Nonparametric Ordinal Classification with Monotonicity Constraints

Nicola Barile and Ad Feelders

Utrecht University, Department of Information and Computing Sciences,
P.O. Box 80089, 3508TB Utrecht, The Netherlands,
{barile,ad}@cs.uu.nl

Abstract. In many applications of ordinal classification we know that the class label must be increasing (or decreasing) in the attributes. Such relations are called monotone. We discuss two nonparametric approaches to monotone classification: OSDL and MOCA. Our conjecture is that both methods have a tendency to overfit on the training sample, because their basic class probability estimates are often computed on a few observations only. Therefore, we propose to smooth these basic probability estimates by using weighted k nearest neighbour. Through substantial experiments we show how this adjustment improves the classification performance of OSDL considerably. The effect on MOCA on the other hand is less conclusive.

1 Introduction

In many applications of data analysis it is reasonable to assume that the response variable is increasing (or decreasing) in one or more of the attributes or features. Such relations between response and attribute are called monotone. Besides being plausible, monotonicity may also be a desirable property of a decision model for reasons of explanation, justification and fairness. Consider two applicants for the same job, where the one who scores worse on all criteria gets the job.

While human experts tend to feel uncomfortable expressing their knowledge and experience in terms of numeric assessments, they typically are able to state their knowledge in a semi-numerical or qualitative form with relative conviction and clarity, and with less cognitive effort [9]. Experts, for example, can often easily indicate which of two probabilities is smallest. In addition to requiring less cognitive effort, such relative judgments tend to be more reliable than direct numerical assessments [18].

Hence, monotonicity constraints occur frequently in learning problems and such constraints can be elicited from subject area experts with relative ease and reliability. This has motivated the development of algorithms that are able to enforce such constraints in a justified manner. Several data mining techniques have been adapted in order to be able to handle monotonicity constraints in one form or another. Examples are: classification trees [19, 10, 6], neural networks [20, 21], Bayesian networks [1, 11] and rules [8].

In this paper, we confine our attention to two nonparametric approaches to monotone classification: OSDL [7, 15] and MOCA [4]. These methods rely on the estimation of the class probabilities for each observed attribute vector. These basic estimates as we will call them are then further processed in order to extend the classifier to the entire attribute space (by interpolation), and to guarantee the monotonicity of the resulting classification rule. Because the basic estimates are often based on very few observations, we conjecture that OSDL and MOCA are prone to overfitting. Therefore we propose to smooth the basic estimates by including observations that are near to where an estimate is required. We perform a substantial number of experiments to verify whether this indeed improves the classification performance.

This paper is organized as follows. In the next section, we establish some concepts and notation that will be used throughout the paper. In section 3 we give a short description of OSDL and MOCA and establish similarities and differences between them. We also provide a small example to illustrate both methods. In section 4 we propose how to adapt the basic estimates that go into OSDL and MOCA, by using weighted k nearest neighbour. Subsequently, these adapted estimates are tested experimentally in section 5. We compare the original algorithms to their adapted counterparts, and test whether significant differences in predictive performance can be found. Finally, we draw conclusions in section 6.

2 Preliminaries

Let $\mathbf{X}$ denote the vector of attributes, which takes values $\mathbf{x}$ in a p-dimensional input space $\mathcal{X} = \times \mathcal{X}_i$, and let Y denote the class variable which takes values y in a one-dimensional space $\mathcal{Y} = \{1, 2, \ldots, q\}$, where q is the number of class labels. We assume that the values in $\mathcal{X}_i$, $i = 1, \ldots, p$, and the values in $\mathcal{Y}$ are totally ordered. An attribute X_i has a *positive* influence on Y if for all $x_i, x_i' \in \mathcal{X}_i$:

$$x_i \leq x_i' \Rightarrow P(Y|x_i, \mathbf{x}_{-i}) \preceq P(Y|x_i', \mathbf{x}_{-i}) \tag{1}$$

where $\mathbf{x}_{-i}$ is any value assignment to the attributes other than X_i [22]. Here $P(Y|x_i, \mathbf{x}_{-i}) \preceq P(Y|x_i', \mathbf{x}_{-i})$ means that the distribution of Y for attribute values $(x_i, \mathbf{x}_{-i})$ is stochastically smaller than for attribute values $(x'_i, \mathbf{x}_{-i})$, that is

$$F(y|x_i, \mathbf{x}_{-i}) \geq F(y|x'_i, \mathbf{x}_{-i}), \quad y = 1, 2, \ldots, q$$

where $F(y) = P(Y \leq y)$. In words: for the larger value of X_i, larger values of Y are more likely. A negative influence is defined analogously, where for larger values of X_i smaller values of Y are more likely. Without loss of generality, we henceforth assume that all influences are positive. A negative influence from X_i to Y can be made positive simply by reordering the values in $\mathcal{X}_i$.

