Monotone Relabeling in Ordinal Classification

Ad Feelders, Universiteit Utrecht, The Netherlands
e-mail: ad@cs.uu.nl

***Abstract*—In many applications of data mining we know beforehand that the response variable should be increasing (or decreasing) in the attributes. Such relations between response and attributes are called monotone. In this paper we present a new algorithm to compute an optimal monotone classification of a data set for convex loss functions. Moreover, we show how the algorithm can be extended to compute *all* optimal monotone classifications with little additional effort. Monotone relabeling is useful for at least two reasons. Firstly, models trained on relabeled data sets often have better predictive performance than models trained on the original data. Secondly, relabeling is an important building block for the construction of monotone classifiers. We apply the new algorithm to investigate the effect on the prediction error of relabeling the training sample for k nearest neighbour classification and classification trees. In contrast to previous work in this area, we consider *all* optimal monotone relabelings. The results show that, for small training samples, relabeling the training data results in significantly better predictive performance.**

I. Introduction

In many applications of data mining we know beforehand that the response variable is increasing (or decreasing) in one or more of the attributes or features. For example, the sales price of a house - all else equal - increases with lot size, and smoking increases the probability of lung cancer. Such relations between response and attribute are called monotone. Examples also abound in ranking problems [7]. Consider, for example, the problem of ranking documents with respect to their relevance to a particular query. Typical attributes are counts of query terms in the abstract or title of the document. One would expect an increasing relationship between these counts and document relevance.

In this paper we present a new algorithm to compute an optimal monotone classification of a data set for convex loss functions. Monotone relabeling is useful for at least two reasons. Firstly, models trained on relabeled data often have better predictive performance than models trained on the original data. Improved performance of classifiers when trained on monotonized data has, for example, been reported in [11]. Hence, monotone relabeling can be viewed as a form of data cleaning. As an aside, we note that classifiers trained on monotone data are not necessarily monotone themselves.

Secondly, relabeling is an important building block for the construction of monotone classifiers. It is, for example, essential to the monotone nearest neighbour algorithm presented in [5], and in [9] the leaf nodes of a classification tree are relabeled in order to make it monotone. Also, some algorithms for learning monotone models require a monotone dataset [15], [12].

However, there are typically many loss-optimal monotone relabelings. Picking a single solution therefore to some extent leads to arbitrary results. We consider the problem of efficiently computing all optimal monotone classifications of a data set. We apply the relabeling algorithm to investigate the effect on the prediction error, of relabeling the training sample for a k nearest neighbour classifier and classification trees.

This paper is organized as follows. In the next section we introduce the required notions and notations for this paper. In section III we present the new relabeling algorithm and prove its correctness. An advantage of the algorithm is that it can be easily extended to compute all optimal relabelings, as is shown in section IV. In section V we discuss related work, and in section VI we show how the relabeling algorithm can be used as a data cleaning tool to improve the predictive performance of a k nearest neigbour classifier and classification trees. Finally, in section VII we draw conclusions and indicate possibilities for future work.

II. Preliminaries

Let $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ denote the set of N observations on attribute vector $\mathbf{X} = (X_1, X_2, \ldots, X_p)$ and ordinal class label Y. We assume the values of each attribute and the values of the class label are linearly ordered. Specifically, if there are k distinct class labels, we code them as $\{1, 2, \ldots, k\}$.

Let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ denote the set of *distinct* attribute vectors in D, and let $n(\mathbf{x}_i)$ and $n(\mathbf{x}_i, j)$ denote the number of observations in D with attribute values $\mathbf{x}_i$, respectively the number of observations in D with attribute values $\mathbf{x}_i$ and class label j. If the class label is increasing in attribute X, then all else equal, if X increases, the class label should not decrease. Combining these constraints for all attributes, we have the monotonicity constraint

$$\mathbf{x}_i \preceq \mathbf{x}_j \Rightarrow y_i \leq y_j,$$

where

$$\mathbf{x}_i \preceq \mathbf{x}_j \Leftrightarrow x_{i,h} \leq x_{j,h} \qquad \forall h = 1, \ldots, p.$$

