Exploitation of Pairwise Class Distances for Ordinal Classification

J. Sánchez-Monedero[1], Pedro A. Gutiérrez[1], Peter Tiňo[2], C. Hervás-Martínez[1]

[1]Department of Computer Science and Numerical Analysis, University of Córdoba, Córdoba 14071, Spain.

[2]School of Computer Science, The University of Birmingham, Birmingham B15 2TT, United Kingdom.

Keywords: ordinal classification, ordinal regression, support vector machines, threshold model, latent variable

Abstract

Ordinal classification refers to classification problems in which the classes have a natural order imposed on them because of the nature of the concept studied. Some ordinal classification approaches perform a projection from the input space to 1-dimensional

(latent) space that is partitioned into a sequence of intervals (one for each class). Class identity of a novel input pattern is then decided based on the interval its projection falls into. This projection is trained only indirectly as part of the overall model fitting. As with any latent model fitting, direct construction hints one may have about the desired form of the latent model can prove very useful for obtaining high quality models. The key idea of this paper is to construct such a projection model directly, using insights about the class distribution obtained from pairwise distance calculations. The proposed approach is extensively evaluated with eight nominal and ordinal classifiers methods, ten real world ordinal classification datasets, and four different performance measures. The new methodology obtained the best results in average ranking when considering three of the performance metrics, although significant differences are found only for some of the methods. Also, after observing other methods internal behaviour in the latent space, we conclude that the internal projection do not fully reflect the intra-class behaviour of the patterns. Our method is intrinsically simple, intuitive and easily understandable, yet, highly competitive with state-of-the-art approaches to ordinal classification.

1 Introduction

Ordinal classification or ordinal regression is a supervised learning problem of predicting categories that have an ordered arrangement. When the problem is really exhibiting an ordinal nature, it is expected that this order is also present in the data input space (Hühn and Hüllermeier, 2008). The samples are labelled by a set of ranks with an

ordering amongst the categories. In contrast to nominal classification, there is an ordinal relationship throughout the categories and it is different from regression in that the number of ranks is finite and exact amounts of difference between ranks are not defined. In this way, ordinal classification lies somewhere between nominal classification and regression.

Ordinal regression should not be confused with sorting or ranking. Sorting is related to ranking all samples in the test set, with a total order. Ranking is related to ranking with a relative order of samples and a limited number of ranks. Of course, ordinal regression can be used to rank samples, but its objective is to obtain a good accuracy, and, at the same time, a good ranking.

Ordinal classification problems are important, since they are common in our everyday life where many problems require classification of items into naturally ordered classes. Examples of these problems are the teaching assistant evaluation (Lim et al., 2000), car insurance risk rating (Kibler et al., 1989), pasture production (Barker, 1995), preference learning (Arens, 2010), breast cancer conservative treatment (Cardoso et al., 2005), wind forescasting (Gutiérrez et al., 2013) or credit rating (Kim and Ahn, 2012).

Variety of approaches have been proposed for ordinal classification. For example, Raykar et al. (2008) learns ranking functions in the context of ordinal regression and collaborative filtering datasets. Kramer et al. (2010) map the ordinal scale by assigning numerical values and then apply a regression tree model. The main problem with this simple approach is the assignment of a numerical value corresponding to each class, without a principled way of deciding the true metric distances between the ordinal scales. Also, representing all patterns in a class by the same value may not reflect

the relationships among the patterns in a natural way. In this paper we propose that the numerical values associated with different patterns may differ (even within the same class), and, most importantly, the value for each individual pattern is decided based of its relative localization in the input space.

Other simple alternative that appeared in the literature tried to impose the ordinal structure through the use of cost-sensitive classification, where standard (nominal) classifiers are made aware of ordinal information through penalizing the misclassification error, commonly selecting a cost equal to the absolute deviation between the actual and the predicted ranks (Kotsiantis and Pintelas, 2004). This is suitable when the knowledge about the problem is sufficient to completely define a cost matrix. However, when this is not possible, this approach is making an important assumption about the distances between the adjacent labels, all of them being equal, which may not be appropriate.

The third direct alternative suggested in the literature is to transform the ordinal classification problem into a nested binary classification one (Frank and Hall, 2001; Waegeman and Boullart, 2009), and then to combine the resulting classifier predictions to obtain the final decision. It is clear that ordinal information allows ranks to be compared. For a given rank k, an associated question could be "is the rank of pattern $\mathbf{x}$ greater than k?". This question is exactly a binary classification problem, and ordinal classification can be solved by approaching each binary classification problem independently and combining the binary outputs to a rank (Frank and Hall, 2001). Other alternative (Waegeman and Boullart, 2009) imposes explicit weights over the patterns of each binary system in such a way that errors on training objects are penalized proportionally to the absolute difference between their rank and k. Binarization of ordinal

regression problems can also be tackled from augmented binary classification perspective, i.e. binary problems are not solved independently, but a single binary classifier is constructed for all the subproblems. For example, Cardoso and Pinto da Costa (2007) add additional dimensions and replicate the data points through what is known as the data replication method. This augmented space is used to construct a binary classifier and the projection onto the original one results in an ordinal classifier. A very interesting framework in this direction is that proposed by Li and Lin (2007); Lin and Li (2012), reduction from cost-sensitive ordinal ranking to weighted binary classification (RED), which is able to reformulate the problem as a binary problem by using a matrix for extension of the original samples, a weighting scheme and a V-shaped cost matrix. An attractive feature of this framework is that it unifies many existing ordinal ranking algorithms, such as perceptron ranking (Crammer and Singer, 2005) and support vector ordinal regression (Chu and Keerthi, 2007). Recently, in (Fouad and Tiňo, 2012) the Learning Vector Quantization (LVQ) is adapted to the ordinal case in the context of prototype based learning. In that work the order information is utilized to select class prototypes to be adapted, and to improve the prototype update process.

Vast majority of proposals addressing ordinal classification can be grouped under the umbrella of *threshold methods* (Verwaeren et al., 2012). These methods assume that ordinal response is a coarsely measured latent continuous variable, and model it as real intervals in one dimension. Based on this assumption, the algorithms seek a direction onto which the samples are projected and a set of thresholds that partition the direction into consecutive intervals representing ordinal categories (McCullagh, 1980; Verwaeren et al., 2012; Herbrich et al., 2000; Crammer and Singer, 2001; Chu and Keerthi, 2005).

Proportional Odds Model (POM) (McCullagh, 1980) is a standard statistical approach in this direction, where the latent variable is modelled by using a linear combination of the inputs and a probabilistic distribution is assumed for the patterns projected by this function. Crammer and Singer (2001) generalized the online perceptron algorithm with multiple thresholds to perform ordinal ranking. Support Vector Machines (SVMs) (Cortes and Vapnik, 1995; Vapnik, 1999) were also adapted for ordinal regression, first by the large-margin algorithm of Herbrich et al. (2000). The main drawback of this first proposal was that the problem size was a quadratic function of the training data size. A related more efficient approach was presented by Shashua and Levin (2002), who excluded the inequality constraints on the thresholds. However this can result in non desirable solutions because the absence of constrains can lead to difficulties in imposing order on thresholds. Chu and Keerthi (2005) explicitly and implicitly included the constraints in the model formulation (Support Vector for Ordinal Regression, SVOR), deriving the associated dual problem and the optimality conditions. From another perspective, discriminant learning has been adapted to the ordinal set-up by (apart from maximizing between-class distance and minimizing within-class distance) trying to minimize distance separation between projected patterns of consecutive classes (Kernel Discriminant Learning for Ordinal Regression, KDLOR) (Sun et al., 2010). Finally, threshold models have also been estimated by using a Bayesian framework (Gaussian Processes for Ordinal Regression, GPOR) (Chu and Ghahramani, 2005), where the latent function is modelled using Gaussian Processes and then all the parameters are estimated by Maximum Likelihood optimization.

While threshold approaches offer an interesting perspective on the problem of ordi-

nal classification, they learn the projection from the input space onto the 1-dimensional latent space only indirectly as part of the overall model fitting. As with any latent model fitting, direct construction hints one may have about the desired form of the latent model can prove very useful for obtaining high quality models. The key idea of this paper is to construct such a projection model directly, using insights about the class distribution obtained from pairwise distance calculations. Indeed, our motivation stems from the fact that the order information should also be present in the data input space and it could be interesting to take advantage from it to construct an useful variable for ordering the patterns using the ordinal scale. Additionally, regression is clearly the most natural way to approximate this continuous variable. As a result, we propose to construct the ordinal classifier in two stages: 1) the input data is first projected into a one dimensional variable by considering the relative position of the patterns in the input space, and, 2) a standard regression algorithm is applied to learn a function to predict new values of this derived variable.

The main contribution of the current work is the projection onto a one dimensional variable, which is done by a guided projection process. This process exploits the ordinal space distribution of patterns in the input space. A measure of how *'well'* a pattern is located within its corresponding class region is defined by considering the distances between patterns of the adjacent classes in the ordinal scale. Then, a projection interval is defined for each class, and the centres of those intervals (for non-boundary classes) are associated with the *'best'* located patterns for the corresponding classes (quantified by the measure mentioned above). For the boundary classes (first and last in the class order), the extreme end points of their projection intervals are associated with the most

separated patterns of those classes. All the other patterns are assigned proportional positions in their corresponding class intervals, again according to their *'goodness'* values expressing how *'well'* a pattern is located within its class. We refer to this projection as *Pairwise Class Distances* (PCD) based projection. The behaviour of this projection is evaluated over synthetic datasets, showing an intuitive response and a good ability to separate adjacent classes even in non-linear settings.

Once the mapping is done, our framework allows to design effective ordinal ranking algorithms based on well-tuned regression approaches. The final classifier constructed by combining PCD and a regressor is called *Pairwise Class Distances Ordinal Classifier* (PCDOC). In this contribution, PCDOC is implemented using ϵ-Support Vector Regression (ϵ-SVR) (Schölkopf and Smola, 2001; Vapnik, 1999) as the base regressor, although any other properly handled regression method could be used.

We carry out an extensive set of experiments on ten real world ordinal regression datasets, comparing our approach with eight state-of-the-art methods. Our method, though simple, holds out very well. Under four complementary performance metrics, the proposed method obtained the best mean ranking for three of the four metrics.

The rest of the paper is organized as follows. Section 2 introduces the ordinal classification problem and performance metrics we use to evaluate the ordinal classifiers. Section 3 explains the proposed data projection method and the classification algorithm. It also evaluates the behaviour of the projection using two synthetic datasets, and the performance of the classification algorithm under situations that may hamper classification. The following section presents the experimental design, datasets, alternative ordinal classification methods that will be compared with our approach and discusses

the experimental results. Finally, the last section sums up key conclusions and points to future work.

2 Ordinal classification

This section briefly introduces the mathematical notation and the ordinal classification performance metrics. Finally, the last subsection includes the threshold model formulation.

2.1 Problem formulation

In an ordinal classification problem, the purpose is to learn a mapping ϕ from an input space $\mathbb{X}$ to a finite set $\mathcal{C} = \{C_1, C_2, \ldots, C_Q\}$ containing Q labels, where the label set has an order relation $C_1 \prec C_2 \prec \ldots \prec C_Q$ imposed on it. The symbol $\prec$ denotes the ordering between different ranks. A rank for the ordinal label can be defined as $\mathcal{O}(\mathcal{C}_q) = q$. Each pattern is represented by a K-dimensional feature vector $\mathbf{x} \in \mathbb{X} \subseteq \mathbb{R}^K$ and a class label $y \in \mathcal{C}$. The training dataset $\mathbf{T}$ is composed of N patterns $\mathbf{T} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{X},\ y_i \in \mathcal{C}, i = 1, \ldots, N\}$, with $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iK})$.

Given the above definitions, an ordinal classifier should be constructed taking into account two goals. First, the nature of the problem implies that the class order is somehow related to the distribution of patterns in the space of attributes $\mathbb{X}$, and also to the topological distribution of the classes. Therefore the classifier must exploit this a priori knowledge about the input space (Hühn and Hüllermeier, 2008). Second, when evaluating an ordinal classifier, the performance metrics must consider the order of the classes

so that misclassifications between adjacent classes should be considered less important than the ones between non-adjacent classes, more separated in the class order. For example, given an ordinal dataset of weather prediction $\{$*Very cold*, *Cold*, *Mild*, *Hot*, *Very hot*$\}$ with the natural order between classes $\{$*Very cold* $\prec$ *Cold* $\prec$ *Mild* $\prec$ *Hot* $\prec$ *Very hot*$\}$, it is straightforward to think that predicting class *Hot* when the real class is *Cold* represents a more severe error than that associated with a *Very cold* prediction. Thus, specialized measures are needed for evaluating ordinal classifiers performance (Pinto da Costa et al., 2008)(Cruz-Ramírez et al., 2011).

2.2 Ordinal classification performance metrics

In this work, we utilize four evaluation metrics quantifying the accuracy of N predicted ordinal labels for a given dataset $\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N\}$, with respect to the true targets $\{y_1, y_2, \ldots, y_N\}$:

1. Acc: the accuracy (Acc), also known as Correct Classification Rate[1], is the rate of correctly classified patterns:

$$Acc = \frac{1}{N}\sum_{i=1}^{N}[\![\hat{y}_i = y_i]\!],$$

where y_i is the true rank, $\hat{y}_i$ is the predicted rank and $[\![c]\!]$ is the indicator function, being equal to 1 if c is true, and to 0 otherwise. Acc values range from 0 to 1 and they represent a global performance on the classification task. Although Acc is widely used in classification tasks, is it not suitable for some type of problems, such as imbalanced datasets (Sánchez-Monedero et al., 2011) (very

[1] *Acc* is referred as Mean Zero-One Error when expressed as an error.

different number of patterns for each class) or ordinal datasets (Baccianella et al., 2009).

