Ordinal regression methods: survey and experimental study

Pedro Antonio Gutiérrez, Member, IEEE, María Pérez-Ortiz, Javier Sánchez-Monedero, Francisco Fernández-Navarro, and César Hervás-Martínez, Senior Member, IEEE

Abstract—Ordinal regression problems are those machine learning problems where the objective is to classify patterns using a categorical scale which shows a natural order between the labels. Many real-world applications present this labelling structure and that has increased the number of methods and algorithms developed over the last years in this field. Although ordinal regression can be faced using standard nominal classification techniques, there are several algorithms which can specifically benefit from the ordering information. Therefore, this paper is aimed at reviewing the state of the art on these techniques and proposing a taxonomy based on how the models are constructed to take the order into account. Furthermore, a thorough experimental study is proposed to check if the use of the order information improves the performance of the models obtained, considering the most significant published approaches within the taxonomy. The results confirm that ordering information benefits ordinal models improving their accuracy and the closeness of the predictions to actual targets in the ordinal scale.

Index Terms—Ordinal regression, ordinal classification, binary decomposition, threshold methods, augmented binary classification, proportional odds model, support vector machines, discriminant learning, artificial neural networks

1 INTRODUCTION

Learning to classify or to predict numerical values from prelabelled patterns is one of the central research topics in machine learning and data mining [1–3]. However, less attention has been paid to ordinal regression (also called ordinal classification) problems, where the labels of the target variable exhibit a natural ordering. For example, student satisfaction surveys usually involve rating teachers based on an ordinal scale {poor, average, good, very good, excellent}. Hence, class labels are imbued with order information, e.g. a sample vector associated with class label average has a higher rating (or better) than another from the poor class, but good class is better than both. When dealing with this kind of problems, two facts are decisive: misclassification costs are not the same for different errors (it is clear that misclassifying an excellent teacher as poor should be more penalised than misclassifying him/her as very good) and the ordering information can be used to construct more accurate models. A further distinction is made by Anderson [4], which differentiates two major types of ordinal categorical variables, “grouped continuous variables” and “assessed ordered categorical variables”. The first one is a discretised version of an underlying continuous variable, which could be observed itself. The second one covers those variables where a user provides his/her judgement on the grade of the ordered categorical variable. However, imposing an ordering is meaningful for both cases.

Ordinal regression problems are very common in many research areas, and they have been frequently considered as standard nominal problems which can lead to non-optimal solutions. Indeed, ordinal regression problems can be said to be between classification and regression, presenting some similarities and differences. Some of the fields where ordinal regression is found are medical research [5–11], age estimation [12], brain computer interface [13], credit rating [14–17], econometric modelling [18], face recognition [19–21], facial beauty assessment [22], image classification [23], wind speed prediction [24], social sciences [25], text classification [26], and more. All these works are examples of application of specifically designed ordinal regression models, where the ordering consideration improves their performance with respect to their nominal counterparts.

In statistics, ordinal data were firstly studied by using a link function able to model the underlying probability for generating ordinal labels [4]. The field of ordinal regression has evolved in the last decade, with a plethora of noteworthy research progress made in supervised learning [27], from support vector machine (SVM) formulations [28, 29] to Gaussian processes [30] or discriminant learning [31], to name a few. However, up to the authors’ knowledge, these methods have not yet been categorised in a general taxonomy, which is essential for further research and for identifying the developments made and the present state of existing methods. This paper contributes a review of the state-of-the-art of ordinal regression, a taxonomy proposal to
better organise the advances in this field, and an experimental study with a complete repository of datasets and a total of 16 ordinal regression methods (including a software tool to run and test all the methods).

Several objectives motivate the experimental study. First of all, our focus is on evaluating the necessity of taking ordering information into account. In [32], ordinal meta-models were compared with respect to their nominal counterparts to check their ability to exploit ordinal information. The work concludes that such meta-methods do exploit ordinal information and may yield better performance. However, as will be analysed in this work, specifically designed ordinal regression methods can further improve the results with respect to meta-model approaches. Another study [33] argues that ordinal classifiers may not present meaningful advantages over the analogue non-ordinal methods, based on accuracy and Cohen’s Kappa statistic [34]. The results of the present review show that statistically significant differences are found when using measures which take the order into account, which is the case of the Mean Absolute Error (MAE), i.e. the average deviation between predicted and actual targets in number of categories. The second main motivation of this paper is to provide some guidelines to decide on the best methods in terms of accuracy, MAE and computational time. Since there are not specific repositories of ordinal regression datasets, proposals are usually evaluated using discretised regression ones, where the target variable is simply divided into different bins or classes. 24 of these discretised datasets are used for our study, in addition to 17 real benchmark ordinal regression datasets extracted from public repositories. The last objective is to evaluate whether the methods behave differently depending on the nature of the datasets.

This paper is a significant extension of a preliminary conference version [35]: a deeper analysis of the state-of-the-art has been performed, including most recent proposals and a taxonomy to group them. Moreover, the experimental study includes more methods and datasets. The rest of the paper is organised as follows. Section 2 introduces the problem of ordinal regression and briefly describes its differences from some related machine learning topics outside the scope of this paper. Section 3 revises ordinal regression state-of-the-art by grouping different methods with a proposed taxonomy. The main representatives of each family are then empirically compared in Section 4 where the experiments are described and the corresponding results are studied and discussed. Finally, Section 5 deals with the main achievements.

2 Notation and nature of the problem
2.1 Problem definition

The ordinal regression problem consists on predicting the label \( y \) of an input vector \( x \), where \( x \in \mathcal{X} \subseteq \mathbb{R}^K \) and \( y \in \mathcal{Y} = \{C_1, C_2, \ldots, C_Q\} \), i.e. \( x \) is in a \( K \)-dimensional input space and \( y \) is in a label space of \( Q \) different labels.

These labels form categories or groups of patterns, and the objective is to find a classification rule or function \( f : \mathcal{X} \rightarrow \mathcal{Y} \) to predict the categories of new patterns, given a training set of \( N \) points, \( D = \{(x_i, y_i), i = 1, \ldots, N\} \). A natural label ordering is included for ordinal regression, \( C_1 \prec C_2 \prec \ldots \prec C_Q \), where \( \prec \) is an order relation given by the nature of the classification problem. Many ordinal regression measures and algorithms consider the rank of the label, i.e. the position of the label in the ordinal scale, which can be expressed by the function \( O(\cdot) \), in such a way that \( O(C_q) = q \), \( q = 1, \ldots, Q \). The difference between this setting and other related ones is now established. The assumption of an order between class labels makes that two different elements of \( \mathcal{Y} \) can be always compared by using the relation \( \prec \), which is not possible under the nominal classification setting. If compared to regression (where \( y \in \mathbb{R} \)), it is true that real values in \( \mathbb{R} \) can be ordered by the standard \( < \) operator, but labels in ordinal regression (\( y \in \mathcal{Y} \)) do not carry metric information, so the category serves as a qualitative indication of the pattern rather than a quantitative one.

2.2 Ordinal regression in the context of ranking and sorting

Although ordinal regression has been paid attention recently, the amount of related research topics is worth to be mentioned. First, it is important to remark the differences between ordinal regression and other related ranking problems. There are three terms to be clarified: ranking, sorting and multipartite ranking.