Considering the constraints (1) corresponding to all positive influences together, we get the constraint:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathcal{X} : \mathbf{x} \preceq \mathbf{x}' \Rightarrow P(Y|\mathbf{x}) \preceq P(Y|\mathbf{x}'), \tag{2}$$

where the order on $\mathcal{X}$ is the product order

$$\mathbf{x} \preceq \mathbf{x}' \Leftrightarrow \forall i = 1, \ldots, p : x_i \leq x'_i.$$

It is customary to evaluate a classifier on the basis of its error-rate or 0/1 loss. For classification problems with ordered class labels this choice is less obvious. It makes sense to incur a higher cost for those misclassifications that are "far" from the true label, than to those that are "close". One loss function that has this property is L_1 loss:

$$L_1(i, j) = |i - j| \qquad\qquad i, j = 1, \ldots, q \tag{3}$$

where i is the true label, and j the predicted label. We note that this is not the only possible choice. One could also choose L_2 loss for example, or another loss function that has the desired property that misclassifications that are far from the true label incur a higher loss. Nevertheless, L_1 loss is a reasonable candidate, and in this paper we confine our attention to this loss function. It is a well known result from probability theory that predicting the median minimizes L_1 loss.

A median m of Y has the property that $P(Y \leq m) \geq 0.5$ and $P(Y \geq m) \geq 0.5$. The median may not be unique. Let m_ℓ denote the smallest median of Y and let m_u denote the largest median. We have [15]

$$P(Y|\mathbf{x}) \preceq P(Y|\mathbf{x}') \Rightarrow m_\ell(\mathbf{x}) \leq m_\ell(\mathbf{x}') \wedge m_u(\mathbf{x}) \leq m_u(\mathbf{x}')$$

The above result shows that predicting the smallest (or largest) median gives an allocation rule $c : \mathcal{X} \rightarrow \mathcal{Y}$ that satisfies $\forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}$:

$$\mathbf{x} \preceq \mathbf{x}' \Rightarrow c(\mathbf{x}) \leq c(\mathbf{x}'), \tag{4}$$

that is, a lower ordered input cannot have a higher class label. Kotlowski [14] shows that if a collection of probability distributions satisfies the stochastic order constraint (2), then the Bayes allocation rule $c_{\text{B}}(\cdot)$ satisfies the monotonicity constraint (4), provided the loss function is *convex.* This encompasses many reasonable loss functions but not 0/1 loss, unless the class label is binary.

Let $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ denote the set of observed data points in $\mathcal{X} \times \mathcal{Y}$, and let Z denote the set of distinct $\mathbf{x}$ values occurring in D. We define the downset of $\mathbf{x}$ with respect to Z to be the set $\{\mathbf{x}' \in Z : \mathbf{x}' \preceq \mathbf{x}\}$. The upset of $\mathbf{x}$ is defined analogously. Any real-valued function f on Z is *isotonic* with respect to $\preceq$ if, for any $\mathbf{x}, \mathbf{x}' \in Z$, $\mathbf{x} \preceq \mathbf{x}'$ implies $f(\mathbf{x}) \leq f(\mathbf{x}')$. Likewise, a real-valued function a on Z is *antitonic* with respect to $\preceq$ if, for any $\mathbf{x}, \mathbf{x}' \in Z$, $\mathbf{x} \preceq \mathbf{x}'$ implies $a(\mathbf{x}) \geq a(\mathbf{x}')$.

3 OSDL and MOCA

In this section we give a short description of OSDL and MOCA, and discuss their similarities and differences.

3.1 OSDL

The ordinal stochastic dominance learner (OSDL) was developed by Cao-Van [7] and generalized by Lievens et al. in [15]. Recall that Z is the set of distinct $\mathbf{x}$ values present in the training sample D. Let

$$\hat{P}(y|\mathbf{x}) = \frac{n(\mathbf{x}, y)}{n(\mathbf{x})}, \qquad \mathbf{x} \in Z, y = 1, \ldots, q$$

where $n(\mathbf{x})$ denotes the number of observations in D with attribute values $\mathbf{x}$, and $n(\mathbf{x}, y)$ denotes the number of observations in D with attribute values $\mathbf{x}$ and class label y. Furthermore, let

$$\hat{F}(y|\mathbf{x}) = \sum_{j \leq y} \hat{P}(j|\mathbf{x}), \qquad \mathbf{x} \in Z$$

denote the unconstrained maximum likelihood estimate of

$$F(y|\mathbf{x}) = P(Y \leq y|\mathbf{x}), \mathbf{x} \in Z.$$

To obtain a collection of distribution functions that satisfy the stochastic order restriction, Cao-Van [7] defines:

$$F^{\min}(y|\mathbf{x}_0) = \min_{\mathbf{x} \preceq \mathbf{x}_0} \hat{F}(y|\mathbf{x}) \tag{5}$$

and

$$F^{\max}(y|\mathbf{x}_0) = \max_{\mathbf{x}_0 \preceq \mathbf{x}} \hat{F}(y|\mathbf{x}), \tag{6}$$

where $\mathbf{x} \in Z$. If there is no point $\mathbf{x}$ in Z such that $\mathbf{x} \preceq \mathbf{x}_0$, then $F^{\min}(y|\mathbf{x}_0) = 1$ $(y = 1, \ldots, q)$, and if there is no point $\mathbf{x}$ in Z such that $\mathbf{x}_0 \preceq \mathbf{x}$, then $F^{\max}(y|\mathbf{x}_0) = 0$ $(y = 1, \ldots, q-1)$, and $F^{\min}(q|\mathbf{x}_0) = 1$.