2. MAE: The Mean Absolute Error (MAE) is the average absolute deviation of the predicted ranks from the true ranks (Baccianella et al., 2009):

$$MAE = \frac{1}{N}\sum_{i=1}^{N} e(\mathbf{x}_i),$$

where $e(\mathbf{x}_i) = |\mathcal{O}(y_i) - \mathcal{O}(\hat{y}_i)|$. The MAE values range from 0 to $Q-1$. Since Acc does not reflect the category order, MAE is typically used in the ordinal classification literature together with Acc (Pinto da Costa et al., 2008; Agresti, 1984; Waegeman and De Baets, 2011; Chu and Keerthi, 2007; Chu and Ghahramani, 2005; Li and Lin, 2007). However, neither Acc, nor MAE are suitable for problems with imbalanced classes. This is rectified e.g. in the *average* MAE ($AMAE$) (Baccianella et al., 2009) measuring the mean performance of the classifier across all classes.

3. $AMAE$: This measure evaluates the mean of the MAEs across classes (Baccianella et al., 2009). It has been proposed as a more robust alternative to MAE for imbalanced datasets – a very common situation in ordinal classification, where extreme classes (associated with rare situations) tend to be less populated.

$$AMAE = \frac{1}{Q}\sum_{j=1}^{Q} MAE_j = \frac{1}{Q}\sum_{j=1}^{Q}\frac{1}{n_j}\sum_{i=1}^{n_j} e(\mathbf{x}_i),$$

where $AMAE$ values range from 0 to $Q-1$ and n_j is the number of patterns in class j.

4. τ_b: The Kendall's τ_b is a statistic used to measure the association between two measured quantities. Specifically, it is a measure of the rank correlation (Kendall,

1962):

$$\tau_b = \frac{\sum_{i,j=1}^{N} \hat{c}_{ij} c_{ij}}{\sqrt{\sum_{i,j=1}^{N} \hat{c}_{ij}^2 \sum_{i,j=1}^{N} c_{ij}^2}},$$

where $\hat{c}_{ij}$ is $+1$ if $\hat{y}_i$ is greater than (in the ordinal scale) $\hat{y}_j$, 0 if $\hat{y}_i$ and $\hat{y}_j$ are the same, and -1 if $\hat{y}_i$ is lower than $\hat{y}_j$, and the same for c_{ij} (using y_i and y_j). τ_b values range from -1 (maximum disagreement between the prediction and the true label), to 0 (no correlation between them) and to 1 (maximum agreement). τ_b has been advocated as a better measure for ordinal variables because it is independent of the values used to represent classes (Cardoso and Sousa, 2011) since it works directly on the set of pairs corresponding to different observations. One may argue that shifting the predictions one class would will keep the same τ_b value whereas the quality of the ordinal classification is lower. However, note that since there is a finite number of classes, shifting all predictions by one class will have detrimental effect in the boundary classes and so would substantially decrease the performance, even as measured by τ_b. As a consequence, τ_b is an interesting measure for ordinal classification but should be used in conjunction with other ones.

2.3 Latent variable modelling for ordinal classification

Latent variable models or threshold models are probably the most important type of ordinal regression models. These models consider the ordinal scale as the result of coarse measurements of a continuous variable, called the latent variable. It is typically assumed that the latent variable is difficult to measure or cannot be observed itself (Verwaeren et al., 2012). The threshold model can be represented with the following

general expression:

$$f(\mathbf{x}, \boldsymbol{\theta}) = \begin{cases} C_1, & \text{if } g(\mathbf{x}) \leq \theta_1, \\ C_2, & \text{if } \theta_1 < g(\mathbf{x}) \leq \theta_2, \\ \vdots & \\ C_Q, & \text{if } g(\mathbf{x}) > \theta_{Q-1}, \end{cases} \tag{1}$$

where $g : \mathbb{X} \to \mathbb{R}$ is the function that projects data space onto the 1-dimensional latent space $\mathcal{Z} \subseteq \mathbb{R}$ and $\theta_1 < \ldots < \theta_{Q-1}$ are the thresholds that divide the space into ordered intervals corresponding to the classes.

In our proposal, it is assumed that a model $\phi : \mathbb{X} \to \mathcal{Z}$ can be found that links data items $\mathbf{x} \in \mathbb{X}$ with their latent space representation $\phi(\mathbf{x}) \in \mathcal{Z}$. We place our proposal in the context of latent variable models for ordinal classification because of its similarity to these models. In contrast to other models employing a one dimensional latent space, e.g. POM (McCullagh, 1980), we do not consider variable thresholds, but impose fixed values for $\boldsymbol{\theta}$. However, suitable dimensionality reduction is given due attention: first, by trying to exploit the ordinal structure of the space $\mathbb{X}$, and second we explicitly put external pressure on the margins between the classes in $\mathcal{Z}$ (see Section 3.2).

3 Proposed method

Our approach is different from the previous ones in that it does not implicitly learn latent representations of the training inputs. Instead, we impose how training inputs $\mathbf{x}_i$ are going to be represented through $z_i = \phi(\mathbf{x}_i)$. Then, this representation is generalized to the whole input space by training a regressor on the $(\mathbf{x}_i, z_i)$ pairs, resulting in a projection function $g : \mathbb{X} \to \mathcal{Z}$. To ease the presentation, we will sometimes write

training input patterns $\mathbf{x}$ as $\mathbf{x}^{(q)}$ to explicitly reflect their class label rank q (i.e. the class label of $\mathbf{x}$ is C_q).

3.1 Pairwise Class Distance (PCD) projection

To describe the Pairwise Class Distance (PCD) projection, first, we define a measure $w_{\mathbf{x}^{(q)}}$ of *"how well"* a pattern $\mathbf{x}^{(q)}$ is placed within other instances of class C_q, by considering its Euclidean distances to the patterns in adjacent classes. This is done on the assumption of ordinal pattern distribution in the input space $\mathbb{X}$. For calculating this measure, the minimum distances of a pattern $\mathbf{x}_i^{(q)}$ to patterns in the previous and next classes, C_{q-1} and C_{q+1}, respectively, are used. The minimum distance to the previous/next class is

$$\kappa(\mathbf{x}_i^{(q)}, q \pm 1) = \min_{\mathbf{x}_j^{(q\pm1)}} \left\{ ||\mathbf{x}_i^{(q)} - \mathbf{x}_j^{(q\pm1)}|| \right\}, \tag{2}$$

where $||\mathbf{x} - \mathbf{x}'||$ is the Euclidean distance between $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^K$. Then,

$$w_{\mathbf{x}_i^{(q)}} = \begin{cases} \dfrac{\kappa(\mathbf{x}_i^{(q)}, q+1)}{\max_{\mathbf{x}_n^{(q)}} \left\{ \kappa(\mathbf{x}_n^{(q)}, q+1) \right\}}, \text{ if } q = 1, \\ \dfrac{\kappa(\mathbf{x}_i^{(q)}, q-1) + \kappa(\mathbf{x}_i^{(q)}, q+1)}{\max_{\mathbf{x}_n^{(q)}} \left\{ \kappa(\mathbf{x}_n^{(q)}, q-1) + \kappa(\mathbf{x}_n^{(q)}, q+1) \right\}}, \text{ if } q \in \{2, \ldots, Q-1\}, \\ \dfrac{\kappa(\mathbf{x}_i^{(q)}, q-1)}{\max_{\mathbf{x}_n^{(q)}} \left\{ \kappa(\mathbf{x}_n^{(q)}, q-1) \right\}}, \text{ if } q = Q, \end{cases} \tag{3}$$

where the sum of the minimum distances of a pattern with respect to adjacent classes is normalized across all patterns of the class, so that $w_{\mathbf{x}_i^{(q)}}$ has a maximum value of 1.

Figure 1 shows the idea of minimum distances for each pattern with respect to the patterns of the adjacent classes. In this figure, patterns of the second class are considered. The example illustrates how the $w_{\mathbf{x}^{(2)}}$ value is obtained for the pattern $\mathbf{x}^{(2)}$ marked

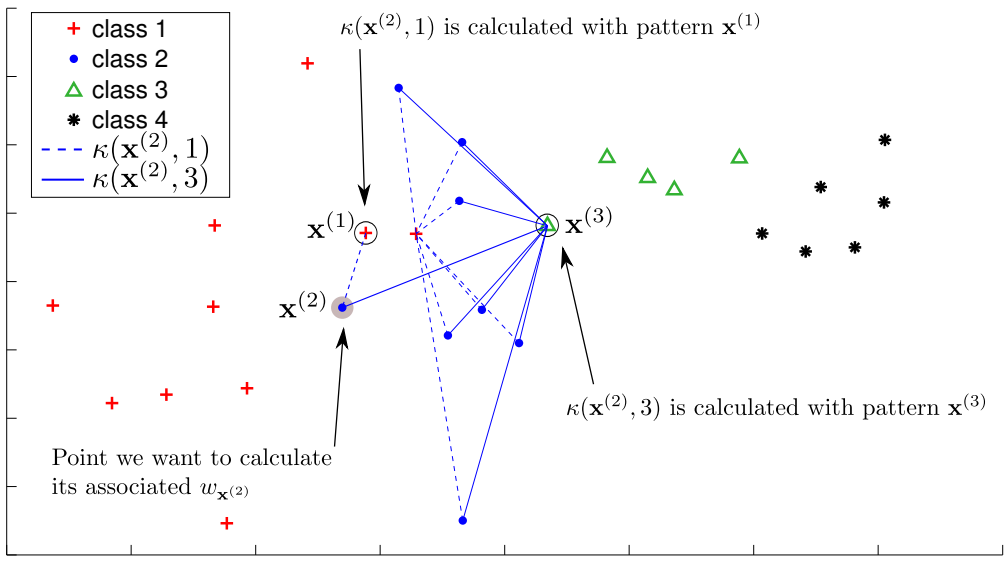


Figure 1: Illustration of the idea of minimum Pairwise Class Distances. All the minimum distances of patterns of class C_2 regarding patterns of adjacent classes are painted with lines. $\mathbf{x}^{(2)}$ is the point we want to calculate its associated $w_{\mathbf{x}^{(2)}}$.

with a circle. For distances between $\mathbf{x}^{(2)}$ and class 1 patterns, the item $\mathbf{x}^{(1)}$ has the minimum distance, so $\kappa(\mathbf{x}^{(2)}, 1)$ is calculated by using this pattern. For distances between $\mathbf{x}^{(2)}$ and class 3 patterns, $\kappa(\mathbf{x}^{(2)}, 3)$ is the minimum distance between $\mathbf{x}^{(2)}$ and $\mathbf{x}^{(3)}$.

By using $w_{\mathbf{x}_i^{(q)}}$, we can derive a latent variable value $z_i \in \mathcal{Z}$. Before continuing, thresholds must be defined in order to stablish the intervals on $\mathcal{Z}$ which correspond to each class, so that calculated values for z_i may be positioned on the proper interval. Also, predicted values $\hat{z}_i$ of unseen data would be classified in different classes according to these thresholds (see Subsection 3.3), in a similar way to any other threshold model. For the sake of simplicity, $\mathcal{Z}$ is defined between 0 and 1, and the thresholds are

positioned in the uniform manner[2] :

$$\theta = \{\theta_1, \theta_2, \ldots, \theta_Q\} = \{1/Q, 2/Q, \ldots, 1\}\,. \tag{4}$$

Considering θ, the centres $c_q \in \{c_1, c_2, \ldots, c_Q\}$ for $\mathcal{Z}$ values belonging to class C_q are set to: $c_1 = 0$, $c_Q = 1$ and

$$c_q = \frac{q}{Q} - \frac{1}{2Q},\ q = 2, 3, ...Q - 1. \tag{5}$$

We now construct z_i values for training inputs $\mathbf{x}_i^{(q)}$ by considering the following criteria. If $\mathbf{x}_i^{(q)}$ has similar minimum distances $\kappa(\mathbf{x}_i^{(q)}, q-1)$ and $\kappa(\mathbf{x}_i^{(q)}, q+1)$ (and consequently a high value of $w_{\mathbf{x}_i^{(q)}}$), the resulting z_i value should be closer to c_i, so that intuitively, we consider this pattern as *well located* within its class. If $\kappa(\mathbf{x}_i^{(q)}, q-1)$ and $\kappa(\mathbf{x}_i^{(q)}, q+1)$ are very different (and consequently a low value of $w_{\mathbf{x}_i^{(q)}}$ is obtained), the pattern $\mathbf{x}_i^{(q)}$ is closer to one of these classes and so the corresponding z_i value should be closer to the interval of $\mathcal{Z}$ values of the closest adjacent class, $q-1$ or $q+1$. This idea is formailized in the following expression:

$$z_i \quad = \quad \phi\left(\mathbf{x}_i^{(q)}\right) \quad = \quad \begin{cases} c_1 + (1 - w_{\mathbf{x}_i^{(1)}}) \cdot \frac{1}{Q}, \text{ if } q = 1, \\ c_q - (1 - w_{\mathbf{x}_i^{(q)}}) \cdot \frac{1}{2Q}, \text{ if } q \in \{2, \ldots, Q-1\} \text{ and} \\ \qquad\qquad \kappa(\mathbf{x}_i^{(q)}, q-1) \leq \kappa(\mathbf{x}_i^{(q)}, q+1), \\ c_q + (1 - w_{\mathbf{x}_i^{(q)}}) \cdot \frac{1}{2Q}, \text{ if } q \in \{2, \ldots, Q-1\} \text{ and} \\ \qquad\qquad \kappa(\mathbf{x}_i^{(q)}, q-1) > \kappa(\mathbf{x}_i^{(q)}, q+1), \\ c_Q - (1 - w_{\mathbf{x}_i^{(Q)}}) \cdot \frac{1}{Q}, \text{ if } q = Q, \end{cases} \tag{6}$$

[2]This does not in any way hamper generality, as our regressors defining g will be smooth non-linear functions.

