 percent of all classifications with conflicts.

4 Discussion

Instead of using the information conveyed in a set of conflicting rules (and the class distributions of the whole training set, if using Naive Bayes) and making a qualified guess of which class is most probable given the rules, we suggest doing a second round of rule induction in hope of finding a better separation of the examples. This is often possible due to that the induction algorithm only has to consider the examples that are relevant to region in which the example to be classified resides. This idea has been empirically demonstrated to significantly outperform two previous methods for resolving rule conflicts: frequency-based classification and Naive Bayes.

Furthermore, as seen by the experiments, it is worthwhile to use the weighting of the examples that is given by the first set of rules, because it helps in the second round to produce a better set of rules. This confirms the earlier observation in [7], that examples in the intersection of conflicting rules are more important than other examples.

One of the major draw backs of using Double induction is of course the computional cost, but if accuracy is of uttermost importance and quick response time is not, then it is a useful technique.

One issue that needs further investigation is the use of other weighting schemes than the one used in our experiments which is quite conservative. Another issue is to resolve new conflicts that occur between rules obtained in the second round, by continuing recursively instead of using Naive Bayes after the second round. This recursive step could be done until no conflicts are present in the resulting rules or until no new rules can be found.

References

1. Henrik Boström. Rule discovery system user manual, 2003.
2. P. Clark and R. Boswell. Rule induction with CN2: Some recent improvements. In *Proc. Fifth European Working Session on Learning*, pages 151–163, Berlin, 1991. Springer.

3. P. Clark and T. Niblett. The cn2 induction algorithm. *Machine Learning, 3, 261-283*, 1989.
4. J. Fürnkranz and G. Widmer. Incremental reduced error pruning. In W.W. Cohen and H. Hirsh, editors, *Proceedings of the 11th International Conference on Machine Learning*, pages 70–77. Morgan Kaufmann, 1994.
5. Johannes Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 1999.
6. R. Kohavi, B. Becker, and D. Sommerfield. Improving simple bayes. In *In Proceedings of the European Conference on Machine Learning*, 1997.
7. Tony Lindgren and Henrik Boström. Classification with intersecting rules. In *Proceedings of the 13th International Conference on Algorithmic Learning Theory (ALT'02)*, pages 395–402. Springer-Verlag, 2002.
8. Duda R. O. and Hart P. E. *Pattern Classification and Scene Analysis*. Wiley, 1973.
9. J.R. Quinlan. Induction of decision trees. *Machine Learning, 1, 81-106*, 1986.
10. R. Rivest. Learning decision lists. *Machine Learning, 2(3), 229-246*, 1987.