Note that $\forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}$:

$$\mathbf{x} \preceq \mathbf{x}' \ \Rightarrow F^{\min}(y|\mathbf{x}) \geq F^{\min}(y|\mathbf{x}') \tag{7}$$

$$\mathbf{x} \preceq \mathbf{x}' \Rightarrow F^{\max}(y|\mathbf{x}) \geq F^{\max}(y|\mathbf{x}'). \tag{8}$$

Proposition (7) holds, since the downset of $\mathbf{x}$ is a subset of the downset of $\mathbf{x}'$, and the minimum taken over a given set is never above the minimum taken over one of its subsets. Proposition (8) follows similarly.

In the *constant interpolation* version of OSDL, the final estimates are obtained by putting

$$\tilde{F}(y|\mathbf{x}_0) = \alpha F^{\min}(y|\mathbf{x}_0) + (1-\alpha) F^{\max}(y|\mathbf{x}_0), \tag{9}$$

with $\alpha \in [0, 1]$.

This rule is used both for observed data points, as well as for new data points. The interpolation parameter α is a free parameter whose value can be selected so as to minimize empirical loss on a test sample. Note that $\tilde{F}$ satisfies the stochastic order constraint, because both (7) and (8) hold. More sophisticated interpolation schemes called *balanced* and *double balanced* OSDL are discussed in [15]; we refer the reader to this paper for details. These OSDL versions are also included in the experimental evaluation that is presented in section 5.

3.2 MOCA

In this section, we give a short description of MOCA. For each value of y, the MOCA estimator $F^*(y|\mathbf{x}), \mathbf{x} \in Z$ minimizes the sum of squared errors

$$\sum_{\mathbf{x} \in Z} n(\mathbf{x}) \left\{ \hat{F}(y|\mathbf{x}) - a(\mathbf{x}) \right\}^2 \tag{10}$$

within the class of antitonic functions $a(\mathbf{x})$ on Z. This is an isotonic regression problem. It has a unique solution, and the best time complexity known is $O(|Z|)^4$ [17]. The algorithm has to be performed $q-1$ times, since obviously $F^*(q|\mathbf{x}) = 1$.

Note that this estimator satisfies the stochastic order constraint $\forall \mathbf{x}, \mathbf{x}' \in Z$:

$$\mathbf{x} \preceq \mathbf{x}' \Rightarrow F^*(y|\mathbf{x}) \geq F^*(y|\mathbf{x}') \qquad y = 1, \ldots, q \tag{11}$$

by construction.

Now the isotonic regression is only defined on the observed data points. Typically the training sample does not cover the entire input space, so we need some way to estimate $F(y|\mathbf{x}_0)$ for points $\mathbf{x}_0$ not in the training sample. Of course these estimates should satisfy the stochastic order constraint with respect to $F^*(Y|\mathbf{x})$. Hence, we can derive the following bounds:

$$F^{\min}(y|\mathbf{x}_0) = \max_{\mathbf{x}_0 \preceq \mathbf{x}} F^*(y|\mathbf{x}) \qquad y = 1, \ldots, q \tag{12}$$

and

$$F^{\max}(y|\mathbf{x}_0) = \min_{\mathbf{x} \preceq \mathbf{x}_0} F^*(y|\mathbf{x}) \qquad y = 1, \ldots, q \tag{13}$$

If there is no point $\mathbf{x}$ in Z such that $\mathbf{x} \preceq \mathbf{x}_0$, then we put $F^{\max}(y|\mathbf{x}_0) = 1$ $(y = 1, \ldots, q)$, and if there is no point $\mathbf{x}$ in Z such that $\mathbf{x}_0 \preceq \mathbf{x}$, then we put $F^{\min}(y|\mathbf{x}_0) = 0$ $(y = 1, \ldots, q-1)$, and $F^{\min}(q|\mathbf{x}_0) = 1$.

Because F^* is antitonic we always have $F^{\min}(y) \leq F^{\max}(y)$. Any choice from the interval $[F^{\min}(y), F^{\max}(y)]$ satisfies the stochastic order constraint with respect to the training data.

A simple interpolation scheme that is guaranteed to produce globally monotone estimates is to take the convex combination

$$\breve{F}(y|\mathbf{x}_0) = \alpha F^{\min}(y|\mathbf{x}_0) + (1-\alpha) F^{\max}(y|\mathbf{x}_0), \tag{14}$$

with $\alpha \in [0, 1]$. Note that for $\mathbf{x}_0 \in Z$, we have $\breve{F}(y|\mathbf{x}_0) = F^*(y|\mathbf{x}_0)$, since both $F^{\min}(y|\mathbf{x}_0)$ and $F^{\max}(y|\mathbf{x}_0)$ are equal to $F^*(y|\mathbf{x}_0)$. The value of α can be chosen so as to minimize empirical loss on a test sample.