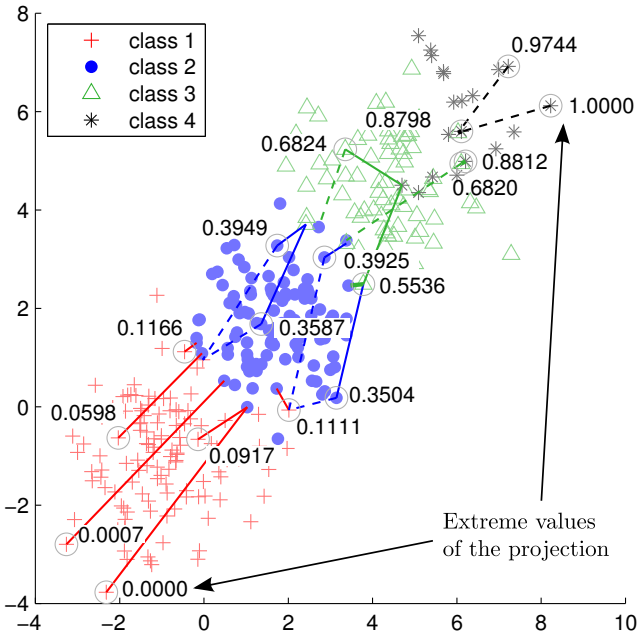


Figure 2: Example of the generated z_i values on a synthetic dataset with a linear order relationship.

where $w_{\mathbf{x}_i^{(q)}}$ is defined in Eq. (3), c_q is the centre of class interval corresponding to C_q (see Eq. (5)) and Q is the number of classes. Eq. (6) guarantees that all z values lie in the correct class interval[3]. This methodology for data projection is called *Pairwise Class Distances* (PCD).

3.2 Analysis of the proposed projection in synthetic datasets

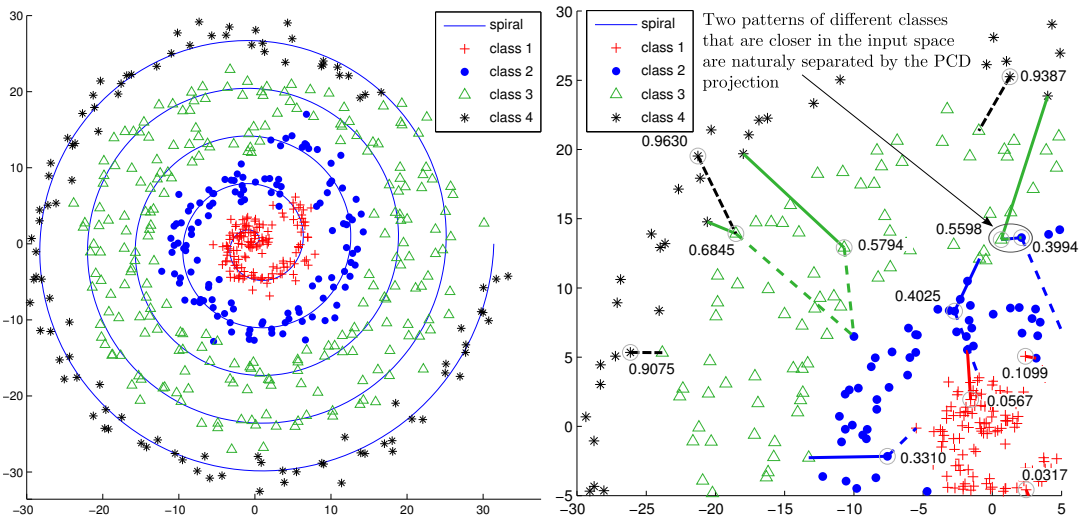
For illustration purposes, we generated synthetic ordinal classification datasets in $\mathbb{X} \in \mathbb{R}^2$ with four classes ($Q = 4$). Figure 2 shows the patterns of a synthetic dataset, *SyntheticLinearOrder*, with a linear order between classes, whereas Figure 3 shows *SyntheticNonLinearOrder* dataset, with a non-linear ordinal relationship between classes.

[3]Recall that the threshold set θ delimiting class intervals is defined in Eq. (4).

Points at *SyntheticLinearOrder* were generated by adding an uniform noise to points of a line. Points in *SyntheticNonLinearOrder* were generated by adding a Gaussian noise to points on a spiral. In both figures, points belonging to different classes are marked with different colours and symbols. Besides the points, the figures also illustrate basic concepts of the proposed method on example points (surrounded by grey circles). For these points, the minimum distances are illustrated with lines of the corresponding class colour. The minimum distance of a point to the previous and next class patterns are marked with dashed and solid lines, respectively. For selected points we show the value of the PCD projection (calculated using Eq. (6)).

In Figure 2 it can be seen that the z value increases for patterns of the higher classes, and this value varies depending of the position of the pattern $\mathbf{x}^{(q)}$ in the space with respect to the patterns $\mathbf{x}^{(q-1)}$ and $\mathbf{x}^{(q+1)}$ of adjacent classes. Extreme values, $z = 0.0$ and $z = 1.0$ correspond to the patterns more distant from the classes 1 and Q respectively (and with a maximum $w_{\mathbf{x}^{(q)}}$ value). *SyntheticNonLinearOrder* in Figure 3 is designed to demonstrate that the PCD projection is suitable for more complex ordinal topologies of the data. This is, for any topology in a ordinal dataset, it is expected that patterns of classes $q - 1$ and $q + 1$ are always the closest ones to the patterns of class q, and PCD will take advantage from this situation to decide the relative order of the pattern within its class, even when this is produced in a non-linear manner.

Figure 4a and Figure 4b show histograms of the PCD projections from the synthetic datasets in Figure 2 and Figure 3, respectively. The thresholds θ that divide the z values of the different classes are also included. Observe that the z values of the different classes are clearly separated, and that they are compacted within a range which is always



(a) General disposition of the points of the dataset.

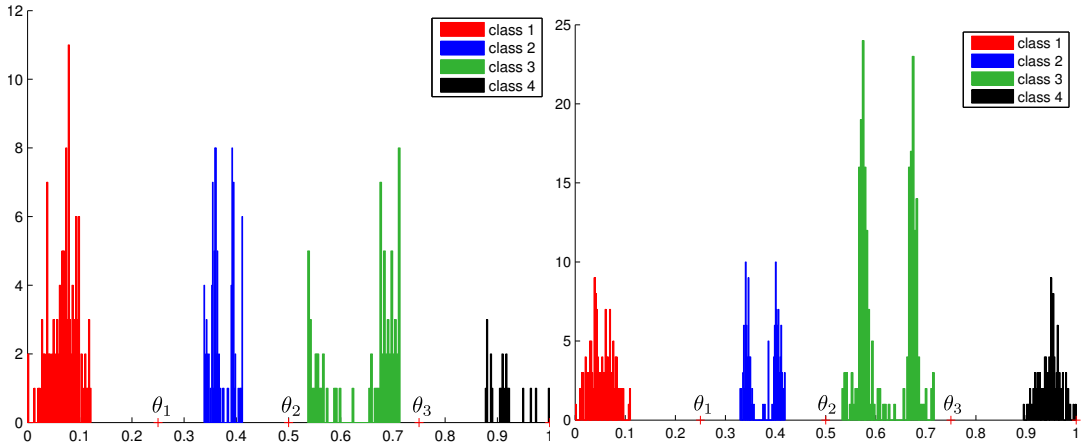
(b) Upper left area of the dataset.

Figure 3: Example of the generated z_i values on the synthetic dataset with a non-linear class order structure. Figure on the right shows a zooming over the upper left area at the center of the dataset shown on the left.

smaller than the range initially indicated by the thresholds. This is due to the scaling of the z values in Eq. (3), where the $w_{\mathbf{x}^{(q)}}$ value cannot be zero, so a pattern can never be located 'close' to the boundary separating intervals of adjacent classes.

3.3 Algorithm for ordinal classification

Once the PCD projections have been obtained for all training inputs, we construct a new training set $\mathbf{T}' = \left\{(\mathbf{x}_i, \phi(\mathbf{x}_i^{(y_i)}) \mid (\mathbf{x}_i, \mathrm{y}_i) \in \mathbf{T}\right\}$. Any generic regression tool can be trained on $\mathbf{T}'$ to obtain the projection function $g : \mathbb{X} \rightarrow \mathcal{Z}$. In this respect, our method is quite general, allowing the user to choose his or her favorite regression method or any other improved regression tool introduced in the future. The resulting algorithm, named Pairwise Class Distances for Ordinal Classification (PCDOC), is de-



(a) *SyntheticLinearOrder* dataset. (b) *SyntheticNonLinearOrder* dataset.

Figure 4: Histograms of the PCD projection of the synthetic datasets.

scribed in two steps in Figure 5 and Figure 6.

It is expected that formulating the problem as a regression problem would help the model to capture the ordinal structure of the input and output spaces, and their relationship. In addition, due to the nature of the regression problem, it is expected that the performance of the classification task will be improved regarding metrics that consider the difference between the predicted and actual classes within the linear class order, such as MAE or $AMAE$, or the correlation between the target and predicted values, such as τ_b. Experimental results confirm this hypothesis in Section 4.3.

3.4 PCDOC performance analysis in some controlled experiments

Analysis of the influence of dimensionality and class overlapping. This section analyses the performance of the PCDOC algorithm under situations that may hamper classification: class overlapping and large dimensionality of the data. For this purpose, different synthetic datasets have been generated by sampling random points from Q

PCDOC Training: $\{g, \boldsymbol{\theta}\} = \mathrm{PCDOCtr}(\mathbf{T})$.

Require: Training dataset $\mathbf{T}$.

Ensure: Regressor (g) and thresholds ($\boldsymbol{\theta}$).

1: Calculate thresholds $\boldsymbol{\theta}$ and centres $\mathbf{c}$ according to Eq. (4) and Eq. (5).

2: For each pattern, calculate z_i according to Eq. (6): $z_i = \phi(\mathbf{x}_i^{(q)})$.

3: Build a regressor g, considering z as the regression response variable: $z = g(\mathbf{x})$.

4: **return** $\{g, \boldsymbol{\theta}\}$

Figure 5: PCDOC regression training algorithm pseudocode.

PCDOC Prediction: $\hat{y} = \mathrm{PCDOCpr}(\mathbf{x}, g, \boldsymbol{\theta})$.

Require: Regressor (g), thresholds ($\boldsymbol{\theta}$) and test input ($\mathbf{x}$).

Ensure: Predicted label ($\hat{y}$).

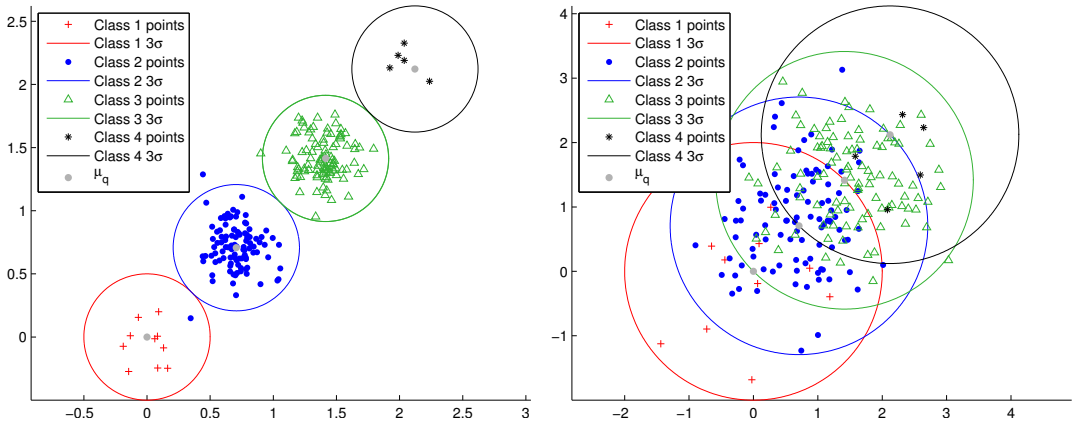
1: Predict the latent variable value using the regressor g: $\hat{z} = g(\mathbf{x})$.

2: Map the $\hat{z}$ value to the corresponding class using f as defined in Eq. (1): $\hat{y} = f(\hat{z}, \boldsymbol{\theta})$.

3: **return** $\hat{y}$

Figure 6: PCDOC classification algorithm for unseen data.

Gaussian distributions, where Q is the number of classes, so that each class points are random samples of the corresponding Gaussian distribution. In order to easily control the overlap of the classes, the variance (σ^2) is kept constant independently of the number of dimensions (K). In addition, the Q centres (means μ_q) are set up in order to keep the distance of 1 between two adjacent class means independently of K. Under this situation, each coordinate of adjacent class means is separated by $\triangle_\mu = 1/\sqrt{K}$ so that



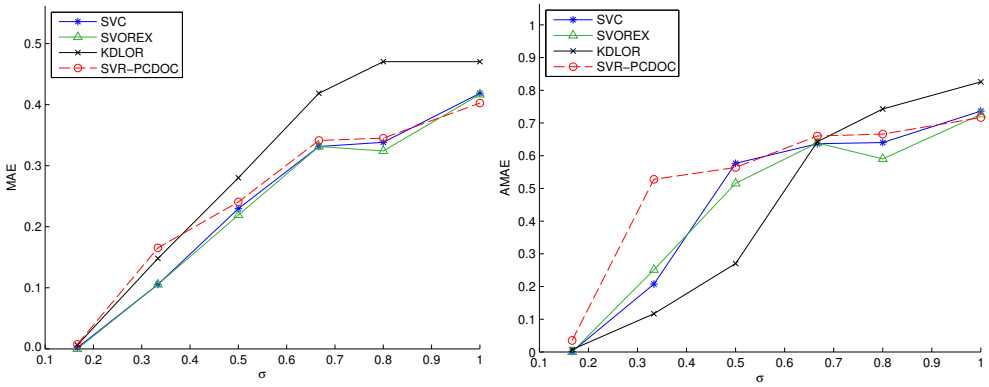
(a) Synthetic Gaussian dataset with $\sigma = 0.167$. (b) Synthetic Gaussian dataset with $\sigma = 0.667$.

Figure 7: Synthetic Gaussian dataset example for two dimensions.

$\mu_1 = 0$, $\mu_2 = \mu_1 + \triangle_\mu$, $\mu_3 = \mu_2 + \triangle_\mu$ and so on.