We say a pair of points $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$ from D represents a monotonicity violation if:

$$\mathbf{x}_i \preceq \mathbf{x}_j \text{ and } y_i > y_j.$$

A monotone classification $c(\cdot)$ of D assigns a class label to each attribute vector in D, in such a way that $\forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$:

$$\mathbf{x}_i \preceq \mathbf{x}_j \Rightarrow c(\mathbf{x}_i) \leq c(\mathbf{x}_j). \tag{1}$$

The objective is to find a monotone classification that minimizes some given loss function. Let $H(\ell, j)$ denote the

loss incurred if we allocate an observation with class j to class ℓ. The total loss of $c(\cdot)$ is given by:

$$\mathcal{H}(c(\cdot)) = \sum_{i=1}^{n}\sum_{j=1}^{k} n(\mathbf{x}_i, j)H(c(\mathbf{x}_i), j). \tag{2}$$

The objective is to minimize (2) subject to constraint (1).

We close this section with a number of useful concepts. A subset L of $\mathcal{X}$ is a *lower set* of $\mathcal{X}$ with respect to $\preceq$, if $\mathbf{x} \in L$, $\mathbf{x}' \in \mathcal{X}$, and $\mathbf{x}' \preceq \mathbf{x}$ imply $\mathbf{x}' \in L$. Hence, if a lower set contains a particular element, it is required to also contain all lower ordered elements. Likewise, a subset U of $\mathcal{X}$ is an *upper set* if $\mathbf{x} \in U$, $\mathbf{x}' \in \mathcal{X}$, and $\mathbf{x} \preceq \mathbf{x}'$ imply $\mathbf{x}' \in U$.

The *downset* $\downarrow_j$ of a point $\mathbf{x}_j$ with respect to $(\mathcal{X}, \preceq)$ is given by

$$\downarrow_j = \{\mathbf{x}_i \in \mathcal{X} : \mathbf{x}_i \preceq \mathbf{x}_j\}$$

The *upset* of a point is defined analogously. Note that the downset of a point is a lower set, and the upset of a point is an upper set. Furthermore, we state without proof that the collection of lower sets (upper sets) is closed with respect to union and intersection. The complement of an upper set is a lower set and vice versa.

III. A New Relabeling Algorithm

We model the problem of computing a monotone relabeling with minimal empirical loss as a convex cost closure problem [8]. Let

$$f_i(\ell) = \sum_{j=1}^{k} n(\mathbf{x}_i, j)H(\ell, j),$$

denote the empirical loss incurred when the objects in D with attribute values $\mathbf{x}_i$ are assigned to class ℓ. If H is convex (e.g. absolute error or square error), then so is f, since in that case f is a nonnegative weighted sum of convex functions. With each attribute vector $\mathbf{x}_i$ in $\mathcal{X}$, we associate weights

$$w_i(\ell) = f_i'(\ell) = f_i(\ell + 1) - f_i(\ell),$$

for $\ell = 1, \ldots, k-1$. Hence $w_i(\ell)$ is the increase in loss when the class assigned to $\mathbf{x}_i$ is incremented from ℓ to $\ell + 1$.

In a slight abuse of notation we henceforth write $i \in \mathcal{X}$ rather than $\mathbf{x}_i \in \mathcal{X}$, and likewise in similar cases. For any subset S of $\mathcal{X}$, let

$$w_S(\ell) = \sum_{i \in S} w_i(\ell)$$

denote the weight of S.

Proposition 1. *There is an upper set $U^*(\ell)$ with minimum weight, that is a subset of all other minimum weight upper sets.*

Proof: Straightforward. □

Let $U^*(\ell)$ be the minimal minimum weight upper set (MMWUS),of $(\mathcal{X}, \preceq)$ with weights $w_i(\ell)$.

Proposition 2. *For all optimal relabelings $\mathbf{y}^*$: $y_j^* > \ell$ for all $j \in U^*(\ell)$.*

Proof: (Note: we will sometimes suppress dependency on ℓ in order to simplify notation.) Let $\mathbf{y}^*$ be an optimal solution, and suppose that $y_j^* \leq \ell$ for some $j \in U^*$. Call the set of all such points L. Note that L must be a lower set of U^*, otherwise $\mathbf{y}^*$ would violate the order constraints. Hence, $U^* \setminus L$ is an upper set of $(\mathcal{X}, \preceq)$. We conclude that

$$\sum_{j \in L} w_j(\ell) \equiv \sum_{j \in L} f_j'(\ell) < 0, \tag{3}$$

otherwise $U^* \setminus L$ would not have a larger weight than U^* and hence U^* would not be a MMWUS.