Since MOCA should produce a class prediction, we still have to specify an allocation rule. MOCA allocates $\mathbf{x}$ to the smallest median of $\breve{F}(Y|\mathbf{x})$:

$$c^*(\mathbf{x}) = \min_y : \breve{F}(y|\mathbf{x}) \geq 0.5.$$

First of all, note that since $\breve{F}(y)$ satisfies the stochastic order constraint (2), c^* will satisfy the monotonicity constraint given in (4). Furthermore, it can be shown (see [4]) that c^* minimizes L_1 loss

$$\sum_{i=1}^{N} |y_i - c(\mathbf{x}_i)|$$

within the class of monotone integer-valued functions $c(\cdot)$. In other words, of all monotone classifiers, c^* is among the ones (there may be more than one) that minimize L_1 loss on the training sample.

3.3 An Example

To illustrate OSDL and MOCA, we present a small example. Suppose we have two real-valued attributes X_1 and X_2 and a ternary class label Y, that is, $\mathcal{Y} = \{1, 2, 3\}$. Consider the dataset given in Figure 1.

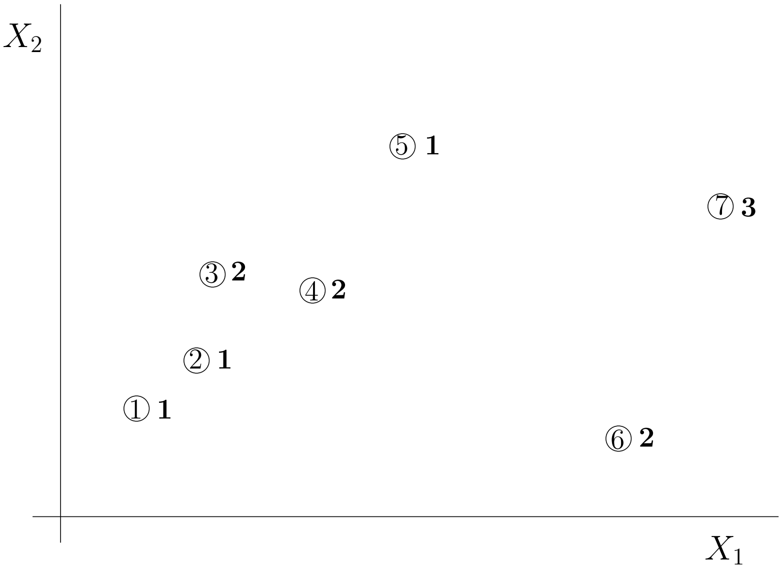

Fig. 1. Data for example. Observations are numbered for identification. Class label is printed in boldface next to the observation.

Table 1 gives the different estimates of F. Since all attribute vectors occur only once, the estimates $\hat{F}$ are based on only a single observation. The attribute vector of observation 5 is bigger than that of 3 and 4, but observation 5 has a smaller class label. This leads to order reversals in $\hat{F}(1)$. We have $\hat{F}(1|3)$ and $\hat{F}(1|4)$ smaller than $\hat{F}(1|5)$ (where in a slight abuse of notation we are conditioning on observation numbers), but observation 5 is in the upset of 3 and 4. In this case, the antitonic regression resolves this order reversal by averaging these violators:

$$F^*(1|3) = F^*(1|4) = F^*(1|5) = \frac{0+0+1}{3} = \frac{1}{3}$$

This is the only monotonicity violation present in $\hat{F}$ so no further averaging is required. We explain the computation of the *constant interpolation* version of OSDL estimate through an example. In Table 1 we see that $\tilde{F}(1|3) = 1/2$. This is computed as follows:

$$F^{\min}(1|3) = \min\{\hat{F}(1|1), \hat{F}(1|2), \hat{F}(1|3)\} = \min\{1, 1, 0\} = 0,$$

since the downset of observation 3 is $\{1, 2, 3\}$. Likewise, we can compute

$$F^{\max}(1|3) = \max\{\hat{F}(1|3), \hat{F}(1|5), \hat{F}(1|7)\} = \max\{0, 1, 0\} = 1,$$

since the upset of observation 3 is $\{3, 5, 7\}$. Combining these together with $\alpha = 0.5$ we get:

$$\tilde{F}(1|3) = \alpha F^{\min}(1|3) + (1 - \alpha) F^{\max}(1|3) = \frac{1}{2}$$

Table 1. Maximum Likelihood, MOCA and OSDL ($\alpha = 0.5$) estimates of $F(1)$ and $F(2)$.

	$\hat{F}$ (MLE)			$\breve{F}$(MOCA)			$\tilde{F}$(OSDL)			
obs	1	2	y	1	2	c^*	1	2	$c_{\min}$	$c_{\max}$
1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1
3	0	1	2	1/3	1	2	1/2	1	1	2
4	0	1	2	1/3	1	2	1/2	1	1	2
5	1	1	1	1/3	1	2	1/2	1	1	2
6	0	1	2	0	1	2	0	1	2	2
7	0	0	3	0	0	3	0	1/2	2	3

The MOCA allocation rule c^* allocates to the smallest median of $\breve{F}$, which gives a total absolute error of 1, since observation 5 has label 1, but is predicted to have label 2 by c^*. All other predictions of c^* are correct. An absolute error of 1 is the minimum achievable on the training sample for any monotone classifier. For OSDL we have given two allocation rules: one that assigns to the smallest median ($c_{\min}$) and one that assigns to the largest median ($c_{\max}$). The former has an absolute error of 3 on the training sample, and the latter achieves the minimum absolute error of 1: it is identical to c^* in this case.