Ranking generally refers to those problems where the algorithm is given a set of ordered labels [36], with one label for each pattern, and the objective is to learn a rule able to rank patterns by using this discrete set of labels. The induced ordering should be partial with respect to the patterns, in the sense that ties are allowed. This rule should be able to obtain a good ranking, but not to classify patterns in the correct class. For example, if the labels predicted by a classifier are shifted one category (in the ordinal scale) with respect to the actual ones, the classifier will still be a perfect ranker.

Another term, sorting [36] is referred to the problem where the algorithm is given a total order for the training dataset and the objective is to rank new sets during the test phase. As we can see, this is equivalent to a ranking problem where the size of the label set is equal to the number of training points, \( Q = N \). Ties are not allowed for the prediction. Again, the interest is in learning a function that can give a total ordering of the patterns instead of a concrete label.

The multipartite ranking problem is a generalisation of the well-known bipartite ranking one. Multipartite ranking can be seen as an intermediate point between ranking and sorting. It is similar to ranking because training patterns are labelled with one of \( Q \) ordered ratings (\( Q = 2 \) for bipartite ranking), but here the goal is to learn from them a ranking function able to induce a total order
in accordance with the given training ratings \([37]–[39]\), which is similar to sorting. The objective of multivariate ranking is to obtain a ranking function which ranks “high” classes ahead of “low” classes (in the ordinal scale), being this a refinement of the order information provided by an ordinal classifier, as the latter does not distinguish between objects within the same category. ROC analysis, which evaluates the ability of classifiers to sort positive and negative instances in terms of the area under the ROC curve, is a clear example of training a binary classifier to perform well in a bipartite ranking problem. The relationship between multivariate ranking and ordinal classification is discussed in [38]. An ordinal regression classifier can be used as a ranking function by interpreting the class labels as scores. However, this type of scoring will produce a large number of ties (which is not desirable for multivariate ranking). On the other hand, a multivariate ranking function \(f(\cdot)\) can be turned into an ordinal classifier by deriving thresholds to define an interval for each class, but how to find the optimal thresholds is an open issue.

A more general term is learning to rank, gathering different methods in which the goal is to automatically construct a ranking model from training data [40]. Methods used for the three previously mentioned problems can be used for learning to rank ones. Moreover, ordinal regression can be used as a learning to rank algorithm, where the categories are individually evaluated for each training pattern, using a finite ordinal scale. In this context, we refer now to the categorisation presented in [40], which establishes different families of ranking model structures: pointwise or itemwise ranking (where the relevance of an input vector \(x\) is predicted by using either real-valued scores or ordinal labels), pairwise ranking (where the relative order between two input vectors \(x\) and \(x'\) is tried to be predicted, i.e. the local comparison nature of ranking, which can be easily cast to binary classification) and listwise ranking (where the algorithms try to order a finite set of patterns \(S = \{x_1, x_2, \ldots, x_N\}\) by minimising the inconsistency between the predicted permutation and the training permutation). Ordinal regression methods are pointwise ranking models, where each vector is assigned an ordinal label in order to rank it. In this way, they can be used for ranking as an alternative to both pairwise and listwise structures, which have serious problems of scalability with the size of the training dataset [41], the former needing to examine all pairs of patterns and the latter considering all possible permutations of the training data.

In summary, ordinal regression is a pointwise approach to classify data, where the labels exhibit a natural order. It is related to the problems of ranking, sorting and multivariate ranking, but, during the test phase, its objective is to obtain correct labels or labels as close as possible to the correct ones, not a correct relative partial order of the patterns (ranking), a total order of patterns in accordance to the order of the training set (sorting) or a total order in accordance to the training labels (multipartite ranking).

## 2.3 Advanced related topics

In this section, other advanced methods related to ordinal regression are surveyed. They are outside the scope of this paper, as they consider different learning settings.

Monotonic classification [42]–[44] is a special class of ordinal classification task, where there are monotonicity constraints between features and decision classes, i.e. \(x \geq x' \Rightarrow f(x) \geq f(x')\) [45], where \(x \geq x'\) means that \(x\) dominates \(x'\), i.e. \(x_k \geq x'_k, k = 1, \ldots, K\). Monotonic classification tasks are very common in real-world problems [43] (e.g. consider the case where employers must select their employees based on their education and experience), where monotonicity may be an important model requirement for justifying the decision made. This kind of problems have been approached, for example, by decision trees [43], [46] and rough set theory [44].

A recent work is concerned with transductive ordinal regression [27], where a SVM model is derived to learn from a set of labelled and unlabelled patterns. The core of their formulation is an objective function that caters to several commonly used loss functions in transductive settings, but for ordinal regression. This SVM model is combined with a proposed label swapping scheme for multiple class transduction to derive ordinal decision boundaries that pass through a low-density region of the augmented labelled and unlabelled data. Another related work [47] considers transfer learning in the same context, where the objective is to obtain a classifier for new target domains using the available label information of other related source domains. The proposed method spans the feasible solution space with an ensemble of ordinal classifiers from the multiple relevant source domains, using the maximum margin criterion.

Uncertainty has been included in ordinal regression models in two different ways. Nondeterministic ordinal classifiers (defined as those allowed to predict more than one label for some patterns) are considered in [48]. In [49] a kernel model is proposed for those ordinal problems where partial class memberships probabilities are available instead of crisp labels.

One step forward [50] considers those problems where the prediction labels follow a circular order (e.g. directional predictions).

## 3 An Ordinal Regression Taxonomy

In this section, a taxonomy of ordinal regression methods is proposed. With this purpose we firstly review what have been referred to as naïve approaches, in the sense that the model is obtained by using other standard machine learning prediction algorithms (e.g. nominal classification or standard regression). Secondly, ordinal binary decomposition approaches are reviewed, the main idea being to decompose the ordinal problem into several binary ones, which are separately solved by multiple models or by one multiple-output model. The third group will
include the set of methods known as threshold models, which are based on the general idea of approximating a real value predictor and then dividing the real line into intervals. The taxonomy proposed is given in Fig. 1.

### 3.1 Naïve approaches

Ordinal regression problems can be easily simplified into other standard problems, which generally involves making some assumptions. As will be later discussed, these methods can be very competitive given that, even though these assumptions may not hold, they inherit the performance of very well-tuned models.

#### 3.1.1 Regression

One idea is to cast all the different labels \( \{C_1, C_2, \ldots, C_Q\} \) into real values \( \{r_1, r_2, \ldots, r_Q\} \) \[51\], where \( r_i \in \mathbb{R} \), and then to apply standard regression techniques \[2\], \[52\], \[53\] (such as neural networks, support vector regression...). Typically, the value of each label is related to its position in the ordinal scale, i.e. \( r_i = i \). For example, Kramer et al. \[54\] map the ordinal scale by assigning numerical values, applying a regression tree model and rounding the results for assigning the class when predicting new values. They also evaluate the possibility of using the median, the mode, or the rounded mean of all the patterns in the leaves of the tree. The main problem with these approaches is that real values used for the labels may hinder the performance of the regression algorithms, and there is no principled way of deciding the value a label should have without prior information about the problem, since the distance between classes is unknown. Moreover, regression learners will be more sensitive to the representation of the label rather than its ordering \[55\]. A recent alternative is proposed in \[56\], where, instead of choosing arbitrary ordered values for the different labels, the variable is reconstructed by examining the different pairwise class distances.

