Classification with Intersecting rules

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. Several rule induction schemes generate hypotheses in the form of unordered rule sets. One important problem that has to be addressed when classifying examples with such hypotheses is how to deal with overlapping rules that predict different classes. Previous approaches to this problem calculate class probabilities based on the union of examples covered by the overlapping rules (as in CN2) or assumes rule independence (using naive Bayes). It is demonstrated that a significant improvement in accuracy can be obtained if class probabilities are calculated based on the intersection of the overlapping rules, or in case of an empty intersection, based on as few intersecting regions as possible.

1 Introduction

Methods for rule induction have been studied for more than two decades within the field of machine learning. They include various techniques such as divide-and-conquer (recursive partitioning), that generates hierarchically organized rules (decision trees) [4], and separate-and-conquer (covering) that generates overlapping rules. The sets of rules generated by separate-and-conquer may either be treated as ordered (decision lists) [5] or unordered [3, 2]. In case of inducing decision trees or decision lists, there is no need for resolving classification conflicts among the rules. In the former case this is due to that the rules are non-overlapping and hence there is only one single rule that is applicable to any given problem. In the latter case, this is due to that the first applicable rule in the list is always used.

In this work we focus on the problem of how to deal with overlapping rules that predict different classes in unordered rule sets. Previous approaches to this problem calculate class probabilities based on the union of examples covered by the overlapping rules (as in CN2 [2]) or assumes rule independence (using naive Bayes). We propose a novel approach to this problem that bases the calculation of class probabilities on the intersection, rather than the union, of the overlapping rules, or in case of an empty intersection, on as few intersecting regions as possible. The new method, called *intersection-based classification*, is compared to the two previous methods in an empirical evaluation.

The paper is organized as follows. In the next section, we briefly describe the two previous methods, which are here referred to as *union-based classification* and *naive Bayes classification*, together with an informal presentation of the novel method, *intersection-based classification*. In section three, we describe the algorithm in more detail and briefly present the system in which it has been implemented. The empirical evaluation is given in section four, and finally, in section five, we give some concluding remarks and point out directions for future research.

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 a new method.

2.1 Union-based classification

The system CN2 [2] resolves classification conflicts among rules in the following way. Given the examples in figure 1, the class frequencies in union of the rules that covers the example to be classified 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 $covers(R, C)$ gives the number of examples of class C that are covered by R. This means that CN2 would classify the example as belonging to the negative class (-). More generally:

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

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:

$$P(H|E) = P(H)\frac{P(E|H)}{P(E)}$$

where H is a hypotesis (in our case, a class label for the example to be classified) and E is our evidence (the rules that cover the example). As usual, since $P(E)$

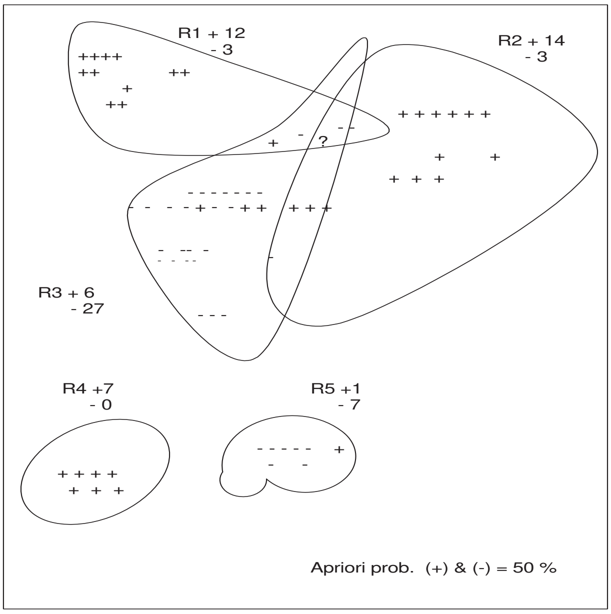

Fig. 1. Three rules covering an example to be classified (marked with ?). The training examples are labeled with their respective classes (+ and -).

does not affect the relative order of different hypotheses according to probability, it is ignored.

The naive Bayes assumption is that each piece of evidence is conditionally independent in relation to other evidence given the hypotesis. Hence, the maximum a posteriori probable hypotesis (MAP) according to the naive Bayes assumption 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.