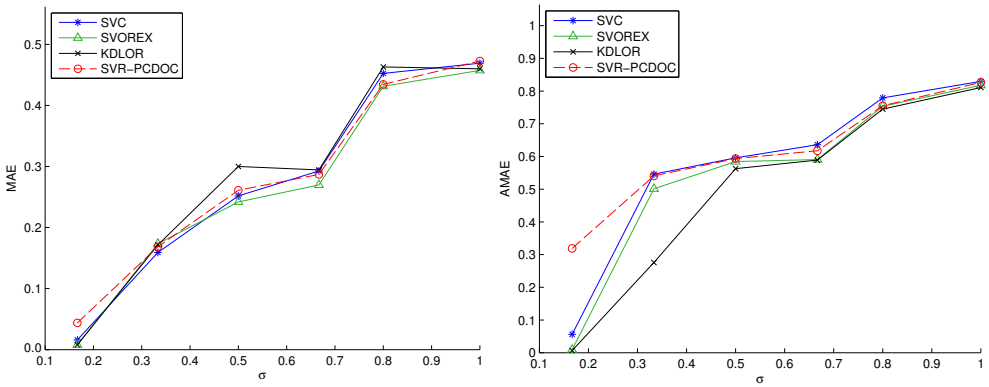
The number of features tested (input space dimensionality) were $K \in \{10, 50, 100\}$ and the different width values are $\sigma \in \{0.167, 0.333, 0.500, 0.667, 0.800, 1.000\}$, so that 18 datasets were generated. The number of patterns for each class from one to four was 10, 100, 100 and 5. Figure 7 shows two of these datasets generated with different variance values for $K = 2$.

For these experiments, our approach uses the Support Vector Regression (SVR) algorithm as the model for the z variable (the method will be referred to as SVR-PCDOC). We have also included three methods as baseline methods: the *C*-Support Vector Classification (SVC) (Cortes and Vapnik, 1995; Vapnik, 1999), the Support Vector Ordinal Regression with explicit constraints (SVOREX) (Chu and Keerthi, 2005, 2007) and the Kernel Discriminant Learning for Ordinal Regression (KDLOR) (Sun et al., 2010). As in the next experimental section (Section 4), the experimental design includes 30 stratified random splits (with 75% of patterns for training and the remain-



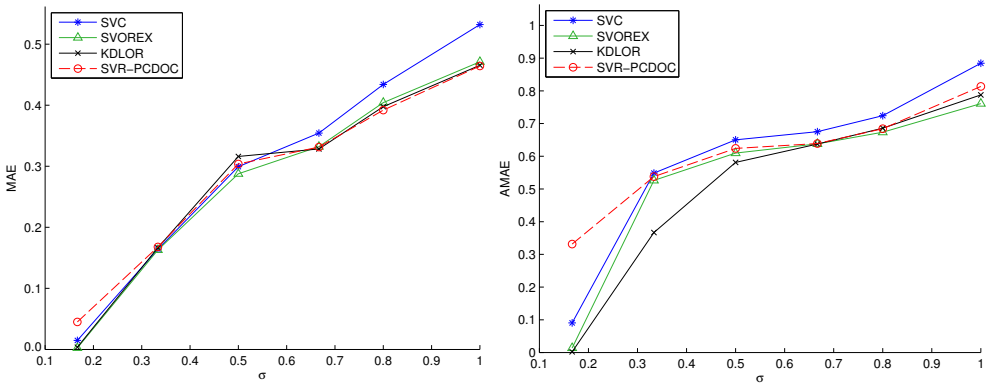
(a) MAE with $K = 10$.

(b) $AMAE$ with $K = 10$.



(c) MAE with $K = 50$.

(d) $AMAE$ with $K = 50$.



(e) MAE with $K = 100$.

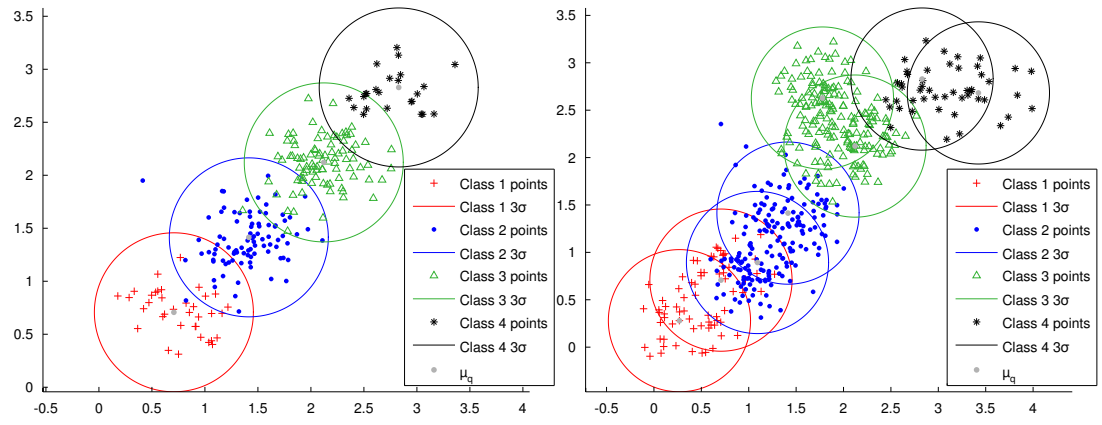
(f) $AMAE$ with $K = 100$.

Figure 8: MAE and $AMAE$ performance for synthetic Gaussian dataset with distribution $\mathbf{x} = \mathcal{N}(\mu,\, \sigma^2 \mathbf{I}_{K\times K})$ and $\mathbf{x} \in \mathbb{X} \subseteq \mathbb{R}^K$, where $\mathbf{I}_{K\times K}$ is the identity matrix.

ing for generalization). The mean MAE and $AMAE$ generalization results are used for comparison purposes at Figure 8. For further details about experimental procedure, methods description and hyper-parameters optimization please refer to the next experimental section (Subsection 4.2).

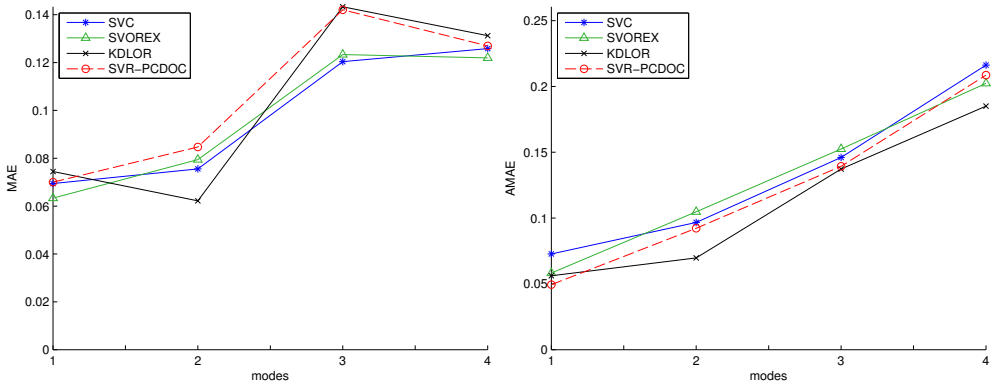
From the results depicted at Figure 8, we can generally conclude that the three methods, except KDLOR, have similar MAE performance degradation with the increase of class overlapping and dimensionality. Figure 8a shows that SVR-PCDOC have a slightly worse performance than SVC and SVOREX. However, in experiments with higher K (Figures 8c and 8e) the performance of the three ordinal methods varies in a similar way. In partigular, in Figure 8e we can observe that SVC performance decreases with high overlapping and high dimensionality, whereas the ordinal methods have similar performance here. From the analysis of the $AMAE$ performance we can conclude that KDLOR outperforms the rest of the methods in cases of low class overlapping. Regarding our method, we can conclude that compared with the other methods its $AMAE$ performance is worse in the case of low class overlap. However, in general, our method seems more robust when the class overlap increases.

Analysis of the influence of data multimodality. This section extends the above experiments to the case of multimodal data, the datasets are generated with $K = 2$ and $\sigma^2 = 0.25$, and the number of modes per class is varied. Figure 9a presents the unimodal case. The datasets with more modes per class are generated in the following way. A Gaussian distribution is set up as in the previous section, with center μ_q. For each class, each additional Gaussian distribution is centered in a random location within



(a) (b)

Figure 9: Illustration of the unimodal and bimodal cases of the synthetic Gaussian dataset example for $K = 2$ and $\sigma^2 = 0.25$.



(a) MAE performance. (b) $AMAE$ performance.

Figure 10: MAE and $AMAE$ performance for synthetic Gaussian dataset with distribution $\mathbf{x} = \mathcal{N}(\mu,\, \sigma^2\mathbf{I}_{K\times K})$ and $\mathbf{x} \in \mathbb{X} \subseteq \mathbb{R}^K$, where $\mathbf{I}_{K\times K}$ is the identity matrix (being $K = 2$ and $\sigma^2 = 0.25$ for all the synthetic datasets).

the hyper-sphere with center μ_q and radius 0.75. Then, patterns are sampled from each distribution. For each class, we considered different number of modes, from one mode to four modes. The number of patterns generated for each mode was 36, 90, 90 and 24

for class 1, 2, 3 and 4, respectively, using the same number for all modes of a class. An example of the bimodal case (two Gaussian distributions per class) is shown in Figure 9b, having 72, 180, 180 and 48 patterns for class 1, 2, 3 and 4, respectively.

Experiments were carried out as in the previous section, and MAE and $AMAE$ generalization results are depicted in Figure 10. Regarding MAE, Figure 10a reveals that the fours methods perform similarly in datasets with one and four modes but they differ on performance for the two and three modes. Only considering MAE, SVRPCDOC has the worse performance in case two and three. Nevertheless, considering $AMAE$ results at Figure 10b, SVRPCDOC and KDLOR achieve the best results. The different behaviour of the methods depending on the performance measure can be explained by observing the nature of the bimodal dataset (see Figure 9b), where the majority of the patterns are from classes two and three. In this context, the optimization done by SVOREX and SVC can move the decision thressholds to better classify patterns of these two classes at the expense of misclassifying class one and four patterns, especially patterns of those classes placed on the class boundaries (see Figure 9b).

4 Experiments

In this section we report on extensive experiments that were performed to check the competitiveness of the proposed methodology. Source code of the proposed method, synthetic datasets analysis code and real ordinal data sets partitions used for the experiments are available at a public website[4].

[4] `http://www.uco.es/ayrna/`

4.1 Ordinal classification datasets and experimental design

To the best of our knowledge, there are no public specific datasets repositories for real ordinal classification problems. The ordinal regression benchmark datasets repository provided by Chu et. al (Chu and Ghahramani, 2005) is the most widely used repository in the literature. However, these datasets are not real ordinal classification datasets but regression ones. To turn regression into ordinal classification, the target variable was discretized into Q different bins (representing classes), with equal frequency or equal width. However, there are potential problems with this approach. If equal frequency labelling is considered, the datasets do not exhibit some characteristics of typical complex classification tasks, such as class imbalance. On the other hand, severe class imbalance can be introduced by using the same binning width. Finally, as the actual target regression variable exists with observed values, the classification problem can be simpler than on those datasets where the variable z is really unobservable and has to be modelled.

We have therefore decided to use a set of real ordinal classification datasets publicly available at the UCI (Asuncion and Newman, 2007) and *mldata.org* repositories (Sonnenburg, 2011) (see Table 1 for data description). All of them are ordinal classification problems, although one can find literature where the ordering information is discarded. The nature of the target variable is now analysed for two example datasets. *bondrate* dataset is a classification problem where the purpose is to assign the right ordered category to bonds, being the category labels $\{C_1 = \mathrm{AAA}, C_2 = \mathrm{AA}, C_3 = \mathrm{A}, C_4 = \mathrm{BBB}, C_5 = \mathrm{BB}\}$. These labels represent the quality of a bond and are assigned by credit rating agencies, AAA being the highest quality and BB the worst one. In this case, classes AAA, AA, A are more similar than classes BBB and BB so that no

assumptions should be done about the distance between classes both in the input and latent space. Other example is *eucalyptus* dataset, in this case the problem is to predict which eucalyptus seedlots are best for soil conservation in a seasonally dry hill country. Being the classes $\{C_1 = \text{none}, C_2 = \text{low}, C_3 = \text{average}, C_4 = \text{good}, C_5 = \text{best}\}$, it cannot be assumed an equal width for each class in the latent space.

Regarding the experimental set up, 30 different random splits of the datasets have been considered, with 75% and 25% of the instances in the training and test sets respectively. The partitions were the same for all compared methods, and, since all of them are deterministic, one model was obtained and evaluated (in the test (generalization) set), for each split. All nominal attributes were transformed into as many binary attributes as the number of categories. All the datasets were property standardized.

4.2 Existing methods used for comparison purposes

For comparison purposes, different state-of-the-art methods have been included in the experimentation (all of them mentioned in the Introduction):

- **Gaussian Processes for Ordinal Regression (GPOR)** (Chu and Ghahramani, 2005), presents a probabilistic kernel approach to ordinal regression based on Gaussian processes where a threshold model that generalizes the *probit* function is used as the likelihood function for ordinal variables. In addition, Chu applies the automatic relevance determination (ARD) method proposed by (Mackay, 1994) and (Neal, 1996) to the GPOR model. When using GPOR with ARD feature selection, we will refer the algorithm to as GPOR-ARD.

- **Support Vector Ordinal Regression (SVOR)** (Chu and Keerthi, 2005)(Chu and

Table 1: Datasets used for the experiments (N is the number of patterns, K is the number of attributes and Q is the number of classes).

Dataset	N	K	Q	Ordered Class Distribution
automobile	205	71	6	(3,22,67,54,32,27)
bondrate	57	37	5	(6,33,12,5,1)
contact-lenses	24	6	3	(15,5,4)
eucalyptus	736	91	5	(180,107,130,214,105)
newthyroid	215	5	3	(30,150,35)
pasture	36	25	3	(12,12,12)
squash-stored	52	51	3	(23,21,8)
squash-unstored	52	52	3	(24,24,4)
tae	151	54	3	(49,50,52)
winequality-red	1599	11	6	(10,53,681,638,199,18)

Keerthi, 2007), proposes two new support vector approaches for ordinal regression. Here, multiple thresholds are optimized in order to define parallel discriminant hyperplanes for the ordinal scales. The first approach, with explicit inequality constraints on the thresholds, derive the optimal conditions for the dual problem, and adapt the SMO algorithm for the solution, and we will refer to it as **SVOREX**. In the second approach, the samples in all the categories are allowed to contribute errors for each threshold, therefore there is no need of including the inequality constraints in the problem. This approach is named a SVOR with implicit constraints (**SVORIM**).