#### 3.1.2 Nominal classification

Ordinal classification problems are usually considered from a standard nominal perspective, and the order between classes is simply ignored. Some researchers routinely apply nominal response data analysis methods (yielding results invariant to the permutation of the categories) to both nominal and ordinal target variables alike because they are both categorical \[57\]. Nominal classification algorithms ignore the ordering of the labels, thus requiring more training data \[55\]. The Support Vector Machine paradigm (SVM) \[58\] is perhaps the most common kernel learning method for statistical pattern recognition. Beyond the application of the kernel trick to allow non-linear decision discriminants, and the slack-variables to avoid inseparability, relax the constraints and handle noisy data, the original binary SVM had to be reformulated to deal with multiclass problems \[59\].

#### 3.1.3 Cost-sensitive classification

A more advanced method that can be considered in this group is cost-sensitive learning. Many real-world applications of machine learning and data mining require the evaluation of the learned system with different costs for different types of misclassification errors \[60\]. This is the case with ordinal regression, although the exact costs for misclassification can not be always evaluated a priori. The cost of misclassifications can be forced to be different depending on the distance between real and predicted classes, in the ordinal scale. The work of Kotsiantis and Pintelas \[61\] considers cost-sensitive classification, by using absolute costs (i.e. the element \( c_{ij} \) of the cost matrix \( C \) is equal to the difference in the number of categories, \( c_{ij} = |i - j| \)). Different algorithms are shown to obtain better \( MAE \) values when cost matrices are used, without harming (in fact even improving) accuracy \[61\]. We include two cost matrices for a five class problem in Table 1, with the absolute cost matrix and the quadratic cost \( c_{ij} = (i - j)^2 \), together with a zero-one cost matrix, which is the one assumed in nominal classification. Other possibilities are to choose asymmetric costs or non-convex two-Gaussian cost \[41\]. Again, the main problem is that, without a priori knowledge of the ordinal regression problem, it is not clear which cost matrix is more suitable.

**Fig. 1.** Proposed taxonomy for ordinal regression methods

**TABLE 1**

| Example of different cost matrices for a five class classification problems, with class labels \( y \in \mathcal{Y} = \{C_1, C_2, C_3, C_4, C_5\} \). |
|---|---|---|
| Zero-one | Absolute cost | Quadratic cost |
| \([1 1 1 1 1]\) | \([0 1 2 3 4]\) | \([0 1 4 9 16]\) |
| \([1 1 0 1 1]\) | \([1 0 1 2 3]\) | \([1 0 1 4 9]\) |
| \([1 1 1 0 1]\) | \([2 1 0 1 2]\) | \([4 1 0 1 4]\) |
| \([1 1 1 1 0]\) | \([3 2 1 0 1]\) | \([9 4 1 0 1]\) |
| \([1 1 1 1 1]\) | \([4 3 2 1 0]\) | \([16 9 4 1 0]\) |

Actual class labels are arranged in rows, while predicted class labels are arranged in columns.
TABLE 2

Binary decompositions for a 5-class ordinal problem, with class labels \( y \in Y = \{C_1, C_2, C_3, C_4, C_5\} \).

<table>
<thead>
<tr>
<th>Nominal decompositions</th>
<th>Ordinal decompositions</th>
</tr>
</thead>
<tbody>
<tr>
<td>OneVsAll</td>
<td>OrderedPartitions</td>
</tr>
<tr>
<td>(– + + + +)</td>
<td>OneVsNext</td>
</tr>
<tr>
<td>( – + + + +)</td>
<td>OneVsFollowers</td>
</tr>
<tr>
<td>(– + + + +)</td>
<td>OneVsPrevious</td>
</tr>
<tr>
<td>( – + + + +)</td>
<td></td>
</tr>
</tbody>
</table>

3.2 Ordinal binary decompositions

This group includes all those methods which are based on decomposing the ordinal target variable into several binary ones, which are then estimated by a single or multiple models. A summary of the decompositions is given in Table 2 where five classes are considered, each method generating a different decomposition matrix. Columns of the matrix correspond to the binary subproblems and rows to the role of each class for each subproblem. The symbol + is associated to the positive class and the symbol – to the negative one. If the class is not used in the specific binary subproblem, no symbol is included in the corresponding position. OneVsAll and OneVsOne formulations are nominal classification methods (and should be listed as naïve approaches), but they have been included in this table for comparison purposes. Note the high number of binary decompositions needed by OneVsOne (in this case, 10 combinations).

Two main issues have to be taken into account when analysing the methods herein presented: 1) some of them are based on the idea of training a different model for each subproblem (multiple model approaches), while others learn one single model for all the subproblems; 2) apart from defining how to decompose the problem, it is important to define a rule for predicting new patterns, once the decision values are obtained. For the prediction phase, the corresponding binary codes of Table 2 can be considered as part of the error-correcting output codes (ECOC) framework [62], where the predicted class is the one closest to the code formed by all binary responses. Taking the first criterion into account, we have divided ordinal binary decomposition algorithms into multiple model and multiple-output single model approaches.

3.2.1 Multiple model approaches

Ordinal information gives us the possibility of comparing the different labels. For a given rank \( q \), a direct question can be the following, “Is the label of pattern \( x \) greater than \( q \)?” [41]. This question is clearly a binary classification problem, so ordinal classification can be solved by considering each binary classification problem independently and combining the binary outputs into a label, which is the approach followed by Frank and Hall in [63] (this decomposition is called OrderedPartitions in Table 2). In their work, Frank and Hall considered C4.5 as the binary classifier and the decision of the different binary classifiers were combined by using associated probabilities \( p_q = P(y = \sum q | x), q = 1, \ldots, Q - 1 \):

\[
P(y = C_1 | x) = 1 - p_1, P(y = C_Q | x) = p_Q - 1,
\]

\[
P(y = C_q | x) = p_{q-1} - p_q, 2 \leq q \leq Q - 1.
\]

Note that this approach may lead to negative probability estimates [64], given that binary classifiers are independently learned and nothing assures that \( p_{q-1} < p_q \). When there is no need for proper probability estimations, prediction can be done by selecting the maximum.

In the work of Waegeman et al. [65], this framework is used but explicit weights over the patterns of each binary system are imposed, in such a way that errors on training objects are penalised proportionally to the absolute difference between their rank and \( q \) (the category examined). Additionally, labels for the test set are obtained by combining the estimated outcomes \( y_q \) of all the \( Q-1 \) binary classifiers. The interpretation of these binary outcomes \( y_qi \in \{+1, -1\}, q = 1, \ldots, Q-1, i = 1, \ldots, N \), intuitively leads to \( y_i > C_q \) if \( y_qi = +1 \). In this way, the rank \( k \) is assigned to pattern \( x_i \) so that \( y_qi = -1, \forall q < k \), and \( y_qi = +1, \forall q \geq k \). As stated by the authors, this strategy can result in ambiguities for some test patterns, and they should be solved by using similar techniques to those considered for nominal classification. A very similar scheme is proposed in [12], where the weights are obtained slightly differently, and different kernels are used for the different binary classification sub-problems.