If we again consider the example shown in figure 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 Intersection-based classification

The idea behind intersection-based classification is that if we are given some training examples in the intersecting region of the overlapping rules, this information should be used for the classification together with Bayes rule. In other words, it should be checked whether it is possible to be less naive than naive Bayes. In case there are some training examples that are covered by all rules that cover the example to be classified, Bayes rule can be used directly without having to assume independence of all rules. That is, from Bayes' rule:

$$P(H|E) = P(H)\frac{P(E|H)}{P(E)}$$

we obtain the following expression for the maximum a posteriori probable hypothesis:
$h_{MAP} = argmax_{Class_i \in Classes} P(Class_i)P(Rule_1 \wedge Rule_2 \wedge \ldots \wedge Rule_n|Class_i)$

If we again look at figure 1, intersection-based classification would assign the negative (-) class to the unknown example, since there exists a negative (-) example in the intersection between rule $R_1 \wedge R_2 \wedge R_3$ and the apriori is 40/80. This gives the negative class a value of $40/80 * (1+1)/(40+2) = 2.4e-2$ using Laplace correction (i.e., it is assumed that there is one additional example for each class that is covered by all rules), while the positive class gets a value of $40/80 * 1/(40+2) = 1.2e-2$.

However, if there are no training examples at all in the intersection, we can check whether a small number of (non-empty) intersecting regions can be formed.

Assume that there is no negative (-) training example in the intersection of $R_1 \cap R_2 \cap R_3$ in figure 1. The smallest number of elements in a partition of this set, such that the intersecting region of each element (subset) covers a non-empty set of training examples, is two. There are three possible partitions of this size, such that each intersection covers a non-empty sets of training examples: [[1],[2,3]],[[2],[1,3]] and [[3],[1,2]].

The probability values for partition one are:

$$Pos = 40/80 * (12+1)/(40+2) * (2+1)/(40+2) = 0.0111$$

$$Neg = 40/80 * (3+1)/(40+2) * (2+1)/(40+2) = 0.0034$$

The probability values for partition two is:

$$Pos = 40/80 * (14+1)/(40+2) * (1+1)/(40+2) = 0.0085$$

$$Neg = 40/80 * (3+1)/(40+2) * (2+1)/(40+2) = 0.0034$$

The probability values for partition three is:

$$Pos = 40/80 * (6+1)/(40+2) * (0+1)/(40+2) = 0.0020$$

$$Neg = 40/80 * (27+1)/(40+2) * (2+1)/(40+2) = 0.0238$$

The highest probability value is obtained for the negative class (-), which is the class that will be assigned to the example by the intersection-based classification method.

3 Algorithm for Intersection-based Classification

In this section we first give pseudo-code for the intersection-based classification algorithm, and then explain some parts of the algorithm in more detail.

Table 1. Pseudo-code for the Intersection-based Classification Algorithm.

```
IntersectionBasedClassification(Rules,Classes)
   begin {
     BestClassFreq := 0
     ClassFreq := 1
     NoElements := 0
     repeat
       NoElements := NoElements + 1
       NewPartitions := make_part(NoElements,Rules)
     until not_empty(NewPartitions)
     for each Partition in NewPartitions do {
       for each Class in Classes do {
         for each Part in Partition do
           ClassFreq := ClassFreq * covers(Part,Class)
       ClassFreq  := apriori(Class) * ClassFreq
       if ClassFreq > BestClassFreq then {
         BestClass := Class
         BestClassFreq := ClassFreq
       }
       }
     }
     return BestClass
   }
```

The intersection based classification algorithm takes as input the rules that are applicable to the example to be classified as well as the classes in the current domain.

The make_part function takes two arguments: the first argument tells how many elements the make_part function should generate in the partition of the applicable rules (which are given as the second argument). The make_part function is called in an iterative deepening fashion starting with the number of elements set to one.

The function not_empty goes through the partitions made and returns true if there is some partition for which the intersection of the rules in each subset is non-empty (i.e., contains at least one training example).

The algorithm finally computes the class probability for all generated partitions and returns the class label with maximum probability.

It should be noted that the algorithm may degrade to become identical to naive Bayes, in case none of the rules overlap on the training examples. In that case, all elements in the generated partition will consist of single rules.

4 Empirical evaluation

Intersection-based classification has been implemented in the system Virtual Predict [1], which is a platform for experimenting with various rule induction techniques, e.g., both separate-and-conquer and divide-and-conquer may be employed, and both ordered and unordered rule sets may be generated. The novel method is compared to naive Bayes and union-based classification. We first present the parameter settings that were used in Virtual Predict and describe the data sets that were used in the evaluation, and then give the experimental results.

4.1 Experimental setting

Table 2. Virtual Predict settings used in the experiment

Parameter	Value
Strategy	Separate and Conquer (SAC)
Probability estimate	M estimate, with M = 2
Structure cost	0.5
Measure	Information gain
Incremental reduced error pruning	Most Compressive
Experiment type	10 fold cross validation

There are several parameters that can be adjusted in Virtual Predict. The settings used in our experiments are shown in Table 2, and they all determine how rules are induced. All data sets 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 3 the domains used in the experiment is shown, as well as their main characteristics. The datasets were choosen to be as diverse as possible with respect to both size and difficulty.