- **RED-SVM** (Li and Lin, 2007) applies the reduction from cost-sensitive ordinal ranking to weighted binary classification (RED) framework to SVM. The RED method can be summarized in the following three steps. First, transform all training samples into extended samples by using a coding matrix, and weighting these samples with a cost matrix. Second, all the extended examples are jointly learned by a binary classifier with confidence outputs, aiming at a low weighted $0/1$ loss. Last step is used to convert the binary outputs to a rank. In this paper, the coding matrix considered is the identity and the cost matrix is the absolute value matrix, applied to the standard binary soft-margin SVM.

- **A Simple Approach to Ordinal Regression (ASAOR)** by Frank et. al (Frank and Hall, 2001) is a general method that enables standard classification algorithms to make the use of order information in attributes. For the training process, the method transforms the Q-class ordinal problem into $Q-1$ binary class problems. Any ordinal attribute with ordered values is converted into $Q-1$ binary attributes. The prediction of new instances class is done by estimating the probability of belonging to each of the Q classes with the $Q-1$ models. In the current work, the C4.5 method available in Weka (Hall et al., 2009) is used as the underlying classification algorithm since this is the one initially employed by the authors of ASAOR. In this way, the algorithm is identified as ASAOR(C4.5).

- The **Proportional Odds Model (POM)** is one of the first models specifically designed for ordinal regression (McCullagh, 1980). The model is based on the assumption of stochastic ordering of the space $\mathbb{X}$. Stochastic ordering is satisfied

by a monotonic function (the model) that defines a probability density function over the class labels for a given feature vector $\mathbf{x}$. Due to the thresholds that divide the monotonic function values corresponding to different classes, this method was the first one to be named a *threshold model*. The main problem associated with this model is that the projection is done by considering a linear combination of the inputs (linear projection), which hinders its performance. For the POM model, the `mnrfit` function of Matlab software has been used.

- **Kernel Discriminant Learning for Ordinal Regression (KDLOR)** (Sun et al., 2010) extends the Kernel Discriminant Analysis (KDA) using a rank constraint. The method looks for the optimal projection that maximizes the separation between the projection of the different classes and minimizes the intra-class distance as in traditional discriminant analysis for nominal classes. Crucially, however, the order of the classes in the resulting projection is also considered. The authors claim that, compared with the SVM based methods, the KDA approach takes advantage of the global information of the data and the distribution of the classes, and also reduces the computational complexity of the problem.

- **Support Vector Machine (SVM)** (Cortes and Vapnik, 1995; Vapnik, 1999) nominal classifier is included in the experiments in order to establish a baseline nominal performance. *C*-Support Vector Classification (SVC) available in libSVM 3.0 (Chang and Lin, 2011) is used as the SVM classifier implementation. In order to deal with the multiclass case, a "1-versus-1" approach has been considered, following the recommendations of Hsu and Lin (Hsu and Lin, 2002).

In our approach, the Support Vector Regression (SVR) algorithm is used as the model for the z variable. The method will be referred to by the acronym SVR-PCDOC. The ϵ-SVR available in libSVM is used. The authors of GPOR, SVOREX, SVORIM and RED-SVM provide publicly available software implementations of their methods[5]. In the case of KDLOR, this method has been implemented by the authors using Matlab software (Perez-Ortiz et al., 2011).

Model selection is an important issue and involves selecting the best hyper-parameter combination for all the methods compared. All the methods were configured to use the Gaussian kernel. For the support vector algorithms, i.e. SVC, RED-SVM, SVOREX, SVORIM and ϵ-SVR, the corresponding hyper-parameters (regularization parameter, C, and width of the Gaussian functions, γ), were adjusted using a grid search over each of the 30 training sets by a 5-fold nested cross-validation with the following ranges: $C \in \{10^3, 10^2, \ldots, 10^{-3}\}$ and $\gamma \in \{10^3, 10^2, \ldots, 10^{-3}\}$. Regarding ϵ-SVR, the additional ϵ parameter has to be adjusted. The range considered was $\epsilon \in \{10^3, 10^2, 10^1, 10^0\}$. For KDLOR, the width of the Gaussian kernel was adjusted by using the range $\gamma \in \{10^3, 10^2, \ldots, 10^{-3}\}$, and the regularization parameter, u, for avoiding the singularity problem values were $u \in \{10^{-2}, 10^{-3}, \ldots, 10^{-5}\}$. POM and ASAOR(C4.5) methods have not hyper-parameters. Finally, GPOR-ARD has no hyper-parameters to fix, since the method optimizes the associated parameters itself.

[5] GPOR (`http://www.gatsby.ucl.ac.uk/~chuwei/ordinalregression.html`), SVOREX and SVORIM (`http://www.gatsby.ucl.ac.uk/~chuwei/svor.htm`) and RED-SVM (`http://home.caltech.edu/~htlin/program/libsvm/`)

For all the methods, the MAE measure is used as the performance metric for guiding the grid search to be consistent with the authors of the different state-of-the-art methods. The grid search procedure of SVC at libSVM has been modified in order to use MAE as the criteria for hyper-parameters selection.

4.3 Performance results

Tables 2 and 3 outline the results through the mean and standard deviation (SD) of Acc_{G}, MAE_{G}, $AMAE_{\mathrm{G}}$ and $\tau_{b\mathrm{G}}$ across the 30 hold-out splits, where the subindex G indicates that results were obtained on the (hold-out) generalization fold. As a summary, Table 4 shows, for each performance metric, the mean values of the metrics across all the datasets, and the mean ranking values when comparing the different methods ($R = 1$ for the best performing method and $R = 9$ for the worst one). To enhance readability, in Tables 2, 3 and 4 the best and second-best results are in bold face and italics, respectively.

Table 2: Comparison of the proposed method to other ordinal classification methods and SVC. The mean and standard deviation (SD) of the generalization results are reported for each dataset. The best statistical result is in bold face and the second best result in italics.

	Acc Mean_{SD}									
Method/DataSet	automobile	bondrate	contact-lenses	eucalyptus	newthyroid	pasture	squash-stored	squash-unstored	tae	winequality-red
ASAOR(C4.5)	$0.696_{0.059}$	$0.533_{0.074}$	$\mathit{0.750_{0.085}}$	$0.639_{0.036}$	$0.917_{0.039}$	$\mathbf{0.752_{0.145}}$	$0.603_{0.118}$	$\mathit{0.774_{0.101}}$	$0.395_{0.058}$	$0.603_{0.021}$
GPOR	$0.611_{0.073}$	$\mathbf{0.578_{0.032}}$	$0.606_{0.093}$	$\mathbf{0.686_{0.034}}$	$0.966_{0.024}$	$0.522_{0.178}$	$0.451_{0.101}$	$0.644_{0.162}$	$0.328_{0.041}$	$0.606_{0.015}$
KLDOR	$\mathbf{0.722_{0.058}}$	$0.542_{0.087}$	$0.589_{0.174}$	$0.611_{0.028}$	$\mathit{0.972_{0.019}}$	$\mathit{0.678_{0.125}}$	$\mathbf{0.703_{0.112}}$	$\mathbf{0.828_{0.104}}$	$0.555_{0.052}$	$0.603_{0.017}$
POM	$0.467_{0.194}$	$0.344_{0.161}$	$0.622_{0.138}$	$0.159_{0.036}$	$\mathit{0.972_{0.022}}$	$0.496_{0.154}$	$0.382_{0.152}$	$0.349_{0.143}$	$0.512_{0.089}$	$0.594_{0.017}$
SVC	$\mathit{0.697_{0.062}}$	$\mathit{0.556_{0.069}}$	$\mathbf{0.794_{0.129}}$	$\mathit{0.653_{0.037}}$	$0.967_{0.025}$	$0.633_{0.134}$	$0.656_{0.127}$	$0.700_{0.082}$	$0.539_{0.062}$	$\mathbf{0.636_{0.021}}$
RED-SVM	$0.684_{0.055}$	$0.553_{0.073}$	$0.700_{0.111}$	$0.651_{0.024}$	$0.969_{0.022}$	$0.648_{0.134}$	$0.664_{0.104}$	$0.749_{0.086}$	$0.522_{0.074}$	$0.618_{0.022}$
SVOREX	$0.665_{0.068}$	$0.553_{0.096}$	$0.650_{0.127}$	$0.647_{0.029}$	$0.967_{0.022}$	$0.630_{0.125}$	$0.628_{0.133}$	$0.718_{0.128}$	$0.581_{0.060}$	$0.629_{0.022}$
SVORIM	$0.639_{0.076}$	$0.547_{0.092}$	$0.633_{0.127}$	$0.639_{0.028}$	$0.969_{0.021}$	$0.667_{0.120}$	$0.639_{0.118}$	$0.764_{0.103}$	$\mathbf{0.590_{0.066}}$	$0.630_{0.022}$
SVR-PCDOC	$0.678_{0.060}$	$0.540_{0.101}$	$0.689_{0.095}$	$0.648_{0.029}$	$\mathbf{0.973_{0.020}}$	$0.656_{0.103}$	$\mathit{0.685_{0.123}}$	$0.695_{0.084}$	$\mathit{0.582_{0.064}}$	$\mathit{0.631_{0.022}}$
	MAE Mean_{SD}									
Method/DataSet	automobile	bondrate	contact-lenses	eucalyptus	newthyroid	pasture	squash-stored	squash-unstored	tae	winequality-red
ASAOR(C4.5)	$0.401_{0.095}$	$0.624_{0.079}$	$\mathit{0.367_{0.154}}$	$0.384_{0.042}$	$0.083_{0.039}$	$\mathbf{0.248_{0.145}}$	$0.444_{0.140}$	$\mathit{0.239_{0.109}}$	$0.686_{0.146}$	$0.441_{0.023}$
GPOR	$0.594_{0.131}$	$0.624_{0.062}$	$0.511_{0.175}$	$\mathbf{0.331_{0.038}}$	$0.034_{0.024}$	$0.489_{0.190}$	$0.626_{0.148}$	$0.356_{0.162}$	$0.861_{0.155}$	$0.425_{0.017}$
KLDOR	$\mathbf{0.334_{0.076}}$	$0.587_{0.107}$	$0.539_{0.208}$	$0.424_{0.032}$	$\mathit{0.028_{0.019}}$	$\mathit{0.322_{0.125}}$	$\mathbf{0.308_{0.128}}$	$\mathbf{0.172_{0.104}}$	$0.473_{0.069}$	$0.443_{0.019}$
POM	$0.953_{0.687}$	$0.947_{0.321}$	$0.533_{0.241}$	$2.029_{0.070}$	$\mathit{0.028_{0.022}}$	$0.585_{0.204}$	$0.813_{0.248}$	$0.826_{0.230}$	$0.626_{0.126}$	$0.439_{0.019}$
SVC	$0.446_{0.095}$	$0.624_{0.090}$	$\mathbf{0.311_{0.222}}$	$0.394_{0.042}$	$0.033_{0.025}$	$0.367_{0.134}$	$0.377_{0.160}$	$0.308_{0.090}$	$0.578_{0.083}$	$0.408_{0.020}$
RED-SVM	$\mathit{0.393_{0.073}}$	$0.598_{0.088}$	$0.378_{0.169}$	$\mathit{0.380_{0.027}}$	$0.032_{0.022}$	$0.359_{0.142}$	$0.346_{0.110}$	$0.251_{0.086}$	$0.515_{0.087}$	$0.419_{0.021}$
SVOREX	$0.408_{0.089}$	$\mathit{0.573_{0.121}}$	$0.489_{0.185}$	$0.392_{0.031}$	$0.033_{0.022}$	$0.370_{0.125}$	$0.382_{0.139}$	$0.282_{0.128}$	$0.485_{0.078}$	$0.408_{0.023}$
SVORIM	$0.424_{0.090}$	$0.591_{0.102}$	$0.506_{0.167}$	$0.395_{0.035}$	$0.031_{0.021}$	$0.333_{0.120}$	$0.372_{0.126}$	$\mathit{0.239_{0.109}}$	$\mathit{0.461_{0.081}}$	$\mathit{0.406_{0.022}}$
SVR-PCDOC	$0.397_{0.093}$	$\mathbf{0.568_{0.126}}$	$\mathit{0.367_{0.154}}$	$0.392_{0.038}$	$\mathbf{0.027_{0.020}}$	$0.348_{0.104}$	$\mathit{0.326_{0.141}}$	$0.305_{0.084}$	$\mathbf{0.457_{0.071}}$	$\mathbf{0.400_{0.023}}$

Table 3: Comparison of the proposed method to other ordinal classification methods and SVC. The mean and standard deviation (SD) of the generalization results are reported for each dataset. The best statistical result is in bold face and the second best result in italics.