Because the loss function is convex and $y_j^* \leq \ell$ for $j \in L$, it follows from (3) that

$$\sum_{j \in L} f_j'(y_j^*) \leq \sum_{j \in L} f_j'(\ell) < 0.$$

This means we can improve $\mathbf{y}^*$ by setting $y_j^{**} = y_j^* + 1$, for all $j \in L$. The resulting solution still satisfies the order constraints. The existence of such a better solution contradicts our assumption that $\mathbf{y}^*$ was optimal. □

Summarizing, we have established that for $j \in U^*(\ell)$, $\mathbf{x}_j$ is assigned a label bigger than ℓ in *every* optimal solution.

Let $\bar{U}^*(\ell)$ denote the complement of $U^*(\ell)$.

Proposition 3. *There is an optimal solution with $y_j^* \leq \ell$ for all $j \in \bar{U}^*(\ell)$.*

Proof: Take any optimal solution $\mathbf{y}^*$ with values bigger than ℓ in $\bar{U}^*$. Let U denote the subset of $\bar{U}^*$ with values bigger than ℓ. Note that U must be an upper set of $\bar{U}^*$, otherwise $\mathbf{y}^*$ would violate the monotonicity constraint. Furthermore, we must have $\sum_{j \in U} f_j'(\ell) \geq 0$ because otherwise $U \cup U^*$ would be an upper set with lower weight than U^* which would contradict the assumption that U^* is a minimum weight upper set. From convexity of the loss function, and the fact that $y_j^* - 1 \geq \ell$ for $j \in U$, it follows that $f_j'(y_j^* - 1) \geq f_j'(\ell)$ for all $j \in U$. Therefore $\sum_{j \in U} f_j'(y_j^* - 1) \geq 0$ Hence, loss is not increased by setting $y_j^{**} = y_j^* - 1$ for all $j \in U$. Also this solution still meets the order requirements, so $\mathbf{y}^{**}$ is also an optimal solution. We continue like this until all values $\leq \ell$.□

Algorithm 1 Relabel($\mathcal{X}$, $\preceq$)

```
for ℓ = 1 to k − 1 do
    for i ∈ X do
        w_i ← f_i(ℓ + 1) − f_i(ℓ)
    end for
    U* ← minimal minimum weight upperset of (X,⪯,w)
    for all j ∈ Ū* do
        y*_j ← ℓ
    end for
    X ← X \ Ū*
end for
for i ∈ X do
    y*_i ← k
end for
return y*
```

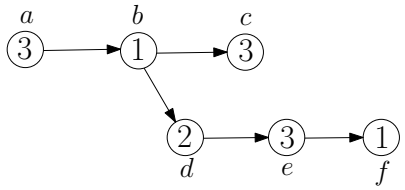

Fig. 1. Order on data points $\{a, \ldots, f\}$. Class labels are shown inside the nodes.

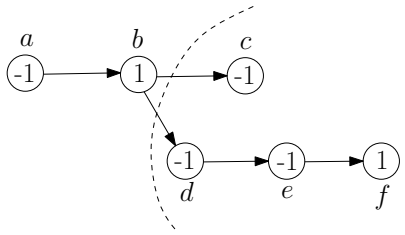

Fig. 2. Weights for $\ell = 1$ with absolute error. The minimal minimum weight upper set is $\{c, d, e, f\}$.

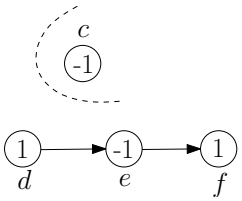

Fig. 3. Weights for $\ell = 2$ with absolute error. The minimal minimum weight upper set is $\{c\}$.

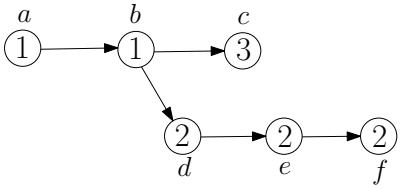

Fig. 4. Optimal solution produced by Algorithm 1. Class labels are shown inside the nodes.