4.2 Experimental results

The results from the eleven domains are shown in Table 4, 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 three classification methods. The last

Table 3. The domains used in the experiment

Domain	Classes	Class distribution	Examples
Shuttle Landing Control	2	47.8, 52.2	278
Car Evaluation	4	4, 4, 22, 70	1728
Balance Scale	3	8, 46, 46	625
Dermatology	6	5.5, 13.3, 14.2, 16.7, 19.7, 30.6	366
The Glass	2	24.1, 75.9	112
Congressional Votes	2	45.2, 54.8	435
KRKI	2	34, 66	1000
Liver-disorders	2	42, 58	345
Ionosphere	2	36, 64	351
Breast Cancer	2	29.7, 70.3	286
Lymphography	4	1.4, 2.7, 41.2, 54.7	148

column shows the percentage of all predictions for which at least two conflicting rules overlap on training data (this gives an upper bound on the amount of examples for which Intersection-based classification may perform in a less naive way than naive Bayes). The p-values according to an exact version of McNemar's test for obtaining the observed difference between the novel method and the two others are given in the columns after their accuracies. It can be seen that Intersection-based classification outperforms both Union-based and naive Bayes classification in all eleven domains. Even when considering only statistically significant differences ($p < 0.05$), intersection-based classification is more accurate in seven out of seven domains. The probability of obtaining this difference (7 wins and 0 losses), given that two methods are equally good, is 0.0078 according to a sign test. This means that the null hypothesis (no improvement is obtained with the new method) can be rejected at a 0.05 significance level.

5 Discussion

Previous approaches to the problem of classifying examples using conflicting rules calculate class probabilities based on the union of examples covered by the overlapping rules (union-based classification) or assumes rule independence (naive Bayes classification). We have demonstrated that a significant improvement in accuracy can be obtained if class probabilities are calculated based on the intersection of the overlapping rules, or in case of an empty intersection, based on as few intersecting regions as possible.

Union-based classification just sums all the covered classes and returns the class with the highest frequency. Note that this means that this strategy weights the examples in the intersection as more important than the rest of the examples. This follows from that the examples in the intersection are counted as many times as the number of conflicting rules. Naive Bayes does also weight the examples

Table 4. Intersecting rules compared with naive Bayes and Intersecting rules compared with Union based classification

Data Set	Inter.	naive B.	Sign.	Union-b.	Sign.	No. Conf.	Prediction
Shuttle	99.64	98.20	0.125	98.56	0.250	1.0	3.6 %
Car	93.75	93.17	4.139e-002	93.23	3.515e-002	25.1	14.5 %
Balance Scale	90.88	84.64	2.706e-007	86.08	5.704e-005	22.0	35.2 %
Dermatology	96.08	94.12	1.563e-002	93.28	1.953e-003	5.6	15.7 %
The Glass	94.64	92.86	0.500	92.86	0.500	0.6	5.4 %
C. Votes	96.78	95.40	7.031e-002	95.86	0.388	7.1	16.3 %
KRKI	99.50	99.20	0.375	95.80	1.455e-010	8.8	8.6 %
Liver-disorders	77.68	69.57	4.056e-005	68.99	1.522e-005	14.8	42.9 %
Ionosphere	92.02	89.46	2.246e-002	89.17	3.088e-002	4.7	13.4 %
Breast Cancer	77.62	71.68	4.883e-004	72.73	5.188e-004	6.1	21.3 %
Lymphography	84.46	77.03	7.385e-003	81.08	0.180	5.7	38.5 %

in the intersection in a similar fashion. Intersection-based classification take this notion to it's extreme and considers examples in the intersection only (if there are any, otherwise it tries to find a partition of the rules in conflict with as few elements as possible, where the intersection of each element covers some examples). The experiment supports the hypothesis that the most important information actually resides in the intersection of the rules.

The number of possible partitions to consider in the worst-case grows exponentially with the number of rules. Hence, using this method together with very large rule sets (e.g., as generated by ensemble learning techniques such as bagging or randomization), calls for more efficient (greedy) methods for partitioning the set of conflicting rules. However, in the current experiment the partitioning did not occur to a large extent, and the maximum number of rules that were applicable to any example was not very high (less than ten), keeping the computational cost at a reasonable level.

References

1. Henrik Boström. Virtual predict user manual, 2001.
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.R. Quinlan. Induction of decision trees. *Machine Learning, 1, 81-106*, 1986.
5. R. Rivest. Learning decision lists. *Machine Learning, 2(3), 229-246*, 1987.