3.4 Comparison of OSDL and MOCA

The reader will have noticed the similarity between MOCA and OSDL: MOCA uses the same interpolation method, and the MOCA definitions of $F^{\min}$ and $F^{\max}$ are the reverse of the corresponding definitions for OSDL. An important difference is that OSDL plugs in the maximum likelihood estimates $\hat{F}$ in equations (5) and (6), whereas MOCA plugs in the isotonic regression estimates F^* in equations (12) and (13). It should be noted that OSDL in principle allows any estimate of $F(y|\mathbf{x})$ to be plugged into equations (5) and (6). From that viewpoint, MOCA can be viewed as an instantiation of OSDL. However, to the best of our knowledge only the unconstrained maximum likelihood estimate has been used in OSDL to date.

Because F^* is plugged in, MOCA is guaranteed to minimize L_1 loss on the training sample. While this is a nice property, the objective is not to minimize L_1 loss on the training sample. It remains to be seen whether this also results in better out-of-sample predictive performance.

It should be noted that if $\hat{F}$ already satisfies the stochastic order restriction, then both methods are identical. In that case the isotonic regression will not make any changes to $\hat{F}$, since there are no order reversals.

Our conjecture is that both methods have a tendency to overfit on the training data. In many applications attribute vectors occur in D only once, in particular when the attributes are numeric. Hence the basic estimates that go into equations (5) and (6) are usually based on a single observation only. Now the interpolation performed in equation (14) will have some *smoothing* effect, but it is the question whether this is sufficient to prevent overfitting. The same reasoning applies to MOCA, but to a lesser extent because the isotonic regression has an additional smoothing effect: in case of order reversals basic estimates are averaged to remove the monotonicity violation. Nevertheless, it is possible that MOCA could be improved by performing the isotonic regression on a smoothed estimate rather than on $\hat{F}$ in (10). This is what we propose in the next section.

4 Weighted *k*NN probability estimation

In order to prevent overfitting in estimating $P(Y \leq y|\mathbf{x})$, $\mathbf{x} \in Z$, we develop a weight-based estimator based on the nearest neighbours principle along the lines of the one introduced in [13].

In the following, we first discuss kNN as a classification technique and then illustrate how we use it to perform probability estimation.

4.1 k Nearest Neighbour Classification

The k *Nearest Neighbor* technique is an example of *instance-based learning*: the training dataset is stored, and the classification of new, unlabelled instances is performed by comparing each of them to the k most similar (least dissimilar) elements to it in the training dataset. The dissimilarity is determined by means of a *distance metric* or *function*, which is a real-valued function d such that for any data points $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$:

1. $d(\mathbf{x}, \mathbf{y}) > 0$, $d(\mathbf{x}, \mathbf{x}) = 0$;
2. $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$;
3. $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$;

The distance measure which we adopted is the *Euclidean distance*:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{p} (x_i - y_i)^2}.$$

In order to avoid attributes that have large values from having a stronger influence than attributes measured on a smaller scale, it is important to normalize the attribute values. We adopted the *Z-score standardization* technique, whereby each value x of an attribute X is replaced by

$$\frac{x - \bar{x}}{s_X},$$

where s_X denotes the sample standard deviation of X. Once a distance measure to determine the neighbourhood of an unlabelled observation $\mathbf{x}_0$ has been selected, the next step to use kNN as a classification technique is represented by determining a criterion whereby the selected labelled individuals will be used to determine the label of $\mathbf{x}_0$.

The most straightforward solution is represented by *(unweighted) majority voting*: the chosen label is the one occurring most frequently in the neighbourhood of $\mathbf{x}_0$.

4.2 Weighted kNN Classification

In kNN it is reasonable to request that neighbours that are closer to $\mathbf{x}_0$ have a greater importance in deciding its class than those that are more distant.

In the *Weighted Nearest Neighbours* approach $\mathbf{x}_0$ is assigned to the class y_0 which has a *weighted majority* among its k nearest neighbours, namely

$$y_0 = \arg\max_y \sum_{i=1}^{k} \omega_i I(y_i = y).$$

Each of the k members $\mathbf{x}_i$ of the neighbourhood of $\mathbf{x}_0$ is assigned a weight ω_i which is inversely proportional to its distance $d = d(\mathbf{x}_0, \mathbf{x}_i)$ from $\mathbf{x}_0$ and which is obtained by means of a *weighting function* or *kernel* $K_\lambda(\mathbf{x}_0, \mathbf{x}_i)$ [12]:

$$\omega_i = K_\lambda(\mathbf{x}_0, \mathbf{x}_i) = G\left(\frac{d(\mathbf{x}_0, \mathbf{x}_i)}{\lambda}\right). \tag{15}$$

Kernels are at the basis of the *Parzen density estimation method* [12]; in that context, the *smoothing parameter* or *bandwidth* λ dictates the width of the window considered to perform the estimation. A large λ implies lower variance (averages over more observations) but higher bias (we essentially assume the true function is constant within the window).