Other ordinal binary decompositions can be found in the literature. The cascade linear utility model [66] considers \( Q-1 \) projections, in such a way that projection \( q \) separates classes \( C_1 \cup \ldots \cup C_{Q-q} \) from class \( C_{Q-q} \), i.e. one class is eliminated for each projection (this is the OneVsPrevious decomposition in Table 2). The predictions are then combined by a union utility function. Finally, binary SVMs were also applied to ordinal regression [15], by making use of the ordinal pairwise partitioning approach [14]. This approach is composed of four different reformulations of the classical OneVsOne and OneVsAll paradigms. OneVsNext considers that each binary classifier \( q \) separates class \( C_q \) from class \( C_{q+1} \), and OneVsFollowers (which is similar to the OneVsPrevious approach in [66] but in the opposite direction) constructs each binary classifier \( q \) for the task of separating class \( C_q \) from classes \( C_{q+1} \cup \ldots \cup C_Q \). The prediction phase is then approached by examining each binary classifier in order, so that, if a model predicts that the pattern is in the class which is isolated (not grouped with other classes), then this is the predicted class. This can be done in a forward manner or in a backward manner [15].

Finally, another possibility [67] is to derive a classifier for each class but separating the labels into groups
of three classes (instead of only two) for intermediate subtasks (labels lower than \( C_q \), label \( C_q \), and labels higher than \( C_q \)), or two classes for the extreme ones. The objective is to incorporate the order information in the subclassification tasks. Although the decomposition for intermediate classes is not binary but ternary, this approach has been included in this group because its motivation is similar to all the aforementioned.

### 3.2.2 Multiple-output single model approaches

Among non-parametric models, one appealing property of neural networks is that they can handle multiple responses in a seamless fashion \([68]\). Usually, as many output neurons as the number of target variables are included in the output layer and targets are presented to the network in the form of vectors \( \mathbf{t}_i \), \( i = 1, \ldots, N \). When applied to nominal classification, the most usual approach is to consider a 1-of-\( Q \) coding scheme \([53]\), i.e. \( \mathbf{t}_i = \{t_{i1}, \ldots, t_{iQ}\} \), \( t_{iq} = 1 \) if \( x_i \) corresponds to an example belonging to class \( C_q \), and \( t_{iq} = 0 \) (or \( t_{iq} = -1 \)), otherwise. In the ordinal regression framework, one can take the ordering information into account to design specific ordinal target coding schemes, which can improve the performance of the methods. Indeed, all the decompositions in Table 2 can be used to train neural networks, by taking each row as the code for the target class, \( \mathbf{t}_i \), and a single model will be obtained for all related subproblems (considering that each output neuron is solving each subproblem). This can be done by assigning a value \((+1, 0 \text{ or } -1)\) to each of the different symbols \((+ \text{ or } -)\) in Table 2. For sigmoidal output neurons, a 1 is assigned for positive symbols (+) and a 0 for negative ones (−). For hyperbolic functions, negative symbols are represented with a −1 and positive ones also with a 1. Those decompositions where a class is not involved should be treated as a “does not matter” condition where, whatever the output response, no error signal should be generated \([69]\).

A generalisation of ordinal perceptron learning \([70]\) in neural networks was proposed in \([21]\). The method is based on two main ideas: 1) the targets are coded using the OrderedPartitions approach; and 2) instead of using the softmax function \([53]\) for the output nodes, a standard sigmoid function is imposed, and the category assigned to a pattern is equal to the index previous to that of the first output node whose value is higher than a predefined threshold \( T \), or when no nodes are left. This method ignores inconsistencies (e.g. a sigmoid with value higher than \( T \) after the index selected).

Extreme learning machines (ELMs) are single-layer feedforward neural networks, where the hidden layer does not need to be tuned given that corresponding weights are randomly assigned. ELMs have demonstrated good scalability and generalisation performance with a faster learning speed when compared to other models such as SVMs \([72]\). They have been adapted to ordinal regression \([73]\), and one of the proposed ordinal ELMs also considers OrderedPartitions targets. Additionally, multiple models are also trained using the OneVsOne and the OrderedPartitions approaches. For the prediction phase, the loss-based decoding approach \([62]\) is utilised, i.e. the chosen label is that which minimises the exponential loss, \( k = \arg \min_{q=1, \ldots, Q} d(M_q, y(x)) \), where \( M_q \) is the code associated to class \( q \) (\( q \)-th row of the coding matrix), \( y(x) \) is the vector of predictions, and \( d(M_q, y(x)) \) is the exponential loss function, \( d(M_q, y(x)) = \sum_{i=1}^{Q} \exp(M_{qi} - y_i(x)) \). The values of the vector \( y(x) \) are assumed to be in the \([-1, +1]\) range, and those of \( M_{qi} \) in the set \([-1, 0, +1] \). The single ELM was found to obtain slightly better generalisation results and also to report the lowest computational time \([73]\). Other adaption of the ELM is found in \([74]\), where an evolutionary algorithm is applied to optimise the different weights of the model by using a fitness function to impose the ordering restriction in model selection. A different approach is taken in \([75]\), where the ordinal constraints are included into the weights connecting the hidden and output layers.

Costa \([69]\) followed a probabilistic framework to propose another neural network architecture able to exploit the ordinal nature of the data. The proposal is based on the joint prediction of constrained concurrent events, which can be turned into a classification task defined in a suitable space through a “partitive approach”. An appropriate entropic loss is derived for \( P(Y) \), i.e. the set of subsets of \( Y \), where \( Y \) is a set of \( Q \) elementary events. A probability for each possible subset should be estimated, leading to a total of \( 2^Q \) probabilities. However, depending on the classification problem, not all possibilities should be examined. For example, this is simplified for random variables taking values in finite ordered sets (i.e. ordinal regression), as well as in the case of independent boolean random variables (i.e. nominal classification). To adapt neural networks to the ordinal case structure, targets were reformulated following the OneVsFollowers approach and the prediction phase was accomplished by considering that, under its constrained entropic loss formulation, the output of the \( q \)-th output neuron estimates the probability that \( q \) and \( q - 1 \) events are both true. This methodology was further evaluated and compared in other works \([64, 76, 77]\).

Although all these neural network approaches consist of a single model, they are trained independently in the sense that the output of the neurons do not depend on the other outputs (only on common nonlinear transformations of the inputs). That is the reason why we have included them into the category of ordinal binary decompositions.

These neural network models can be grouped under the term multitask learning \([78]\) (MTL), which is a learning paradigm that considers the case of simultaneously tackling several related tasks. Any of the different proposals in this field could be applied to train a single model for the different ordinal decompositions analysed in this section. Indeed, one of the existing proposals, MTL via conic programming \([79]\), was validated in the
context of ordinal regression, showing promising results.

3.3 Threshold models

Often, in the ordinal regression paradigm, it is natural to assume that an unobserved continuous variable underlies the ordinal response variable. Such a variable is called a latent variable, and methods based on that assumption are known as threshold models, which are among the most popular approaches for modelling ordinal and ranking problems [49]. These methodologies estimate:

- A function \( f(x) \) that tries to predict the values of the latent variable, acting as a mapping function from feature space to the real one (similar to the ranking function to be learned by multipartite algorithms).
- A set of thresholds \( b = (b_1, b_2, \ldots, b_{Q-1}) \in \mathbb{R}^{Q-1} \) to represent intervals in the range of \( f(x) \), which must satisfy the constraints \( b_1 \leq b_2 \leq \ldots \leq b_{Q-1} \).