Algorithm 1 combines the results presented in proposition 2 and propostion 3, to compute a monotone relabeling that minimizes empirical loss. The algorithm assigns the smallest possible label to each attribute vector. Once an attribute vector has been labeled it is removed from $\mathcal{X}$ (line 9). For brevity, we suppressed the obvious corresponding removal of weights from w and order relations from $\preceq$.

In line 5 of the algorithm, a MMWUS has to be computed. As Picard [14] has shown, this can be done by solving a maximum flow problem on a network that is easily constructed from the partial order on $\mathcal{X}$ and the weights w_i. Hence, Algorithm 1 has the same complexity as solving $k-1$ maximum flow problems.

To illustrate Algorithm 1, consider the ordering on data points $a, \ldots, f$ given in figure 1, where an arrow from a to b means that a directly precedes b in the ordering. The given labeling is not monotone, because a is smaller than b, but has a higher class label. The same is true for e and f. Figure 2 gives the weights for $\ell = 1$ using absolute error. There are two minimum weight upper sets with a weight of -2: $\{a, b, c, d, e, f\}$ and $\{c, d, e, f\}$. The minimal one is $U^* = \{c, d, e, f\}$: in any optimal solution, these points have a label of at least 2. Since $\bar{U}^* = \{a, b\}$ these points get assigned the label 1 by the algorithm (line 7), and they are removed from $\mathcal{X}$ (line 9).

Next, consider the weights for $\ell = 2$ as depicted in figure 3. Now $U^* = \{c\}$, hence c gets a label of at least 3. The points in $\bar{U}^* = \{d, e, f\}$ get assigned the label 2 and are subsequently removed from $\mathcal{X}$. Finally, the unlabeled point c gets assigned the highest class label (line 12). The final solution is depicted in figure 4.

IV. Computation of all optimal solutions

The algorithm presented in section III can be extended to compute all optimal solutions in a quite straightforward way. We showed that if $j \in U^*(\ell)$, then the label assigned to $\mathbf{x}_j$ must be bigger than ℓ. Likewise, one can show that if $j \in L^*(\ell)$, where $L^*(\ell)$ is the minimal maximum weight lower set of $(\mathcal{X}, \preceq)$ with weights $w_i(\ell)$, then the label assigned to $\mathbf{x}_j$ can be at most ℓ. The proof is analogous to the one for U^*, and is therefore ommitted. Hence, by computing $U^*(\ell)$ and $L^*(\ell)$ for $\ell = 1, 2, \ldots, k-1$, we can establish the interval of labels that can be assigned to each attribute vector in an optimal solution. The algorithm for computing these intervals is given in Algorithm 2.

Algorithm 2 Intervals($\mathcal{X}$, $\preceq$)

```
1:  for i ∈ X do
2:      [y_i^min, y_i^max] ← [1, k]
3:  end for
4:  for ℓ = 1 to k − 1 do
5:      for i ∈ X do
6:          w_i ← f_i(ℓ + 1) − f_i(ℓ)
7:      end for
8:      U* ← minimal minimum weight upperset of (X,⪯,w)
9:      for all j ∈ U* do
10:         y_j^min ← ℓ + 1
11:     end for
12:     L* ← minimal maximum weight lower set of (X,⪯,w)
13:     for all j ∈ L* do
14:         y_j^max ← ℓ
15:     end for
16:     X ← X \ L*
17: end for
18: return y^min, y^max
```

To illustrate, consider once more the example given in figure 1. We already saw that for $\ell = 1$, $U^* = \{c, d, e, f\}$. Hence, these points get lower bound 2. Since $L^* = \varnothing$, the upper bounds are not changed. For $\ell = 2$, we have $U^* = \{c\}$, so the lower bound of point c is increased to 3. The minimal

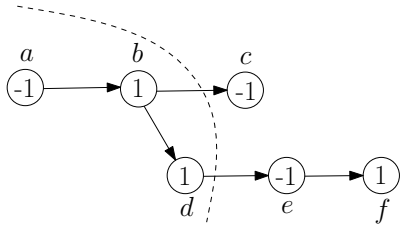

Fig. 5. Weights for $\ell = 2$ with absolute error. The minimal maximum weight lower set is $\{a, b, d\}$.