	$AMAE$ Mean_{SD}									
Method/DataSet	automobile	bondrate	contact-lenses	eucalyptus	newthyroid	pasture	squash-stored	squash-unstored	tae	winequality-red
ASAOR(C4.5)	$0.511_{0.104}$	$1.226_{0.175}$	$\mathit{0.315}_{\mathit{0.124}}$	$0.428_{0.045}$	$0.115_{0.056}$	$\mathbf{0.248_{0.145}}$	$0.502_{0.192}$	$\mathbf{0.256_{0.149}}$	$0.689_{0.151}$	$\mathit{1.045}_{\mathit{0.080}}$
GPOR	$0.792_{0.200}$	$1.360_{0.122}$	$0.651_{0.286}$	$\mathbf{0.362_{0.040}}$	$0.062_{0.049}$	$0.489_{0.190}$	$0.797_{0.234}$	$0.443_{0.226}$	$0.863_{0.164}$	$1.065_{0.065}$
KLDOR	$\mathbf{0.345_{0.104}}$	$\mathit{1.037}_{\mathit{0.270}}$	$0.519_{0.280}$	$0.426_{0.038}$	$0.059_{0.040}$	$\mathit{0.322}_{\mathit{0.125}}$	$\mathbf{0.349_{0.156}}$	$\mathit{0.309}_{\mathit{0.180}}$	$0.471_{0.070}$	$1.258_{0.069}$
POM	$1.026_{0.800}$	$1.103_{0.403}$	$0.535_{0.275}$	$1.990_{0.048}$	$\mathit{0.050}_{\mathit{0.040}}$	$0.585_{0.204}$	$0.815_{0.251}$	$0.791_{0.332}$	$0.627_{0.128}$	$1.085_{0.037}$
SVC	$0.486_{0.125}$	$1.265_{0.183}$	$\mathbf{0.307_{0.277}}$	$0.433_{0.048}$	$0.060_{0.051}$	$0.367_{0.134}$	$0.446_{0.189}$	$0.444_{0.163}$	$0.576_{0.083}$	$1.119_{0.069}$
RED-SVM	$0.468_{0.096}$	$1.184_{0.225}$	$0.385_{0.198}$	$0.414_{0.030}$	$0.057_{0.049}$	$0.359_{0.142}$	$0.391_{0.149}$	$0.348_{0.159}$	$0.513_{0.086}$	$1.068_{0.069}$
SVOREX	$0.518_{0.096}$	$1.072_{0.217}$	$0.517_{0.303}$	$0.411_{0.034}$	$0.054_{0.042}$	$0.370_{0.125}$	$0.433_{0.172}$	$0.426_{0.157}$	$0.484_{0.079}$	$1.095_{0.067}$
SVORIM	$0.523_{0.105}$	$1.114_{0.233}$	$0.589_{0.259}$	$0.420_{0.043}$	$0.055_{0.042}$	$0.333_{0.120}$	$0.427_{0.148}$	$0.367_{0.140}$	$\mathit{0.459}_{\mathit{0.081}}$	$1.093_{0.072}$
SVR-PCDOC	$\mathit{0.440}_{\mathit{0.128}}$	$\mathbf{0.969_{0.224}}$	$0.420_{0.098}$	$\mathit{0.400}_{\mathit{0.043}}$	$\mathbf{0.045_{0.040}}$	$0.348_{0.104}$	$\mathit{0.360}_{\mathit{0.184}}$	$0.396_{0.158}$	$\mathbf{0.455_{0.071}}$	$\mathbf{1.040_{0.096}}$
	τ_b Mean_{SD}									
Method/DataSet	automobile	bondrate	contact-lenses	eucalyptus	newthyroid	pasture	squash-stored	squash-unstored	tae	winequality-red
ASAOR(C4.5)	$0.741_{0.069}$	$0.143_{0.159}$	$\mathit{0.604}_{\mathit{0.216}}$	$\mathit{0.802}_{\mathit{0.025}}$	$0.853_{0.067}$	$\mathbf{0.778_{0.132}}$	$0.415_{0.245}$	$\mathit{0.692}_{\mathit{0.145}}$	$0.243_{0.177}$	$0.496_{0.036}$
GPOR	$0.557_{0.118}$	$0.000_{0.000}$	$0.348_{0.304}$	$\mathbf{0.830_{0.020}}$	$0.938_{0.045}$	$0.461_{0.314}$	$0.075_{0.211}$	$0.420_{0.331}$	$-0.018_{0.108}$	$0.523_{0.026}$
KLDOR	$\mathbf{0.793_{0.056}}$	$0.356_{0.257}$	$0.448_{0.273}$	$0.786_{0.017}$	$0.948_{0.034}$	$\mathit{0.718}_{\mathit{0.133}}$	$\mathbf{0.646_{0.160}}$	$\mathbf{0.764_{0.161}}$	$0.477_{0.114}$	$0.460_{0.028}$
POM	$0.495_{0.283}$	$0.290_{0.302}$	$0.458_{0.309}$	$0.008_{0.038}$	$\mathit{0.949}_{\mathit{0.040}}$	$0.463_{0.237}$	$0.169_{0.304}$	$0.109_{0.305}$	$0.317_{0.129}$	$0.497_{0.025}$
SVC	$0.695_{0.077}$	$0.121_{0.177}$	$0.601_{0.300}$	$0.783_{0.025}$	$0.939_{0.045}$	$0.698_{0.133}$	$0.541_{0.240}$	$0.599_{0.140}$	$0.375_{0.110}$	$0.516_{0.027}$
RED-SVM	$\mathit{0.751}_{\mathit{0.054}}$	$0.254_{0.247}$	$0.577_{0.242}$	$0.800_{0.017}$	$0.943_{0.041}$	$0.707_{0.129}$	$0.601_{0.148}$	$0.662_{0.108}$	$0.417_{0.120}$	$0.525_{0.030}$
SVOREX	$0.749_{0.062}$	$\mathit{0.369}_{\mathit{0.216}}$	$0.425_{0.304}$	$0.794_{0.019}$	$0.941_{0.040}$	$0.691_{0.115}$	$0.534_{0.207}$	$0.592_{0.212}$	$0.445_{0.110}$	$0.531_{0.028}$
SVORIM	$0.748_{0.065}$	$0.299_{0.230}$	$0.382_{0.269}$	$0.792_{0.020}$	$0.944_{0.038}$	$0.710_{0.114}$	$0.542_{0.167}$	$0.656_{0.187}$	$\mathit{0.482}_{\mathit{0.118}}$	$\mathit{0.533}_{\mathit{0.030}}$
SVR-PCDOC	$0.744_{0.076}$	$\mathbf{0.455_{0.218}}$	$\mathbf{0.620_{0.217}}$	$0.795_{0.024}$	$\mathbf{0.952_{0.037}}$	$0.712_{0.102}$	$\mathit{0.610}_{\mathit{0.201}}$	$0.602_{0.133}$	$\mathbf{0.493_{0.101}}$	$\mathbf{0.542_{0.033}}$

Regarding Tables 2 and 3, it can be seen that the majority of methods are very competitive. The best performing method depends on the considered performance metric, as it can be seen from the mean rankings. This is also true when separately considering each of the datasets, and the performance for some datasets varies noticeably if $AMAE_{\mathrm{G}}$ is considered instead of MAE_{G} (see *bondrate*, *contact-lenses*, *eucalyptus*, *squash-unstored* and *winequality-red*). In the case of *winequality-red*, it happens that the second worse method in MAE_{G}, ASAOR(C4.5), is the second best one for $AMAE_{\mathrm{G}}$. It is worthwhile to mention that for *pasture* dataset the mean MAE_{G} and $AMAE_{\mathrm{G}}$ are the same, which is due to the fact that *pasture* is a perfectly balanced dataset (see Section 2.2). In the case of *tae*, MAE_{G} and $AMAE_{\mathrm{G}}$ are very similar since the patterns distribution across classes is very similar. Regarding $\tau_{b\mathrm{G}}$, it is interesting to highlight that a value close to zero of this metric reveals that the classifier predictions are not related to the real values, this is, the classifier performance is similar to the performance of a trivial classifier. This happens for GPOR method in the *bondrate*, *squash-stored* and *tae* datasets, and for POM in the *eucalyptus* dataset.

From Table 4, it can be observed how best mean value across the different datasets is not always translated into best mean ranking (see $\overline{Acc}_{\mathrm{G}}$ and $\overline{R}_{Acc_{\mathrm{G}}}$ columns). We now analyze the results in greater detail, highlighting the best and second best performances. When considering Acc_{G}, SVC is clearly the best method, both in average performance and ranking. KDLOR and SVR-PCDOC are the second best methods, in average value and ranking, respectively. However, results are very different for all the other measures, where the order is included in the evaluation. The best method in average MAE_{G} and ranking of MAE_{G} is SVR-PCDOC and the second best ranks are for KDLOR and

Table 4: Mean results of accuracy ($\overline{Acc}_{\mathrm{G}}$), MAE ($\overline{MAE}_{\mathrm{G}}$), $AMAE$ ($\overline{AMAE}_{\mathrm{G}}$) and τ_b ($\overline{\tau}_{b\mathrm{G}}$) and mean ranking ($\overline{R}_{Acc_{\mathrm{G}}}$, $\overline{R}_{MAE_{\mathrm{G}}}$, $\overline{R}_{AMAE_{\mathrm{G}}}$ and $\overline{R}_{\tau_b\mathrm{G}}$) for the generalization sets.

Method/Metric	$\overline{Acc}_{\mathrm{G}}$	$\overline{R}_{Acc_{\mathrm{G}}}$	$\overline{MAE}_{\mathrm{G}}$	$\overline{R}_{MAE_{\mathrm{G}}}$	$\overline{AMAE}_{\mathrm{G}}$	$\overline{R}_{AMAE_{\mathrm{G}}}$	$\overline{\tau}_{b\mathrm{G}}$	$\overline{R}_{\tau_{b\mathrm{G}}}$
GPOR	0.666	5.40	0.392	5.25	0.534	4.90	0.577	5.20
ASAOR(C4.5)	0.600	6.50	0.485	7.00	0.688	7.00	0.413	7.50
KDLOR	*0.680*	4.20	*0.363*	*4.00*	*0.509*	*3.80*	*0.640*	*3.60*
POM	0.490	7.90	0.778	7.80	0.861	7.00	0.375	6.90
SVC	**0.683**	**3.60**	0.385	5.55	0.550	6.20	0.587	6.20
RED-SVM	0.676	4.15	0.367	*4.00*	0.519	4.10	0.624	4.00
SVOREX	0.667	5.05	0.382	4.75	0.538	4.90	0.607	5.00
SVORIM	0.672	4.40	0.376	4.05	0.538	4.80	0.609	4.20
SVR-PCDOC	0.678	*3.80*	**0.359**	**2.60**	**0.487**	**2.30**	**0.653**	**2.40**

RED-SVM, having similar mean MAE_{G}. $AMAE$ is a better alternative than MAE when the distribution of patterns is not balanced, and this is clearly the case for several datasets (see Table 1). The best values for mean $AMAE_{\mathrm{G}}$ and mean ranking are obtained by SVR-PCDOC, and the second ones are those reported by KDLOR. Finally, the $\tau_{b\mathrm{G}}$ is revealing the most clear differences. When using this metric, the best mean values and ranks are reported by SVR-PCDOC, followed by KDLOR.

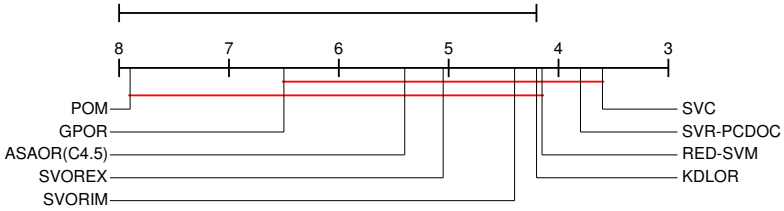
4.4 Statistical comparisons between methods

To quantify whether a statistical difference exists between any of these algorithms, a procedure for comparing multiple classifiers over multiple datasets is employed (Demšar, 2006). First of all, a Friedman's non-parametric test (Friedman, 1940) with a signifi-

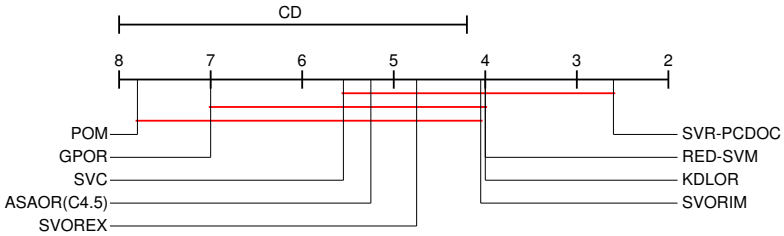
cance level of $\alpha = 0.05$ has been carried out to determine the statistical significance of the differences in the mean ranks of Table 4 for each different measure. The test rejected the null-hypothesis stating that the differences in mean rankings of Acc_{G}, MAE_{G}, $AMAE_{\mathrm{G}}$ and $\tau_{b\mathrm{G}}$ obtained by the different algorithms were statistically significant (with $\alpha = 0.05$). Specifically, the confidence interval for this number of datasets and algorithms is $C_0 = (0, \mathrm{F}_{(\alpha=0.05)} = 2.070)$, and the corresponding F-value for each metric were $3.257 \notin C_0$, $4.821 \notin C_0$, $4.184 \notin C_0$ and $5.099 \notin C_0$, respectively.

On the basis of this rejection, the Nemenyi post-hoc test is used to compare all classifiers to each other (Demšar, 2006). This test considers that the performance of any two classifiers is deemed significantly different if their mean ranks differ by at least the critical difference (CD), which depends on the number of datasets and methods. A 5% significance confidence was considered ($\alpha = 0.05$) to obtain this CD and the results can be observed in Figure 11, which shows CD diagrams as proposed in (Demšar, 2006). Each method is represented as a point in a raking scale, corresponding to its mean ranking performance. CD segments are included to measure the separation needed between methods in order to assess statistical differences. Red lines group algorithms were statistically different mean ranking performance cannot be assessed. Table 4 should also be considered when interpreting this graph.

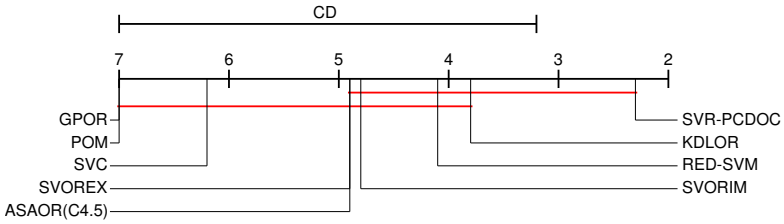
Figure 11a shows that SVC, i.e. the nominal classifier, has the best performance in Acc, this is, when not considering the order of the label prediction errors, and SVR-PCDOC has the second best one. RED-SVM, KDLOR and SVORIM have similar performance here. In Figure 11b, the best mean ranking is for SVR-PCDOC, and SVORIM, KDLOR and RED-SVM have similar performance. However, when con-



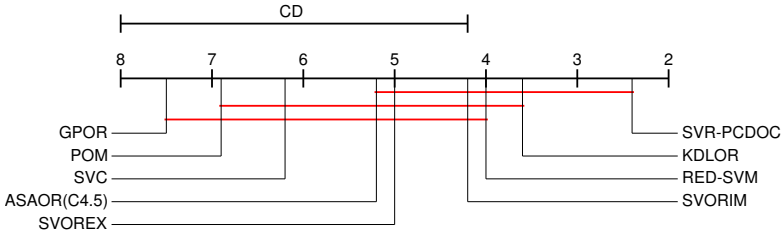
(a) Nemenyi CD diagram comparing generalization Acc mean rankings of the different methods.