Here $G(\cdot)$ can be any function with maximum in $d = d(\mathbf{x}, \mathbf{y}) = 0$ and values that get smaller with growing value of d. Thus the following properties must hold [13]:

1. $G(d) \geq 0$ for all $d \in \mathbb{R}$;
2. $G(d)$ gets its maximum for $d = 0$;
3. $G(d)$ descends monotonically for $d \to \infty$;

In the one-dimensional case, one popular kernel is obtained by using the Gaussian density function $\phi(t)$ as $G(\cdot)$, with the standard deviation playing the role of the parameter λ. In $\mathbb{R}^p$, with $p > 1$, the natural generalization is represented by

$$K_\lambda(\mathbf{x}_0, \mathbf{x}_i) = \frac{1}{\sqrt{2\pi}\lambda} \exp\left\{-\frac{1}{2}\left(\frac{\|\mathbf{x}_0 - \mathbf{x}_i\|}{\lambda}\right)^2\right\},$$

which is the kernel we adopt in our method.

Although the kernel used is in a sense a parameter of wkNN, experience has shown that the choice of kernel (apart from the the *rectangular kernel*, which gives equal weights to all neighbours) is not crucial [12].

In equation (15) it is assumed that λ is a fixed value over the whole of the space of data samples. The optimal value of λ may be location-dependent, giving a large value in regions where the data samples are sparse and a small value where the data samples are densely packed. One solution is represented by the use of *adaptive windows*, where λ depends on the location of the sample in the data space. Let $h_\lambda(x_0)$ be a width function (indexed by λ) which determines the width of the neighbourhood at x_0. Then we have

$$K_\lambda(x_0, x_i) = G\left(\frac{d(x_0, x_i)}{h_\lambda(x_0)}\right)$$

As kernels are used to compute weights in wkNN, we set $h_\lambda(x_0)$ equal to the distance $d(x_0, x_{k+1})$ of x_0 from the first neighbour x_{k+1} that is not taken into consideration [12, 13].

4.3 Using wkNN for Probability Estimation

We adopted the weighted k-nearest neighbour principle to estimate class probabilities for each distinct observation $\mathbf{x} \in Z$ as follows: let $N_k(\mathbf{x})$ be the set of indices in D of all occurences of the k attribute vectors in Z closest to $\mathbf{x}$. Note that $N_k(\mathbf{x})$ may contain more than k elements if some attribute vectors occur multiple times in D. Then

$$\hat{P}(y|\mathbf{x}) = \frac{\sum_{i \in N_k(\mathbf{x})} \omega_i I(y_i = y)}{\sum_{i \in N_k(\mathbf{x})} \omega_i}, \qquad y = 1, \ldots, q, \tag{16}$$

It should be noted that $\mathbf{x}$ is included in its own neighbourhood and its occurrences have a relatively large weight ω_i in (16).

In the case of MOCA, the adoption of this new probability estimator has an impact on the computation of the MOCA estimator not only in terms of the probability estimates that the antitonic regression is performed on but also on the weights used, which are now equal to the cardinality of $N_k(\mathbf{x})$ for each $\mathbf{x} \in Z$. Note that if $k = 1$, then equation (16) produces the maximum likelihood estimates used in standard OSDL and MOCA.

The estimator presented in this section is analogous to the one adopted in [13], where the estimates obtained are used to perform ordinal classification (without monotonicity constraints) by predicting the median.

5 Experiments

We performed a series of experiments on a several real-world datasets in order to determine whether and to what extent MOCA and OSDL would benefit from the new wkNN probability estimator. The results were measured in terms of the average L_1 error rate achieved by the two algorithms.

5.1 Datasets

We selected a number of datasets where monotonicity constraints are likely to apply. We used the KC1, KC4, PC3, and PC4 datasets from the NASA Metrics Data Program [16], the Acceptance/Rejection, Employee Selection, Lecturers Evaluation and Social Workers Decisions datasets from A. Ben-David [5], the Windsor Housing dataset [2], as well as several datasets from the UCI Machine Learning Repository [3]. Table 2 lists all the datasets used.

Table 2. Charasterics of datasets used in the experiments

Dataset	cardinality	#attributes	#labels
Australian Credit	690	14	2
Auto MPG	392	7	4
Boston Housing	506	13	4
Car Evaluation	1728	6	4
Empoyee Rej/Acc	1000	4	9
Employee selection	488	4	9
Haberman survival	306	3	2
KC1	2107	21	3
KC4	122	4	6
Lecturers evaluation	1000	4	5
CPU Performance	209	6	4
PC3	320	15	5
PC4	356	16	6
Pima Indians	768	8	2
Social Workers Decisions	1000	10	4
Windsor Housing	546	11	4

5.2 Dataset Preprocessing

For datasets with a numeric response that is not a count (Auto MPG, Boston Housing, CPU Performance, and Windsor Housing) we discretized the response values into four separate intervals, each interval containing roughly the same number of observations.