TABLE I
INTERVALS FOR OPTIMAL SOLUTIONS COMPUTED WITH ALGORITHM 2.

	init	$\ell = 1$	$\ell = 2$
a	$[1, 3]$	$[1, 3]$	$[1, 2]$
b	$[1, 3]$	$[1, 3]$	$[1, 2]$
c	$[1, 3]$	$[2, 3]$	$[3, 3]$
d	$[1, 3]$	$[2, 3]$	$[2, 2]$
e	$[1, 3]$	$[2, 3]$	$[2, 3]$
f	$[1, 3]$	$[2, 3]$	$[2, 3]$

maximum weight lower set for $\ell = 2$ is $L^* = \{a, b, d\}$ with a weight of 1 (see figure 5). Hence, these points get an upper bound of 2. Table I summarizes the steps of the algorithm, with the final solution in the column for $\ell = 2$.

We now have a straightforward (at least conceptually) procedure to generate all optimal solutions. First of all we note that the value of the loss function in the optimum is known, since taking the lower bound (or the upper bound) of all intervals is always optimal. We construct a monotone labeling, each label taken from the computed interval and verify whether its loss is optimal. Monotonicity is preserved by appropriately constraining the intervals of the remaining points when we pick a label for some point.

V. RELATED WORK

The first publication, to the best of our knowledge, that addresses the problem of exact minimization of an empirical loss function in the context of monotone *classification*, is Dykstra et al. [6]. They show how a monotone relabeling that minimizes absolute error or squared error can be obtained via the isotonic regression. To minimize squared error, a single application of isotonic regression is sufficient; for absolute error, $k - 1$ applications are required. The exact algorithm to compute the isotonic regression with best time complexity requires $O(n^4)$ computations [18], so this appears to be an expensive way to compute an optimal relabeling. Also, it is not clear how their approach could be extended to arbitrary convex loss functions.

Ryu et al. [2] discuss a monotone classification technique called *isotonic separation*. This technique also involves the minimization of empirical loss subject to the monotonicity constraint. The authors express this optimization problem as a linear program, and show that the dual of the linear program is in fact a maximum flow problem. Ryu et al. do not discuss the problem of multiple optima.

Rademaker et al. [16] pioneered monotone relabeling algorithms based on its formulation as a maximum independent set problem. This arises most naturally when considering the minimization of 0-1 loss. Consider the monotonicity violation graph that has an edge between two datapoints whenever they violate the monotonicity constraint, that is $\mathbf{x}_i \preceq \mathbf{x}_j$ and $y_i > y_j$. A maximum independent set in a graph is a maximum size subset of the nodes such that no pair of nodes in the subset are connected by an edge. A maximum independent set in the monotonicity violation graph corresponds to a maximum size monotone subset of the data. Relabeling its complement yields a monotone relabeling that minimizes 0-1 loss. Although the maximum independent set problem for arbitrary graphs is NP-complete, as Möhring [13] shows, it can be solved in polynomial time for the graph of a partial order. Rademaker et al. [17] show how the maximum independent set approach can be applied to the class of V-shaped loss functions, which is more general than the class of convex loss functions. On the other hand, in the general case, their algorithm requires the solution of a maximum independent set problem on a graph with $n \times k$ nodes thus requiring $O(n^3k^3)$ instead of $O(kn^3)$ computations. Furthermore, they do not consider the computation of all optimal relabelings.

Kotlowski et al. [10] consider ordinal classification with monotonicity constraints in the context of rough sets. They also consider the loss-optimal relabeling problem and establish a connection with the isotonic regression. Kotlowski and Slowinski [11] discuss the non-uniqueness of optimal solutions to the relabeling problem, and point out that for the class of *linear* loss functions the lower bound and upper bound of the optimal interval can be determined through two applications of the linear programming algorithm presented in [2]. A linear loss function is defined as

$$H(\ell, j) = \begin{cases} \alpha(\ell - j) & \text{if } \ell > j \\ (1 - \alpha)(j - \ell) & \text{if } \ell \leq j, \end{cases}$$

where $0 < \alpha < 1$. This is a more restrictive class of functions than the class of convex functions, and hence our result is more general.