(b) Nemenyi CD diagram comparing the generalization MAE mean rankings of the different methods.



(c) Nemenyi CD diagram comparing the generalization $AMAE$ mean rankings of the different methods.



(d) Nemenyi CD diagram comparing the generalization τ_b mean rankings of the different methods.

Figure 11: Ranking test diagrams for the mean generalization Acc, MAE, $AMAE$ and τ_b.

Table 5: Differences and Critical Difference (CD) value in rankings in the Bonferroni-Dunn test, using SVR-PCDOC as the control method.

	Method							
Metric	ASAOR(C4.5)	GPOR	KDLOR	POM	SVC	RED-SVM	SVOREX	SVORIM
Acc	1.60	2.70	0.40	4.10•	0.20	0.35	1.25	0.60
MAE	2.65	4.40•	1.40	5.20•	2.95	1.40	2.15	1.45
$AMAE$	2.60	4.70•	1.50	4.70•	3.90•	1.80	2.60	2.50
τ_b	2.80	5.10•	1.20	4.50•	3.80•	1.60	2.60	1.80

Bonferroni-Dunn Test: $CD_{(\alpha=0.05)}$= 3.336

•: Statistical difference with α = 0.05

sidering $AMAE$, it can be seen at Figure 11c that SVR-PCDOC mean ranking distance to the other methods increases, specifically for RED-SVM and SVORIM. Finally, Figure 11d shows the mean rank CD diagram for τ_b where SVR-PCDOC is still having the best mean performance.

It has been noticed that the Nemenyi approach comparing all classifiers to each other in a post-hoc test is not as sensitive as the approach comparing all classifiers to a given classifier (a control method) (Demšar, 2006). The Bonferroni-Dunn test allows this latter type of comparison and, in our case, it is done using the proposed method as the control method for the four metrics. The results of the Bonferroni-Dunn test for $\alpha = 0.05$ can be seen in Table 5, where the corresponding critical values are included. From the results of this test, it can be concluded than SVR-PCDOC does not report a statistically significant difference with respect to the SVM ordinal regression methods, KDLOR and ASAOR(C4.5), but it does when it is compared to POM for all the metrics and compared to GPOR for the ordinal metrics. Moreover, there are

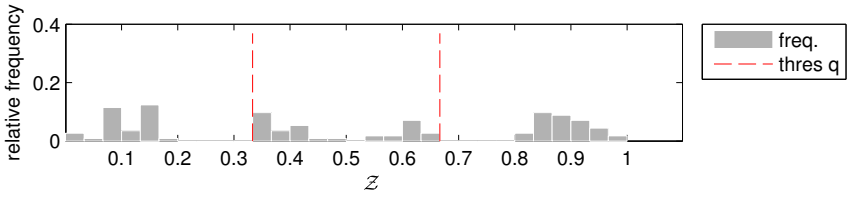
significant differences with respect to SVC, when considering $AMAE$ and τ_b.

From the above experiments, we can conclude that the reference (baseline) nominal classifier, SVC, is improved with statistical differences when considering ordinal classification measures. Regarding ASAOR(C4.5), SVOREX, SVORIM, KDLOR and RED-SVM, whereas the general performance is slightly better, there are no statistically significant differences favouring any of the methods.

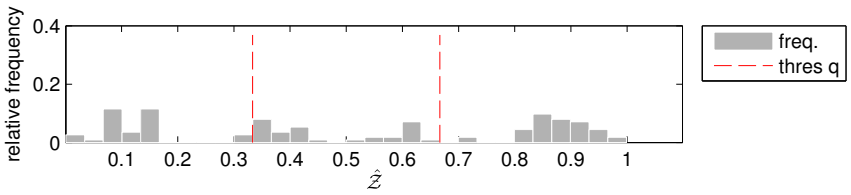
As a summary of the experiments, two important conclusions can be drawn about the performance measures: When imbalanced datasets are considered, Acc_{G} is clearly omitting important aspects of ordinal classification and so does MAE_{G}. If the comparative performance is taken into account, KDLOR and SVR-PCDOC appear to be very good classifiers when the objective is to improve $AMAE_{\mathrm{G}}$ and τ_{G}. The best mean ranking performance is obtained by our method proposed in this paper.

4.5 Latent space representations of the ordinal classes

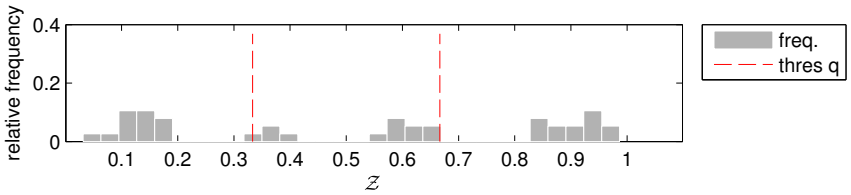
In the previous section we have shown that our simple and intuitive methodology can compete on equal footing with established more complex and/or less direct methods for ordinal classification. In this section we complement this performance based comparison with a deeper analysis of the main ingredient of our and other related approaches to ordinal classification - projection onto the one-dimesional (latent) space naturally representing the ordinal nature of the class organization. In particular, we study how non-linear latent variable models, SVR-PCDOC, KDLOR, SVOREX and SVORIM organize their one-dimensional latent space data projections. For comparison purposes, the latent variable $\mathcal{Z}$ values of the training and generalization data of the first fold of



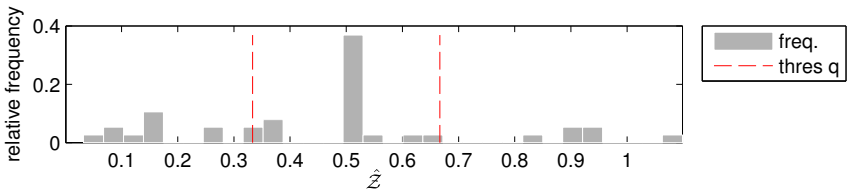
(a) SVR-PCDOC train PCD histogram.



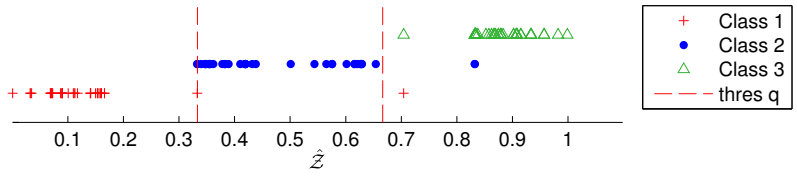
(b) SVR-PCDOC train $\hat{\mathcal{Z}}$ histogram.



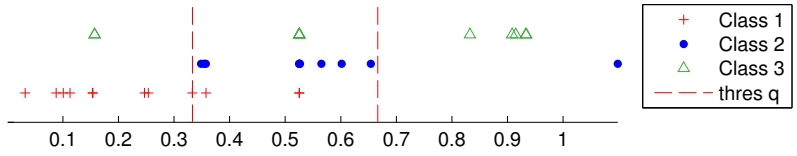
(c) SVR-PCDOC generalization PCD histogram.



(d) SVR-PCDOC generalization $\hat{\mathcal{Z}}$ histogram.



(e) SVR-PCDOC train $\hat{\mathcal{Z}}$ prediction.



(f) SVR-PCDOC generalization $\hat{\mathcal{Z}}$ prediction.

Figure 12: PCD projection and SVR-PCDOC's histograms and $\hat{\mathcal{Z}}$ prediction corresponding to the latent variable of the *tae* dataset: train PCD, train predicted $\hat{\mathcal{Z}}$ by SVR-PCDOC, generalization PCD and generalization predicted $\hat{\mathcal{Z}}$ by SVR-PCDOC. Generalization results are $Acc = 0.582$, $MAE = 0.457$, $AMAE = 0.455$, $\tau_b = 0.493$.

tae dataset are shown (Figure 12). Both histograms and values are plotted so that the behaviour of the models can be analysed. In the case of PCDOC, the PCD projection is also included to see whether the regressor model is close to the PCDOC projection. The histograms represent relative frequency of the projections. SVORIM histograms and latent variable values are not presented since they are similar to the SVOREX ones in the selected dataset.

We first analyse the SVR-PCDOC method. From PCD projections in Figure 12a we deduce that classes C_1 and C_2 contain patterns that are very close in the input space – projection of some patterns from C_2 lies near the threshold that divides the $\mathcal{Z}$ values for the two classes. Analogous comment applies to classes C_2 and C_3. The regressor seems to have learnt the imposed projection reasonably well since the predicted latent values have a histogram similar to the training PCD projection histogram. The generalization PCD projections (Figure 12c) have similar characteristics as the training ones[6]. Note the concentration of values around $\hat{z} = 0.5$ on the prediction of the generalization $\mathcal{Z}$. This concentration of values is due to wrong prediction of class C_1 and C_3 patterns that were both assigned to C_2. This behaviour can be better seen in Figures 12e and 12f, where the modelled latent value for each pattern is shown together with its class label. Indeed, during training some C_1 and C_2 patterns were mapped to positions near the thresholds. This is probably caused by noise or overlapping class distribution in the input space.

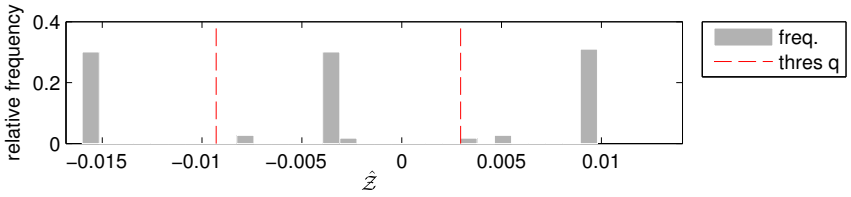
Figure 13 presents latent variable values of KDLOR. The KDLOR method projects

[6]There are much less patterns in the hold-out set than in the training set, making direct comparison of the two histograms problematic

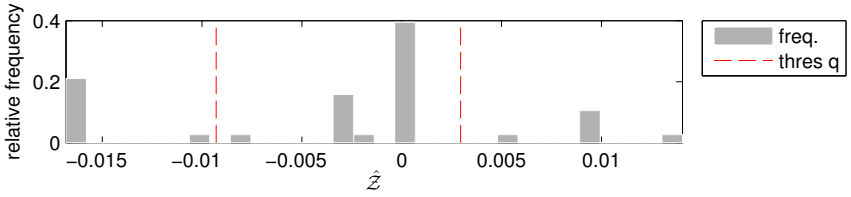
the data onto the latent space by minimizing the intra-class distance while maximizing the inter-class distance of the projections. As a result, the latent representations of the data are quite compact for each class (see training projection histogram in Figure 13a). While this philosophy often leads to superior classification results, the projections are not reflecting the structure of patterns within a single class, that is, the ordinal nature of the data is not fully captured by the model. In addition, KDLOR projections occur in the wrong bins more often than in the case of SVR-PCDOC (see generalization projections $\mathcal{Z}$ in Figure 13d)).

Finally, Figure 14 presents latent representations of patterns by the SVOREX model. As in the KDLOR case, (except for a few patterns) the training latent representations are highly compact within each class. Again, the relative structure of patterns within their classes is lost in the projections.

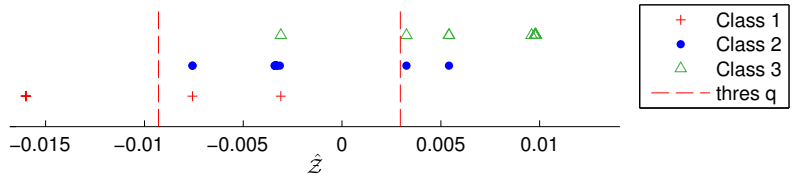
In both models, KDLOR and SVOREX, there is a pressure in the model construction phase to find 1-dimensional projections of the data that result in compact classes, while maximizing the inter-class separation. In the case of KDLOR this is explicitly formulated in the objective function. On the other hand, the key idea behind SVM based approaches is margin maximization. Data projections that maximize inter-class margins implicitly make the projected classes compact. We hypothesise that the pressure for compact within-class latent projections can lead to poorer generalization performance, as illustrated in Figure 14d. In the case of overlapping classes the drive for compact class projections can result in locally highly non-linear projections of the overlapping regions, over which we do not have a direct control (unlike in the case of PCDOC, where the non-linear projection is guided by the relative positions of points



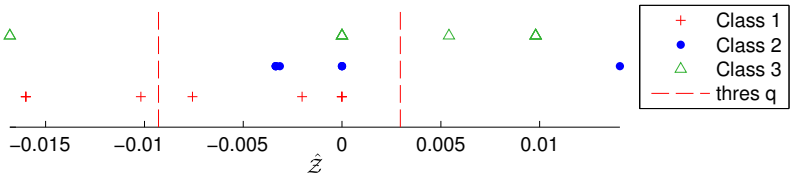
(a) KDLOR train $\hat{\mathcal{Z}}$ histogram.



(b) KDLOR generalization $\hat{\mathcal{Z}}$ histogram.



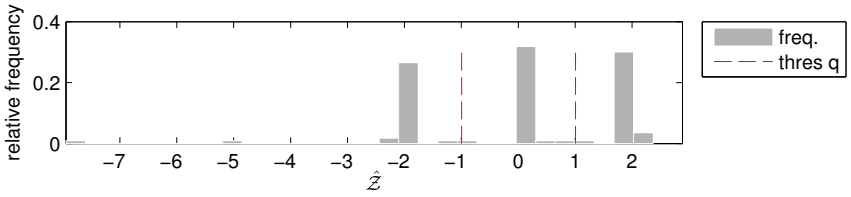
(c) KDLOR train $\hat{\mathcal{Z}}$ prediction.



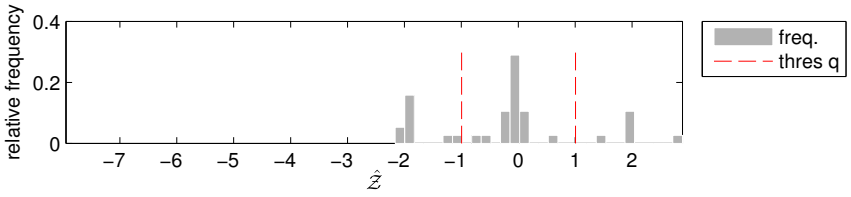
(d) KDLOR generalization $\hat{\mathcal{Z}}$ prediction.