Threshold models can be seen as an extension of naïve regression models. The main difference between these two approaches is that the distances among the different classes are not defined a priori for threshold models, being estimated during the learning process. Although they are also related to (single-model) ordinal binary decomposition approaches, the main difference is that threshold models are based on one single projection vector with multiple thresholds, one for each class.

3.3.1 Cumulative link models

Arising from a statistical background, the Proportional Odds Model (POM) is one of the first models specifically designed for ordinal regression [80], dated back to 1980. It is a member of a wider family of models recognised as Cumulative Link Models (CLMs) [81]. In order to extend binary logistic regression to ordinal regression, CLMs predict probabilities of groups of contiguous categories, taking the ordinal scale into account. In this way, cumulative probabilities \( P(y \leq C_q|x) \) are estimated, which can be directly related to standard probabilities:

\[
P(y \leq C_q|x) = P(y = C_1|x) + \ldots + P(y = C_q|x),
\]

\[
P(y = C_q|x) = P(y \leq C_q|x) - P(y \leq C_{q-1}|x),
\]

with \( q = 2, \ldots, Q \), and considering by definition that \( P(y = C_1|x) = P(y \leq C_1|x) \) and \( P(y \leq C_0|x) = 1 \).

A decision rule \( f : X \rightarrow Y \) is not fitted directly. Instead, stochastic ordering of space \( X \) is satisfied by the following general model form [28]:

\[
g^{-1}(P(y \leq C_q|x)) = b_q - w^T x, q = 1, \ldots, Q,
\]

where \( g^{-1} : [0, 1] \rightarrow (-\infty, +\infty) \) is a monotonic function often referred to as the inverse link function and \( b_q \) is the threshold defined for class \( C_q \). This model is clearly inspired by the latent variable motivation, considering that \( f(x) = w^T x \) is a linear transformation. Consider the error of the model of the latent variable, \( f(x) = w^T x + \epsilon \), where \( \epsilon \) is the random component with zero expectation, \( E[\epsilon] = 0 \), distributed according to \( F_\epsilon \). If a distribution assumption \( F_\epsilon \) is made for \( \epsilon \), the cumulative model is obtained by choosing the inverse distribution \( F_\epsilon^{-1} \) as the inverse link function \( g^{-1} \). The most common choice for the distribution of \( \epsilon \) is the logistic function (which is indeed the one selected for the POM [82]), although probit, complementary log-log, negative log-log or cauchit functions could also be used [81]. If the ordinal response is a coarsely measured latent continuous variable \( f(x) \), label \( C_q \) in the training set is observed if and only if \( f(x) \in [b_{q-1}, b_q] \), where the function \( f \) and \( b = (b_0, b_1, \ldots, b_{Q-1}, b_Q) \) are to be determined from the data. It is assumed that \( b_0 = -\infty \) and \( b_Q = +\infty \), so the real line, defined by \( f(x), x \in X \), is divided into \( Q \) consecutive intervals. Each region separated by two consecutive biases corresponds to a category \( C_q \). The constraints \( b_1 \leq b_2 \leq \ldots \leq b_{Q-1} \) ensure that \( P(y \leq C_q|x) \) increases with \( q \) [83].

As will be seen, all the models in this section are inspired by the POM in the strategy assumed, obtaining a one-dimensional mapping function and dividing the real line into different ordered intervals. This mapping function can be used to obtain more information about the confidence of the predictions by relating it to its proximity to the biases. Additionally, the POM model provides us with a solid probabilistic interpretation. The distribution of \( \epsilon \) is assumed to be the standard logistic function for the POM:

\[
g^{-1}(P(y \leq C_q|x)) = \ln \left( \frac{P(y \leq C_q|x)}{P(y > C_q|x)} \right) = b_q - w^T x,
\]

where \( q = 1, \ldots, Q - 1 \), \( \text{odds}(y \leq C_q|x) = \exp(b_q - w^T x) \), so \( \text{odds}(y \leq C_q|x) = \frac{P(y \leq C_q|x)}{1 - P(y \leq C_q|x)} \). Therefore, the ratio of the odds for two patterns \( x_0 \) and \( x_1 \) are proportional:

\[
\frac{\text{odds}(y \leq C_q|x_1)}{\text{odds}(y \leq C_q|x_0)} = \exp(-w^T (x_1 - x_0)).
\]

More flexible non-proportional alternatives have been developed, one of them simply assuming different \( w \) for each class (which is known as the generalised ordered logit model [84]). Another alternative applies the proportional odds assumption only to a subset of variables (partial proportional odds [85]). Moreover, Tutz [86] presented a general framework for parametric models that extends generalised additive models to incorporate nonparametric parts.

Apart from assuming proportional odds, linear CLMs are rather inflexible since the decision functions are always linear hyperplanes, this generally affecting the performance of the model (as analysed in the experimental section of this work). A non-linear version of the POM model was proposed in [18, 83] by simply setting the projection \( f(x) \) to be the output of a neural network. The probabilistic interpretation of CLMs can be used to apply a maximum likelihood maximisation for setting the network parameters. Gradient descent techniques with proper constraints for the biases serve this purpose. This non-linear generalisation of the POM model based on neural networks was considered in [87], where an
evolutionary algorithm was applied to optimise all the parameters considered. Linear ordinal logistic regression was combined with nonlinear kernel machines using primal-dual relations from Nyström sampling [88]. However, to make the computation of the model feasible, a sub-sample from the data had to be selected, which limits the applicability to those cases where there is a reasonable way to do this [88].

An interesting alternative to CLMs is the so-called ordistic model presented in [89]. The work presents two threshold-based constructions which can be used to generalise loss functions for binary labels, such as the logistic and hinge loss, and another generalisation of the logistic loss based on a probabilistic model for ordered labels. Both constructions are based on including Q−1 thresholds partitioning the real line to Q segments, but they differ in how predictors outside the “correct” segment (or too close to its edges) are penalised. The immediate-threshold construction only penalises the violations of the two thresholds limiting this segment, while the all-threshold one considers all of them.

3.3.2 Support vector machines

Because of their good generalisation performance, SVM models are maybe the most widely applied ones to ordinal regression, their structure being easily adapted to that of threshold models. The proposal of Herbrich et al. [28], [90] is the first SVM based algorithm, where they consider a pairwise approach by deriving a new dataset made up of all possible difference vectors \( x'_i = x_i - x_j \) and \( y_{ij} = \text{sign}(\mathcal{O}(y_i) - \mathcal{O}(y_j)) \), with \( y_i, y_j \in \{C_1, \ldots, C_Q\} \). In contrast, all the SVM pointwise approaches share the common objective of seeking Q−1 parallel discriminant hyperplanes, all of them represented by a common vector \( w \) and the scalars biases \( b_1 \leq \ldots \leq b_{Q-1} \) to properly separate training data into ordered classes. In this sense, several methodologies for the computation of \( w \) and \( \{b_1, \ldots, b_{Q-1}\} \) can be considered. The work of Shawh and Levin [91] introduced two first methods: the maximisation of the margin between the closest neighbouring classes and the maximisation of the sum of margins between classes. Both approaches present two main problems [64]: the model is incompletely specified, because the thresholds are not uniquely defined, and they may not be properly ordered at the optimal solution, since the inequality \( b_1 \leq b_2 \leq \ldots \leq b_{Q-1} \) is not included in the formulation.