For all datasets from the NASA Metrics Data Program the attribute `ERROR_COUNT` was used as the response. All attributes that contained missing values were removed. Furthermore, the attribute `MODULE` was removed because it is a unique identifier of the module and the `ERROR_DENSITY` was removed because it is a function of the response variable. Furthermore, attributes with zero variance were removed from the dataset.

5.3 Experimental results

Each of the datasets was randomly divided into two parts, a training set (containing roughly two thirds of the data) and a validation set. The training set was used to determine the optimal values for k and α in MOCA and OSDL through 10-fold cross validation. We started with $k = 1$ and incremented its value by one until the difference of the average L_1 error between two consecutive iterations for both classifiers was less than or equal to 1^{-6}. For each value of k we determined the the optimal α in $\{0, 0.25, 0.5, 0.75, 1\}$. Once the optimal parameter values were determined, they were used to train both algorithms on the complete training set and then to test them on the validation set. We then performed a paired t-test of the L_1 errors on the validation set to determine whether observed differences were significant. Table 3 lists all the results.

Table 3. Experimental results. The first four columns contain average L_1 errors on the validation set. The final two columns contain p-values. The penultimate column compairs smoothed MOCA to standard MOCA. The final column compairs smoothed OSDL to standard OSDL.

Dataset	MOCA wkNN	OSDL wkNN	MOCA MLE	OSDL MLE	1. vs. 3.	2. vs. 4.
Australian Credit	0.1304	0.1130	0.1348	0.3565	0.3184	0
Auto MPG	0.2977	0.2977	0.2977	0.2977	–	–
Boston Housing	0.5030	0.4675	0.4675	0.5207	0.4929	0.2085
Car Evaluation	0.0625	0.0556	0.0625	0.0556	–	–
Empoyee Rej/Acc	1.2006	1.2066	1.2814	1.2814	0.0247	0.0445
Employee selection	0.3620	0.3742	0.3620	0.4110	1	0.2018
Haberman survival	0.3529	0.3431	0.3529	0.3431	–	–
KC1	0.1863	0.1977	0.1863	0.3940	1	0
KC4	0.8095	0.8095	0.8571	0.8571	0.4208	0.5336
Lecturers evaluation	0.4162	0.4162	0.4102	0.4102	0.8060	0.8060
CPU Performance	0.3571	0.3286	0.3571	0.3571	–	0.5310
PC3	0.1228	0.1228	0.1228	0.1228	–	–
PC4	0.1872	0.1872	0.1872	0.1872	–	–
Pima Indians	0.3008	0.3086	0.3008	0.3086	–	–
Social Workers	0.5359	0.5479	0.5060	0.4940	0.1492	0.0092
Windsor Housing	0.5604	0.5220	0.5604	0.6044	–	0.0249

We first check whether smoothing actually improves the classifiers. Comparing standard OSDL against smoothed OSDL we observe that the latter is signifcantly better (at $\alpha = 0.05$) four times, whereas it is significantly worse one time (for Social Workers Decisions). Furthermore, the smoothed version almost never has higher estimated error (Lecturer Evaluation and Social Workers Decisions being the two exceptions).

Comparing standard MOCA against smoothed MOCA, we observe that the latter is significantly better only once (on Employee Rejection). All other observed differences are not significant.

Table 4. Experimental results for balanced and double balanced OSDL. The first four columns contain average L_1 errors on the validation set. The final two columns contain p-values. The penultimate column compairs smoothed balanced OSDL to standard balanced OSDL. The final column compairs smoothed double balanced OSDL to standard double balanced OSDL.

Dataset	BOSDL wkNN	BOSDL MLE	DBOSDL wkNN	DBOSDL MLE	1. vs. 2.	3. vs. 4.
Australian Credit	0.3565	0.3565	0.3565	0.3565	–	–
Auto MPG	0.2977	0.2977	0.2977	0.2977	–	–
Boston Housing	0.5207	0.5207	0.5207	0.5207	–	–
Car Evaluation	0.0556	0.0556	0.0556	0.0556	–	–
Empoyee Rej/Acc	1.8533	1.9760	1.8533	1.9760	0.1065	0.1065
Employee selection	0.5092	0.5092	0.5092	0.5092	–	–
Haberman survival	0.3431	0.3431	0.3431	0.3431	–	–
KC1	0.1892	0.3030	0.3883	0.3940	0	0.2061
KC4	0.7619	0.8571	0.7857	0.8571	0.1031	0.1829
Lecturers evaluation	0.9251	0.9251	0.9251	0.9251	–	–
CPU Performance	0.4143	0.4143	0.4143	0.4143	–	–
PC3	0.1228	0.1228	0.1228	0.1228	–	–
PC4	0.1872	0.1872	0.1872	0.1872	–	–
Pima Indians	0.2930	0.2930	0.2930	0.2930	–	–
Social Workers	0.6617	0.6557	0.6617	0.6437	0.8212	0.4863
Windsor Housing	0.6044	0.6044	0.6044	0.6044	–	–

In table 4 the effect of smoothing on balanced and double balanced OSDL is investigated. We conclude that smoothing doesn't have much effect in either case: only one significant improvement is found (for KC1). Furthermore, comparing table 3 and table 4, we observe that constant interpolation OSDL with smoothing tends to outperform its balanced and double balanced counterparts.