VI. EXPERIMENTS

The experiments presented in this section mainly serve two purposes:

1) To rigorously show the positive influence of relabeling on the predictive performance of the resulting classifier. We do this by taking into account *all optimal relabelings*, and not just one of them as was done in previous studies, such as [11] and [3].
2) To show the variability in predictive performance with respect to different optimal relabelings of the same data set. This result underlines the importance of considering all possible relabelings in determining its influence on prediction error.

Since we are not proposing a new (monotone) classifier, we do not compare the results to those obtained with other (monotone) classification algorithms.

TABLE II
BASIC PROPERTIES OF THE DATA SETS. N DENOTES THE NUMBER OF OBSERVATIONS, AND n THE NUMBER OF DISTINCT ATTRIBUTE VECTORS.

Data set	N	n	# attr	# lab	% comp.	% mon.
Pima	768	768	8	2	7.32	97.76
LEV	1,000	92	4	5	24.08	95.73
ESL	488	199	4	9	70.65	98.85
Haberman	306	283	3	2	33.71	89.28
KC4	125	116	13	3	2.62	99.81
Wisconsin	194	194	32	2	0.56	96.19
SWD	1,000	117	10	4	12.62	94.21
CPU	209	190	6	4	49.53	98.52
AutoMPG	392	387	4	4	81.26	97.08
Ohsumed	235	55	2	3	77.99	80.79

TABLE III
MEAN ABSOLUTE PREDICTION ERROR OF k NEAREST NEIGHBOUR.

Data set	Original	Monotone	SE
Pima	.320 ± .025 (5)	**.317 ± .024** (5)	.004
LEV	.586 ± .045 (5)	**.558 ± .046** (1)	.020
ESL	.495 ± .045 (1)	**.474 ± .037** (1)	.017
Haberman	.258 ± .021 (11)	**.251 ± .018** (9)	.007
KC4	**.601 ± .082** (3)	.602 ± .078 (3)	.040
Wisconsin	**.253 ± .030** (11)	.253 ± .030 (11)	.001
SWD	.559 ± .041 (11)	**.559 ± .036** (11)	.021
CPU	.533 ± .068 (1)	**.504 ± .062** (1)	.012
AutoMPG	.400 ± .059 (9)	**.383 ± .031** (1)	.013
Ohsumed	.677 ± .057 (9)	**.624 ± .035** (1)	.013

Some basic properties of the data sets are listed in table II. The last but one column of this table gives the number of comparable pairs, that is, pairs of attribute vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ for which $\mathbf{x}_i \preceq \mathbf{x}_j$ or vice versa, expressed as a percentage of the total number of pairs. This gives some indication of the potential benefit of applying monotonicity constraints: the more comparable pairs, the higher the potential benefit. The final column gives the percentage of monontone pairs among the comparable pairs. If this percentage is low, application of the monotonicity constraint becomes questionable. The data sets come from different (public) sources. The Pima Indians Diabetes (Pima), Haberman's Survival (Haberman), Wisconsin Breast Cancer (Wisconsin), Computer Hardware (CPU), and Auto MPG data sets have been taken from the UCI machine learning repository [1]. The data sets LEV (Lecturer Evaluation), ESL (Employee Selection), and SWD (Social Workers Decisions) are available from the Weka website (`http://www.cs.waikato.ac.nz/ml/weka/`) and have been donated by Arie Ben-David. The KC4 dataset has been taken from the NASA data metrics program (`http://mdp.ivv.nasa.gov/`). The Ohsumed data set is available from the LETOR website [1]. For the Ohsumed data, we selected the data for query 3, and attributes 1 and 16. Monotonicity judgements were based on common sense. For example, the Ohsumed attributes are counts of the number of times a query term appears in the title and abstract of a document respectively. The class label indicates the relevance of the document to the query, where a higher label indicates higher relevance (the classes are described as: "irrelevant", "partially relevant" and "highly relevant"). Hence, it makes sense to assume that the class label is increasing in both attributes. The data sets KC4, CPU, and AutoMPG originally had numeric target variables. These numeric targets were discretized into four bins of approximately the same frequency.