Consequently, Chu and Keerthi [29], [92] proposed two different reformulations for the same idea, solving the problem of unordered thresholds at the solution. On the one hand, they imposed explicit constraints on the optimisation problem, only considering adjacent labels for threshold determination (Support Vector Ordinal Regression with Explicit Constraints, SVOREX). On the other hand, patterns in all the categories were allowed to contribute errors for each hyperplane (SVOR with Implicit Constraints, SVORIM), which, as they prove [29], leads to automatically satisfied constraints in the optimal solution (see Lemma 1 of [29]). Let \( N_q \) be the number of patterns of class \( C_q \), and let \( x'^i_q \) be those patterns \( x \) which class label is \( C_q \). The SVORIM learning problem is defined as follows:

\[
\min \frac{1}{2} \|w\| + C \sum_{q=1}^{Q-1} \left( \sum_{j=1}^{N_q} \xi_{ji}^q + \sum_{j=q+1}^{Q} \sum_{i=1}^{N_q} \xi_{ji}^q \right),
\]

subject to the constraints:

\[
w \cdot x'^i_q - b_q \leq -1 + \xi_{ji}^q, \quad \xi_{ji}^q \geq 0, \quad j \in \{1, \ldots, q\},
\]

\[
w \cdot x'^i_q - b_q \geq +1 - \xi_{ji}^q, \quad \xi_{ji}^q \geq 0, \quad j \in \{q+1, \ldots, Q\},
\]

where \( i \in \{1, \ldots, N_q\} \), \( b \in \mathbb{R}^{Q-1} \), \( \xi_{ji}^q \) and \( \xi_{ji}^q \) are the slacks for the \( q \)-th parallel hyperplane (defined for the left and right part of the hyperplanes, respectively). The first group of constraints is focused on the left part of the \( j \)-th hyperplane (classes with \( q \leq j \)), while the second one is focused on the right part (classes with \( q > j \)). They empirically found that SVOREX performed better in terms of accuracy (with a more local behaviour), and SVORIM preceded in terms of absolute deviations in number of classes or \( \text{MAE} \) (with a more global behaviour), and this is justified theoretically based on the loss minimised for each method. The framework of reduction [41] also explains this from the point of view of the cost matrices selected. Our results seem to agree with these conclusions for discretised regression datasets, but the differences are not so clear for real ordinal regression ones. Generalisation properties for some ordinal regression algorithms, including SVOR, were further studied in [93].

In [94], the errors of an ordinal SVM classifier are studied separately depending on whether they correspond to upgrading errors (predicted label higher than the actual one) or downgrading ones (the predicted label being lower than the actual one). Authors address the two-objective problem of finding a classifier maximising simultaneously the two margins, and they show that the whole set of Pareto-optimal solutions can be obtained by solving a quadratic optimisation problem.

Some recent works focused on solving the bottleneck of these SVM proposals, which is usually the high computational complexity to handle larger datasets. Concerning this topic, two different proposals can be distinguished: block-quantised support vector ordinal regression [95] and ordinal-class core vector machines [96]. The former is based on performing kernel \( k \)-means and applying SVOR in the cluster representatives, on the idea of approximating the kernel matrix \( K \) by \( \hat{K} \) which will be composed of \( k^2 \) constant blocks, in such a way that the problem scales with the number of clusters, instead of the dataset size. The latter is an extension of core vector machines [97] in the ordinal regression setting. Finally, an incremental version of SVOR algorithms is proposed in [98].
### 3.3.3 Discriminant learning

Discriminant learning has also been reformulated to tackle ordinal regression \[^{[31]}\]. Discriminant analysis is usually not considered as a classification technique by itself, but rather as a supervised dimensionality reduction. Nonetheless, it is widely used for that purpose, since, as a projection method, the definition of thresholds can be used to discriminate the classes. In general, to allow the computation of the optimal projection for the data, this algorithm analyses two main objectives: the maximisation of the between-class distance, and the minimisation of the within-class distance, by using variance-covariance matrices and the Rayleigh coefficient. In order to reformulate the algorithm for ordinal regression, an ordering constraint over contiguous classes is imposed on the averages of projected patterns of each class, which leads the algorithm to order projected patterns according to their label. This will preserve the ordinal information and avoid some serious ordinal misclassification errors.

The original optimisation problem is transformed and extended with a penalty term \((C)\):

\[
\min J(w, \rho) = w^T S_w w - C \rho,
\]

subject to \(w^T (m_{q+1} - m_q) \geq \rho\), where \(m_q = \frac{1}{N} \sum_{i=1}^{N} x_i\), \(S_w\) is the within-class scatter matrix and \(\rho\) represents the minimum difference of the projected means between consecutive classes (if \(\rho > 0\), the projected means are correctly ranked). This methodology is known as Kernel Discriminant Learning for Ordinal Regression (KDLOR) \[^{[31]}\] and it has been used in some later works \[^{[9]}, [99]\]. In \[^{[100]}\], the KDLOR model is extended by trying to learn multiple orthogonal projections, which are then combined into a final decision function.

The method was extended in \[^{[101]}, [102]\] based on the idea of preserving the intrinsic geometry of the data in the embedded non-linear structure, i.e. in the induced high-dimensional feature space, via kernel mapping. This consideration is the basis of manifold learning \[^{[53]}\], and the algorithms mentioned construct a neighbourhood graph (which takes the ordinal nature of the dataset into account) which is afterwards used to derive the Laplacian matrix and obtain a projection which considers the underlying manifold of the data. A related method is proposed in \[^{[103]}\], where several different projections are iteratively derived.

### 3.3.4 Perceptron learning

PRank \[^{[104]}\] is a perceptron online learning algorithm with the structure of threshold models. It was then extended by approximating the Bayes point, what provides good performance for generalisation \[^{[55]}\].

### 3.3.5 Augmented binary classification

Although the approaches in Subsection 3.2 are simple to implement, their generalisation performance cannot be analysed easily. The two algorithms included in this subsection work differently, and, as later analysed, the models derived are equivalent to threshold models.

A reduction framework can be found in the works of Lin and Li \[^{[41]}, [105]\], where ordinal regression is reduced to binary classification by applying three steps:

1) A coding matrix \(M\) is used to represent the class being examined. Given a coding matrix \(M\) of \((Q-1)\) rows, input patterns \((x_i, y_i)\) are transformed into extended binary patterns by replicating them, \((x_i^{(q)}, y_i^{(q)})\), with:

\[
x_i^{(q)} = (x_i, m_q), \quad y_i^{(q)} = 2[q < O(y_i)] - 1,
\]

where \(q = 1, \ldots, Q-1\), \(m_q\) is the \(q\)-th row of \(M\) and \(O[\cdot]\) is a Boolean test which is 1 if the inner condition is true, and 0 otherwise. \(Q-1\) replicates of each pattern are generated with weights:

\[
w_{i,q} = (Q-1) \cdot |C_{O(y_i),q-1} - C_{O(y_i),q+1}|,
\]

where \(i = 1, \ldots, N\), \(C\) is a V-shaped cost matrix (i.e. \(C_{O(y_i),q-1} \geq C_{O(y_i),q}\) if \(q \leq O(y_i)\), and \(C_{O(y_i),q} \leq C_{O(y_i),q+1}\) if \(q \geq O(y_i)\)). The cost matrix must be defined a priori. An example of this transformation is given in Table 3. As can be seen, the final extended pattern represents the question “Is the rank of \(x\) greater than \(q\)?” \[^{[41]}\]. The weights measure the importance of the pattern for the binary classifier, and they are also used for the theoretical analysis.