6 Conclusion

We have discussed two related methods for nonparametric monotone classification: OSDL and MOCA. The basic class probability estimates used by these algorithms are typically based on very few observations. Therefore we conjectured that they both have a tendency to overfit on the training sample. We have proposed a weighted k nearest neighbour approach to smoothing the basic estimates.

The experiments have shown that smoothing is beneficial for OSDL: the predictive performance was significantly better on a number of datasets, and almost never worse. For MOCA, smoothing seems to have much less effect. This is probably due to the fact that the isotonic regression already smooths the basic estimates by averaging them in case of order reversals. Hence, MOCA is already quite competitive for $k = 1$.

The more sophisticated interpolation schemes of OSDL (balanced and double balanced) do not seem to lead to an improvement over the constant interpolation version on the datasets considered.

References

1. E.A. Altendorf, A.C. Restificar, and T.G. Dietterich. Learning from sparse data by exploiting monotonicity constraints. In F. Bacchus and T. Jaakkola, editors, *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 18–25. AUAI Press, 2005.
2. P.M. Anglin and R. Gençay. Semiparametric estimation of a hedonic price function. *Journal of Applied Econometrics*, 11(6):633–648, 1996.
3. A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.
4. N. Barile and A. Feelders. Nonparametric monotone classification with MOCA. In F. Giannotti, editor, *Proceedings of the Eighth IEEE International Conference on Data Mining (ICDM 2008)*, pages 731–736. IEEE Computer Society, 2008.
5. A. Ben-David, L. Sterling, and Y. Pao. Learning and classification of monotonic ordinal concepts. *Computational Intelligence*, 5:45–49, 1989.
6. Arie Ben-David. Monotonicity maintenance in information-theoretic machine learning algorithms. *Machine Learning*, 19:29–43, 1995.
7. K. Cao-Van. *Supervised ranking, from semantics to algorithms.* PhD thesis, Universiteit Gent, 2003.
8. K. Dembczynski, W. Kotlowski, and R. Slowinski. Ensemble of decision rules for ordinal classification with monotonicity constraints. In *Rough Sets and Knowledge Technology*, volume 5009 of *LNCS*, pages 260–267. Springer, 2008.
9. M.J. Druzdzel and L.C. van der Gaag. Elicitation of probabilities for belief networks: combining qualitative and quantitative information. In P. Besnard and S. Hanks, editors, *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 141–148. Morgan Kaufmann, 1995.
10. A. Feelders and M. Pardoel. Pruning for monotone classification trees. In M.R. Berthold, H-J. Lenz, E. Bradley, R. Kruse, and C. Borgelt, editors, *Advances in Intelligent Data Analysis V*, volume 2810 of *LNCS*, pages 1–12. Springer, 2003.
11. A. Feelders and L. van der Gaag. Learning Bayesian network parameters with prior knowledge about context-specific qualitative influences. In F. Bacchus and T. Jaakkola, editors, *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 193–200. AUAI Press, 2005.
12. R. Hastie, T.and Tibshirani and J. Friedman. *The Elements of Statistical Learning.* Springer, second edition, 2009.
13. K. Hechenbichler and K. Schliep. Weighted k-nearest-neighbor techniques and ordinal classification. Discussion Paper 399, Collaborative Research Center (SFB) 386 Statistical Analysis of discrete structures - Applications in Biometrics and Econometrics, 2004.

14. W. Kotlowski and R. Slowinski. Statistical approach to ordinal classification with monotonicity constraints. In *ECML PKDD 2008 Workshop on Preference Learning*, 2008.
15. S. Lievens, B. De Baets, and K. Cao-Van. A probabilistic framework for the design of instance-based supervised ranking algorithms in an ordinal setting. *Annals of Operations Research*, 163:115–142, 2008.
16. J. Long. NASA metrics data program [http://mdp.ivv.nasa.gov/repository.html]. 2008.
17. W.L. Maxwell and J.A. Muckstadt. Establishing consistent and realistic reorder intervals in production-distribution systems. *Operations Research*, 33(6):1316–1341, 1985.
18. M.A. Meyer and J.M. Booker. *Eliciting and Analyzing Expert Judgment: A Practical Guide*. Series on Statistics and Applied Probability. ASA-SIAM, 2001.
19. R. Potharst and J.C. Bioch. Decision trees for ordinal classification. *Intelligent Data Analysis*, 4(2):97–112, 2000.
20. J. Sill. Monotonic networks. In *Advances in neural information processing systems*, NIPS (Vol. 10), pages 661–667, 1998.
21. M. Velikova, H. Daniels, and M. Samulski. Partially monotone networks applied to breast cancer detection on mammograms. In *Proceedings of the 18th International Conference on Artificial Neural Networks (ICANN)*, volume 5163 of *LNCS*, pages 917–926. Springer, 2008.
22. M.P. Wellman. Fundamental concepts of qualitative probabilistic networks. *Artificial Intelligence*, 44:257–303, 1990.