Figure 13: Prediction of train and generalization $\hat{\mathcal{Z}}$ values corresponding to KDLOR at *tae* dataset. Generalization results are $Acc = 0.555$, $MAE = 0.473$, $AMAE = 0.471$, $\tau_b = 0.477$.

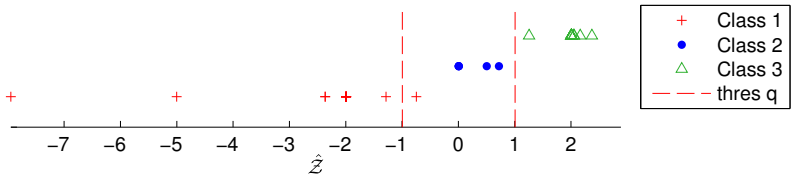
with respect to the other classes). Having such highly expanding projections can result in test points being projected to wrong classes in an arbitrary manner. Even though we provide detailed analysis for one dataset and one fold only, the observed tendencies were quite general across the data sets and hold-out folds.



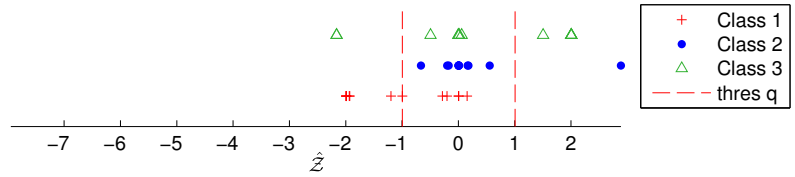
(a) SVOREX train $\hat{\mathcal{Z}}$ histogram.



(b) SVOREX generalization $\hat{\mathcal{Z}}$ histogram.



(c) SVOREX train $\hat{\mathcal{Z}}$ prediction.



(d) SVOREX generalization $\hat{\mathcal{Z}}$ prediction.

Figure 14: Prediction of train and generalization $\hat{\mathcal{Z}}$ values corresponding to SVOREX at *tae* dataset. Generalization results are $Acc = 0.581$, $MAE = 0.485$, $AMAE = 0.484$, $\tau_b = 0.445$

5 Conclusions

This paper addresses ordinal classification by proposing a projection of the input data into a one-dimensional variable, based on the relative position of each pattern with respect to the patterns of the adjacent classes. Our approach is based on a simple and

intuitive idea: instead of implicitly inducing a one dimensional data projection into a series of class intervals (as done in threshold based methods), construct such projection explicitly and in a controlled manner. Threshold methods crucially depend on such projections and we propose that it might be advantageous to have a direct control over how the projection is done, rather than having to rely on its indirect induction through a one-stage ordinal classification learning process.

Applying this one-dimensional projection on the training set yields data on which generalized projection can be trained using any standard regression method. The generalized projection can in turn be applied to new instances which are then classified based to the interval into which their projection falls.

We construct the projection by imposing that the *'best separated'* pattern of each class (i.e. the pattern most distant from the adjacent classes) should be mapped to the centre of the interval representing that class (or in the interval extremes for the extreme, first and the last, classes). All the other patterns are proportionally positioned in their corresponding class intervals around the centres mentioned above. We designed a projection method having such desirable properties and empirically verified its appropriateness on datasets with linear and non-linear class ordering topologies.

We extensively evaluated our method on ten real-world datasets, four performance metrics, a measure of statistical significance and performed comparison with eight alternative methods, including the most recent proposals for ordinal regression and a baseline nominal classifier. In spite of the intrinsic simplicity and straightforward intuition behind our proposal, the results are competitive and consistent with respect to the state-of-the-art in the literature. The mean ranking performance of our method was par-

ticularly impressive, when robust ordinal performance metrics were considered, such as the average mean absolute error or the τ_b correlation coefficient. Moreover, we studied in detail the latent space organization of the projection based methods considered in this paper. We suggest, that while the pressure for compact within class latent projections can make training sample projections nicely compact within classes, it can lead to poorer generalization performance overall.

We also identify some interesting discussion points. Firstly, the latent space thresholds are fixed by the projection with an equal width. This may be interpreted as an assumption of equal widths for each class, which is not always true for all the problems. This would indeed be a problem if we used a linear regressor from the data space to the projection space. However, we employ non-linear projections and the adjustment for unequal 'widths' of the different classes can be naturally achieved within such non-linear mapping from the data to the projection space. Actually, from the model fitting standpoint, having fixed-width class regions in the projection space is desirable. Allowing for variable widths would increase the number of free parameters and would make the free parameters dependent in potentially complicated manner (flexibility of projections versus class widths in the projection space). This may have harmful effect on model fitting, especially if the data set is of limited size. Having less free parameters is also advantageous from the point of view of computational complexity.

The second discussion point is the possible undesirable influence of outliers in the PCD projection. One possible solution can be to place each pattern in the projection considering more classes than just the adjacent ones. However, this idea should be done carefully in order not to decrease the role of ordinal information in the projection. A

direct alternative can be to use k-NN like scheme in Eq. (2), where instead of taking the minimum distance to a point, the average distance to the k closest points of class $q \pm 1$ can be used. This will represent a generalization of the current scheme that calculates distances with $k = 1$. Nevertheless, the inclusion of k would imply the addition of a new free parameter to the training process.

In conclusion, the results indicate that our two-phase approach to ordinal classification is a viable and simple-to-understand alternative to the state-of-art. The projection constructed in the first phase is consistently extracting useful information for ordinal classification. As such it can not only be used as the basis for classifier construction, but also as a starting point for devising measures able to detect and quantify possible ordering of classes in any dataset. This is a matter for our future research.

Acknowledgments

This work has been partially subsidized by the TIN2011-22794 project of the Spanish Ministerial Commision of Science and Technology (MICYT), FEDER funds and the P08-TIC-3745 project of the "Junta de Andalucía" (Spain). The work of Peter Tiňo was supported by a BBSRC grant (no. BB/H012508/1).

References

Agresti, A. (1984). *Analysis of ordinal categorical data*. New York: Wiley.

Arens, R. (2010). Learning SVM Ranking Functions from User Feedback Using Docu-

ment Metadata and Active Learning in the Biomedical Domain. In Fürnkranz, J. and Hüllermeier, E., editors, *Preference Learning*, pages 363–383. Springer-Verlag.

Asuncion, A. and Newman, D. (2007). UCI machine learning repository.

Baccianella, S., Esuli, A., and Sebastiani, F. (2009). Evaluation measures for ordinal regression. In *Proceedings of the Ninth International Conference on Intelligent Systems Design and Applications (ISDA'09)*, pages 283–287.

Barker, D. (1995). Pasture Production dataset. Obtained on October 2011.

Cardoso, J., Pinto da Costa, J., and Cardoso, M. (2005). Modelling ordinal relations with SVMs: an application to objective aesthetic evaluation of breast cancer conservative treatment. *Neural Networks*, 18(5-6):808–817.

Cardoso, J. S. and Pinto da Costa, J. F. (2007). Learning to classify ordinal data: The data replication method. *Journal of Machine Learning Research*, 8:1393–1429.

Cardoso, J. S. and Sousa, R. (2011). Measuring the performance of ordinal classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(8):1173–1195.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2:27:1–27:27.

Chu, W. and Ghahramani, Z. (2005). Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 6:1019–1041.

Chu, W. and Keerthi, S. S. (2005). New approaches to support vector ordinal regression. In *In ICML'05: Proceedings of the 22nd international conference on Machine Learning*, pages 145–152.

Chu, W. and Keerthi, S. S. (2007). Support Vector Ordinal Regression. *Neural Computation*, 19(3):792–815.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Crammer, K. and Singer, Y. (2001). Pranking with ranking. In *Advances in Neural Information Processing Systems 14*, pages 641–647. MIT Press.

Crammer, K. and Singer, Y. (2005). Online ranking by projecting. *Neural Computation*, 17(1):145–175.

Cruz-Ramírez, M., Hervás-Martínez, C., Sánchez-Monedero, J., and Gutiérrez, P. A. (2011). A Preliminary Study of Ordinal Metrics to Guide a Multi-Objective Evolutionary Algorithm. In *Proceedings of the 11th International Conference on Intelligent Systems Design and Applications (ISDA 2011)*, pages 1176–1181, Cordoba, Spain.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30.

Fouad, S. and Tiňo, P. (2012). Adaptive metric learning vector quantization for ordinal classification. *Neural Computation*, 24(11):2825–2851.

Frank, E. and Hall, M. (2001). A simple approach to ordinal classification. In *Pro-*

ceedings of the 12th European Conference on Machine Learning, EMCL '01, pages 145–156, London, UK. Springer-Verlag.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *Annals of Mathematical Statistics*, 11(1):86–92.

Gutiérrez, P., Salcedo-Sanz, S., Hervás-Martínez, C., Carro-Calvo, L., Sánchez-Monedero, J., and Prieto, L. (2013). Ordinal and nominal classification of wind speed from synoptic pressure patterns. *Engineering Applications of Artificial Intelligence*, 26(3):1008–1015.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: an update. *Special Interest Group on Knowledge Discovery and Data Mining Explorer Newsletter*, 11:10–18.

Herbrich, R., Graepel, T., and Obermayer, K. (2000). Large margin rank boundaries for ordinal regression. In Smola, A., Bartlett, P., Schölkopf, B., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*, pages 115–132, Cambridge, MA. MIT Press.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multi-class support vector machines. *IEEE Transaction on Neural Networks*, 13(2):415–425.

Hühn, J. C. and Hüllermeier, E. (2008). Is an ordinal class structure useful in classifier learning? *Int. J. of Data Mining, Modelling and Management*, 1(1):45–67.

Kendall, M. G. (1962). *Rank Correlation Methods*. New York: Hafner Press, 3rd edition.

Kibler, D. F., Aha, D. W., and Albert, M. K. (1989). Instance-based prediction of real-valued attributes. *Computational Intelligence*, 5:51.

Kim, K.-j. and Ahn, H. (2012). A corporate credit rating model using multi-class support vector machines with an ordinal pairwise partitioning approach. *Computers & Operations Research*, 39(8):1800–1811.

Kotsiantis, S. B. and Pintelas, P. E. (2004). A cost sensitive technique for ordinal classification problems. In Vouros, G. and Panayiotopoulos, T., editors, *Methods and applications of artificial intelligence (Proceedings of the Third Hellenic Conference on Artificial Intelligence, SETN 2004)*, volume 3025 of *Lecture Notes in Artificial Intelligence*, pages 220–229.

Kramer, S., Widmer, G., Pfahringer, B., and de Groeve, M. (2010). Prediction of ordinal classes using regression trees. In Ras, Z. and Ohsuga, S., editors, *Foundations of Intelligent Systems*, volume 1932 of *Lecture Notes in Computer Science*, pages 665–674. Springer Berlin / Heidelberg.

Li, L. and Lin, H.-T. (2007). Ordinal regression by extended binary classification. In *Advances in Neural Information Processing Systems*, pages 865–872.

Lim, T.-S., Loh, W.-Y., and Shih, Y.-S. (2000). A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms. *Machine Learning*, 40:203–228.

Lin, H.-T. and Li, L. (2012). Reduction from cost-sensitive ordinal ranking to weighted binary classification. *Neural Computation*, 24(5):1329–1367.

Mackay, D. J. C. (1994). Bayesian methods for backpropagation networks. In Domany, E., van Hemmen, J. L., and Schulten, K., editors, *Models of Neural Networks III*, chapter 6, pages 211–254. Springer-Verlag, New York.

McCullagh, P. (1980). Regression models for ordinal data. *Journal of the Royal Statistical Society. Series B (Methodological)*, 42(2):109–142.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Springer-Verlag, New York, Secaucus, NJ, USA.

Perez-Ortiz, M., Gutierrez, P. A., Garcia-Alonso, C., Salvador-Carulla, L., Salinas-Perez, J. A., and Hervas-Martinez, C. (2011). Ordinal classification of depression spatial hot-spots of prevalence. In *11th International Conference on Intelligent Systems Design and Applications (ISDA)*, pages 1170 –1175.

Pinto da Costa, J. F., Alonso, H., and Cardoso, J. S. (2008). The unimodal model for the classification of ordinal data. *Neural Networks*, 21:78–91.

Raykar, V. C., Duraiswami, R., and Krishnapuram, B. (2008). A fast algorithm for learning a ranking function from large-scale data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1158–1170.

Sánchez-Monedero, J., Gutiérrez, P. A., Fernández-Navarro, F., and Hervás-Martínez, C. (2011). Weighting efficient accuracy and minimum sensitivity for evolving multi-class classifiers. *Neural Processing Letters*, 34(2):101–116.

Schölkopf, B. and Smola, A. J. (2001). *Learning with Kernels: Support Vector Ma-*

chines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning). The MIT Press, 1st edition.

Shashua, A. and Levin, A. (2002). Ranking with Large Margin Principle: Two Approaches. In *NIPS*, pages 937–944. MIT Press.

Sonnenburg, D. S. (2011). Machine Learning Data Set Repository.

Sun, B.-Y., Li, J., Wu, D. D., Zhang, X.-M., and Li, W.-B. (2010). Kernel discriminant learning for ordinal regression. *IEEE Transactions on Knowledge and Data Engineering*, 22(6):906–910.

Vapnik, V. (1999). An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999.

Verwaeren, J., Waegeman, W., and De Baets, B. (2012). Learning partial ordinal class memberships with kernel-based proportional odds models. *Computational Statistics & Data Analysis*, 56(4):928–942.

Waegeman, W. and Boullart, L. (2009). An ensemble of weighted support vector machines for ordinal regression. *International Journal of Computer Systems Science and Engineering*, 3(1):47–51.

Waegeman, W. and De Baets, B. (2011). A Survey on ROC-based Ordinal Regression. In Fürnkranz, J. and Hüllermeier, E., editors, *Preference Learning*, pages 127–154. Springer Berlin Heidelberg.