Next we present the results of the experiments in which we study the effect of relabeling on the predictive accuracy. To this end, we compare the performance of a k nearest neighbour classifier using the original data with one that uses relabeled data. A similar comparison is made for classification trees. To eliminate variation due to the chosen relabeling, we average the performance over all possible relabelings. For k nearest neighbour, the following procedure is repeated 100 times:

1) Draw a sample of size 50 from the data.
2) Use this sample to classify the remaining data with k nearest neighbour, for different values of k.
3) Repeat the previous step for all possible optimal monotone relabelings of the sample and average the errors.

Finally, the errors are averaged over the 100 repetitions of the experiment. If relabeling does have a noise reduction effect, and the problem is indeed monotone, then one would expect an improved performance of the relabeled sample. We use a relatively small sample, because this is the case where applying the monotonicity constraint potentially has a positive effect. As the training sample gets bigger, the benefit of applying a correct constraint typically becomes smaller.

The results are displayed in table III. Before we analyze the results, please note that they were obtained with a very small training sample, and may therefore be much worse than other published results on the same data sets. For each data set, we give the lowest average prediction error found with the original data (the corresponding value of k is shown between brackets), the lowest average prediction error found with the relabeled data, and the average standard error of the prediction error over different optimal relabelings. More precisely, for each of the 100 repetitions of the experiment, we compute the standard deviation of the prediction error over different optimal relabelings of the training sample. If there is only one optimal relabeling, the standard deviation was set to zero. Our general conclusion is that relabeling has a positive effect, since in the majority of cases the relabeled data produces a lower prediction error. The two exceptions to this, KC4 and Wisconsin, are precisely the data sets with the lowest fraction of comparable pairs. If almost all attribute vectors are incomparable, then the monotonicity constraint won't have much effect. Relabeling reduces prediction error in 8 out of 10 data sets. When performing Wilcoxon's signed-ranks test, as recommended by Demšar [4], we obtain a p-value of approximately 0.02. Hence, the observed difference would be considered significant at the conventional $\alpha = 0.05$ level. A second interesting observation is that the optimal value for k tends to be smaller for the relabeled data. An extreme example is AutoMPG where $k = 1$ is the best value for the relabeled data, whereas this is $k = 9$ for the original data.

[1] http://research.microsoft.com/en-us/um/beijing/projects/letor/

TABLE IV
MEAN ABSOLUTE PREDICTION ERROR OF CLASSIFICATION TREES.

Data set	Original	Monotone	SE
Pima	.320 ± .046	**.311 ± .040**	.012
LEV	.636 ± .070	**.607 ± .043**	.040
ESL	.589 ± .086	**.539 ± .049**	.040
Haberman	**.276 ± .028**	.279 ± .027	.016
KC4	.690 ± .143	**.643 ± .089**	.053
Wisconsin	**.255 ± .045**	.258 ± .044	.007
SWD	.610 ± .087	**.605 ± .043**	.043
CPU	.504 ± .067	**.481 ± .058**	.038
AutoMPG	.397 ± .064	**.358 ± .036**	.020
Ohsumed	.695 ± .060	**.619 ± .032**	.012

Our conclusion is that because of the noise reduction effect of relabeling, the danger of overfitting is less severe, and $k = 1$ already produces quite good results.

The standard deviation of the prediction error over different optimal relabelings is substantial in relation to the standard deviation over different training samples: it can make a substantial difference for the prediction error of the classifier which optimal relabeling is chosen. To take an extreme example: for $k = 1$ on the ESL data, the standard deviation as a consequence of drawing different training samples is .037 (for nearest neighbour with relabeling), and the standard deviation as a consequence of choosing different relabelings of the same training sample on average is .017, that is, far from negligible. Therefore we argue that it is better to average over all possible relabelings to get a reliable assessment of the utility of relabeling the data.

A similar comparison was made for classification trees. Here we followed the following procedure:

1) Draw a sample of size 50 from the data.
2) Use 5-fold cross validation on this sample to select the tree from the cost-complexity pruning sequence with lowest error. In case of a tie the smallest tree was selected. Use the selected tree to classify the remaining data.
3) Repeat the previous step for all possible optimal monotone relabelings of the sample and average the errors.

Finally, the errors are averaged over the 100 repetitions of the experiment.