2) A single binary classifier with confidence outputs, \(f(x, m_k)\), is trained for the new weighted extended patterns, aiming at a low weighted 0/1 loss.

3) A classification rule like the following is used to construct a final prediction for new patterns:

\[
r(x) = 1 + \sum_{q=1}^{Q-1} [f(x, m_q) > 0].
\]

All the binary classification problems are solved jointly by computing a single binary classifier. The most striking characteristic of this algorithm is that it unifies many existing ordinal regression algorithms \[^{[41]}\], such as the perceptron ones \[^{[104]}\], kernel ranking \[^{[56]}\], AdaBoost.OR \[^{[106]}\], ORBoost-LR and ORBoost-All thresholded ensemble models \[^{[107]}\], CLM \[^{[81]}\] or several ordinal SVM.
proposals (oSVM [64], SVORIM and SVOREX [29]). Moreover, it is important to highlight the theoretical guarantees provided by the framework, including the derived cost and regret bounds and the proof of equivalence between ordinal regression and binary classification. An extension of this reduction framework was proposed in [108], where ordinal regression is proved to be equivalent to a regular multiclass classification whose distribution is changed. This extension is free of the following restrictions: target functions should be rank-monotonic; and rows of loss matrix are convex.

The data replication method of Cardoso et al. [64] (whose previous linear version appeared in [10]) is a very similar framework, except that it essentially considers the absolute cost, consequently being less flexible. However, for ordinal regression, increasing the error with the absolute difference between the predicted and estimated labels is a natural choice in the absence of any other information [18]. An advantage of the framework of data replication is that it includes a parameter $s$ which limits the number of adjacent classes considered, in such a way that the replicate $q$ is constructed by using the $q - s$ classes to its ‘left’ and the $q + s$ classes to its ‘right’ [64]. This parameter $s \in \{1, ..., Q - 1\}$ plays the role of controlling the increase of data points.

It is worth mentioning that augmented binary classification models and threshold models are closely related, and that is the reason why they have been included in this section. The extended patterns only differ in the new variables introduced by the coding matrix $M$. The original version of the dataset is replicated in different subspaces, with different values for the new variables. By obtaining the intersection of the binary hyperplane in the extended dataset with each of the subspace replicas we derive parallel boundaries in the original dataset [64], with a single projection vector and multiple thresholds.

In fact, SVORIM and reduction to SVM is known to be not so different in formulation [103]. The model of Mathieson [18] (threshold model) is equivalent to the one proposed in [64] (oNN, an augmented binary classification model) if the activation function of the output node is set to the $\text{logsig}$ function and the model is trained to predict the posterior probabilities when fed with the original input variables and the variables generated by the data replication method. The predicted thresholds would be the weights of the connection of the added $Q - 2$ components. Finally, augmented binary classification and ordinal binary decomposition are not disjoint categories. The former class of models does not restrict consistency of binary classifiers, making use of a “voting” of the binary classifiers (see Eq. 1 of [105]). Moreover, all the ordinal decompositions in Table 2 can be viewed as a special case of “cost-sensitive ordinal classification” via augmented binary classification [41].

### 3.3.6 Ensembles

From a different perspective, the confidence of a binary classifier can be regarded as an ordering preference. RankBoost [109] is a boosting algorithm that constructs an ensemble of those confidence functions to form a better ordering preference. Some efforts were made to apply a similar idea for ordinal regression problems, deriving into Ordinal Regression Boosting (ORBoost) [107]. The corresponding thresholded-ensemble models inherit the good properties of ensembles, including more stable predictions and sufficient power for approximating complicated target functions [110]. The model is composed of confidence functions, and their weighted linear combination is used as the projection $f(x)$. A set of thresholds for this projection is also included in the model and iteratively updated with the rest of parameters. Following a similar approach to [85], large margin bounds of the classification error and the absolute error are derived, from which two algorithms are presented: ORBoost with all margins and ORBoost with left-right margins [107]. Two alternative thresholded-ensemble algorithms are presented in [111], both generating an ensemble of ordinal decision rules based on forward stagewise additive modelling.

With a different perspective, the well-known AdaBoost algorithm was recently extended to improve any base ordinal regression algorithm [106]. The extension, AdaBoost.OR, proved to inherit the good properties of AdaBoost, improving both the training and test performances of existing ordinal classifiers. Another ordinal regression version of AdaBoost is proposed in [112], while in this case the adaption is based on considering a cost matrix both in pattern weighting and error updating.

The framework of negative correlation learning (where the ensemble members are learnt in such a way that the correlation between their responses is minimised) was used in the context of ordinal regression [17, 13] by calculating the correlation between the latent variable estimations or, alternatively, between the probabilities obtained by the ensemble members.

### 3.3.7 Gaussian processes

All the previous threshold models can be considered discriminative models in the sense that they estimate directly the posterior $P(y|x)$, or learn a function to map the input $x$ to class labels. On the contrary, generative models learn a model of the joint probability $P(x,y)$ of input patterns $x$ and label $y$, and make the prediction by a Bayesian framework to estimate $P(y|x)$.

Gaussian Processes for Ordinal Regression (GPOR) [30] models the latent variable $f(x)$ using Gaussian Processes, to estimate then all the parameters by means of a Bayesian framework. The values of the latent function $\{f(x_i)\}$ are assumed to be the given by random variables indexed by their input vectors in a zero-mean Gaussian process. Mercer kernel functions approximate the covariance between the functions of two input vectors. Given
the latent function $f$, the joint probability of observing the ordinal variables is $P(D|f) = \prod_{i=1}^{N} P(y_i|f(x_i))$, and the Bayes theorem is applied to write the posterior probability $P(f|D) = \frac{P(D|f)P(f)}{\sum_{f} P(D|f)P(f)}$. A Gaussian noise with zero mean and unknown variance $\sigma^2$ is assumed for the latent functions. The normalisation factor is known as the evidence $D$, more exactly $P(D|\theta)$, is known as the evidence for the vector of hyperparameters $\theta$ and is estimated in the paper by two different approaches: a Maximum a Posteriori approach with Laplace approximation and an Expectation Propagation with variational methods. A more general GPOR was then proposed to tackle multi-classification problems but with a free structure of preferences over the labels [114]. A probabilistic sparse kernel model was proposed for ordinal regression in [115], where a Bayesian treatment was also employed to train the model. A prior over the weights governed by a set of hyperparameters was inspired, by the well-known relevance vector machine. Srijith et al. have proposed a probabilistic least squares version of GPOR [116], two different sparse versions [117] and a semi-supervised version [118].