The results are presented in table IV. We note that in general relabeling results in a reduction of prediction error, the data sets Haberman and Wisconsin being the exceptions. When performing Wilcoxon's signed-ranks test we obtain a p-value of approximately 0.01. Hence, the observed difference would be considered significant at the conventional $\alpha = 0.05$ level. Also note that the variation in prediction error with respect to different optimal relabelings is again substantial. To take an extreme case: for the social workers decisions dataset, it is as big as the variation in prediction error with respect to drawing different training samples.

VII. CONCLUSION

We have presented a new efficient algorithm for computing all optimal monotone classifications of a data set for convex loss functions. We have shown that k nearest neighbour classifiers and classification trees learned from relabeled data tend to have lower prediction error than their counterparts constructed from the original data. Furthermore, we have shown that the particular optimal relabeling chosen can have a substantial influence on the prediction error of the resulting classifier. Therefore, it is important to consider all optimal relabelings in assessing its effect.

The number of relabelings could be so huge that it is not possible to consider them all. Therefore, we are looking into methods to draw a uniform sample from them. Furthermore, our algorithm may initially generate suboptimal solutions from the computed intervals, which then have to be discarded. It would be interesting to investigate whether we can prevent their generation. Finally, we are considering to extend the methods presented to ranking problems where often different loss functions are used.

REFERENCES

[1] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.
[2] R. Chandrasekaran, Y.R. Ruy, V.S. Jacob, and S. Hong. Isotonic separation. *INFORMS Journal On Computing*, 17(4):462–474, 2005.
[3] H. Daniels and M. Velikova. Derivation of monotone decision models from noisy data. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 36(5):705–710, 2006.
[4] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.
[5] W. Duivesteijn and A. Feelders. Nearest neighbour classification with monotonicity constraints. In W. Daelemans, editor, *Proceedings of ECML/PKDD 2008*, volume 5211 of *LNAI*, pages 301–316. Springer, 2008.
[6] R. Dykstra, J. Hewett, and T. Robertson. Nonparametric, isotonic discriminant procedures. *Biometrika*, 86(2):429–438, 1999.
[7] J. Fürnkranz and E. Hüllermeier. Pairwise preference learning and ranking. In N. Lavrač, D. Gamberger, H. Blockeel, and L. Todorovski, editors, *Proceedings of the 14th European Conference on Machine Learning*, pages 145–156, 2003.
[8] D.S. Hochbaum and M. Queyranne. Minimizing a convex cost closure set. *SIAM Journal Of Discrete Mathematics*, 16(2):192–207, 2003.
[9] R. van de Kamp, A. Feelders, and N. Barile. Isotonic classification trees. In N. Adams, editor, *Proceedings of IDA 2009*, volume 5772 of *LNCS*, pages 405–416. Springer, 2009.
[10] W. Kotlowski, K. Dembczynski, S. Greco, and R. Slowinski. Stochastic dominance-based rough set model for ordinal classification. *Information Sciences*, 178:4019–4037, 2008.
[11] W. Kotlowski and R. Slowinski. Statistical approach to ordinal classification with monotonicity constraints. In E. Hüllermeier and J. Fürnkranz, editors, *ECML PKDD 2008 Workshop on Preference Learning*, 2008.
[12] W. Kotlowski and R. Slowinski. Rule learning with monotonicity constraints. In L. Bottou and M. Littman, editors, *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 537–544, 2009.
[13] R.H. Möhring. Algorithmic aspects of comparability graphs and interval graphs. In I. Rival, editor, *Graphs and Order*, pages 41–101. Reidel, 1985.
[14] J.-C. Picard. Maximal closure of a graph and applications to combinatorial problems. *Management Science*, 22(11):1268–1272, 1976.
[15] R. Potharst and J.C. Bioch. Decision trees for ordinal classification. *Intelligent Data Analysis*, 4(2):97–112, 2000.
[16] M. Rademaker, B. De Baets, and H. De Meyer. On the role of maximal independent sets in cleaning data for supervised ranking. In *2006 IEEE International Conference on Fuzzy Systems*, pages 1619–1624. 2006.
[17] M. Rademaker, B. De Baets, and H. De Meyer. Loss optimal monotone relabeling of noisy multi-criteria data sets. *Information Sciences*, 179:4089–4096, 2009.
[18] J. Spouge, H. Wan, and W.J. Wilbur. Least squares isotonic regression in two dimensions. *Journal Of Optimization Theory And Applications*, 117(3):585–605, 2003.