### 3.4 Other approaches and problem formulations

This subsection includes some methods that are difficult to consider in the previous groups. For example, an alternative methodology is proposed by da Costa et al. [76], [77] for training ordinal regression models. The main assumption of their proposal is that the random variable class associated with a given pattern should follow a unimodal distribution. For this purpose, they provide two possible implementations: a parametric one, where a specific discrete distribution is assumed and the associated free parameters are estimated by a neural network; and a non-parametric one, where no distribution is assumed but the error function is modified to avoid errors from distant classes. The same idea was then applied to SVMs in [119] by solving an ordinal problem through a single optimisation process (the all-at-once strategy).

In [120], both decision trees and nearest neighbour (NN) classifiers are applied to ordinal regression problems by introducing the notion of consistency: a small change in the input data should not lead to a 'big jump' in the output decision, i.e. adjacent decision regions should have equal or consecutive labels. This rationale was used as a post-processing mechanism of a standard decision tree and as a pre- or post-processing step for the NN method. An improvement was presented in [121] to reduce the over-regularised decision region artifact by using ensemble learning techniques.

Two ordinal learning vector quantisation schemes, with metric learning, specifically designed for classifying data items into ordered classes, are introduced in [122], [123]. The methods use the order information during training, both in the selection of the prototypes and for determining the way they are updated.

Different prediction methods as a function of the error measure to be minimised are presented in [124]. The paper discusses the fact that the Bayes optimal decision for a classifier which return probability estimates is different depending on the loss function considered for the errors. In this way, for the maximisation of the accuracy one should consider the mode (or maximum probability), but the median of the probability distribution is the optimal decision when minimising the MAE in ordinal regression problems.

### 4 Experimental study

#### 4.1 Experimental design

In this subsection, the experiments are clearly specified, including the datasets and algorithms considered, the parameters to optimise, the performance measures and the statistical tests used for assessing the differences.

##### 4.1.1 Datasets selected

The most widely used dataset repository is the one provided by Chu et al. [30], including different regression benchmark datasets. These datasets are not real ordinal classification problems but regression ones, which are turned into ordinal classification by discretising the target into $Q$ different bins with equal frequency. It is clear that these datasets do not exhibit some characteristics of typical complex classification tasks, such as class imbalance, given that all classes are assigned the same number of patterns. However, we find interesting to check how the algorithms perform in this more controlled environment and to compare the conclusions obtained.

Table 4 shows the characteristics of the 41 datasets, including the number of patterns, attributes and classes, and also the number of patterns per class. The real ordinal classification datasets were extracted from benchmark repositories [UCI (125) and mldata.org (126)], and the regression ones were obtained from the website of W. Chu [119]. For the discretised datasets, we considered $Q = 5$ and $Q = 10$ bins to evaluate the response of the classifiers to the increase in the complexity of the problem. The synthetic toy dataset was generated as proposed in [27] with 300 patterns. All nominal attributes were transformed into as many binary attributes as the number of categories, and all the datasets were properly standardised.

##### 4.1.2 Algorithms selected

We have selected some representatives of the different families included in the proposed taxonomy (see Table 5). It is important to note that naive approaches and ordinal binary decompositions can be applied using almost any base binary classifier or regressor. In our experiments, we have selected in those cases SVMs, given

1. We would like to note that many of these datasets have been previously considered in machine learning literature, but ignoring the ordering information.
2. [http://www.gatsby.ucl.ac.uk/~chuwei/ordinalregression.html](http://www.gatsby.ucl.ac.uk/~chuwei/ordinalregression.html)
that they are suggested by many of the authors of the different works analysed. Starting with naive approaches, the following methods were considered: 1) C-Support Vector Classifier (C-SVC) with OneVsOne and OneVsAll decompositions (SVC1V1 and SVC1VA), because they are the two main approaches when applying SVM to multiclass problems [59]. Although these methods consider binary decompositions, they have been included in the nominal classification group, given that they do not take the class order into account. 2) Support Vector Regression (SVR) applied to a modified dataset where the target variable $Y = \{C_1, C_2, \ldots, C_Q\}$ is mapped to the real values $\{0.1/(Q-1), 2/(Q-1), \ldots, 1\}$. The concrete regression model considered is the $c$-SVR [52]. 3) Cost-Sensitive SVC (CSSVC), which is a $C$-SVC [59] with the OneVsAll decomposition, where absolute costs are included as different weights [127] for the negative class of each decomposition.

Regarding the ordinal binary decompositions, the methods considered are the following: 1) The Ordered-Partitions decomposition was applied to the C-SVC classification algorithm (SVMOP), but including different weights, as proposed by Waegeman et al. [65]. However, given the problem of possible ambiguities recognised by the authors, probability estimates are obtained following the method presented in [128]. Then, the fusion of probabilities of Eq. (1) is performed [63]. 2) The neural network model proposed in [71] (NNOP). This model considers the OrderedPartitions coding scheme for the labels and a rule for decisions based on the first node whose output is higher than a predefined threshold ($T = 0.5$, in our experiments). We consider then the mean square error function over the outputs and the iRProp+ algorithm [129] to optimise the parameters. 3) Finally, the single model ordinal ELM presented in [73] (ELMOP).

The threshold models considered are the following: 1) The POM [52], with the logit link function (the most popular one). 2) A neural network approach based on the POM (NNPOM), similar to the one proposed by Matheson [18]. The cross entropy function is optimised by the iRProp+ algorithm [129]. Threshold constraints are satisfied by substituting the set of parameters $\{b_1, b_2, \ldots, b_Q\}$ by $\{\alpha_1, \alpha_1 + \alpha_2, \ldots, \alpha_1 + \alpha_2 + \cdots + \alpha_Q\}$, which allows unconstrained optimisation of $\{\alpha_1, \ldots, \alpha_Q\}$. 3) Ordinal support vector vector formulations of Chu and Keerthi [29], both explicitly and implicitly constrained alternatives (SVOEX and SVORIM). We have also included a linear version of the SVORIM method (considering the linear kernel instead of the Gaussian one) to check how the kernel trick affects the final performance (SVORLin). 4) KDLOR algorithm presented in [31]. 5) The GPOR method [30] including automatic relevance determination, as proposed by the authors. 6) The reduction from ordinal regression to binary SVM classifiers was also considered (REDSV). The configuration used was the identity coding matrix, the absolute cost matrix and the standard binary soft-margin SVM, as proposed in [41]. 7) Finally, the ORBoost method with all margins [107] (ORBALL). As proposed by the authors, the total number of ensemble members is set to $T = 2000$, and normalised

### Table 4

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Pat.</th>
<th>#Attr.</th>
<th>#Classes</th>
<th>Class distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>pyrim1 (P1)</td>
<td>74</td>
<td>6</td>
<td>3</td>
<td>(15, 5, 4)</td>
</tr>
<tr>
<td>machine5 (M5)</td>
<td>209</td>
<td>7</td>
<td>5</td>
<td>(42, 42, 16)</td>
</tr>
<tr>
<td>housing5 (H5)</td>
<td>506</td>
<td>14</td>
<td>5</td>
<td>(101, 94, 7)</td>
</tr>
<tr>
<td>stock5 (SS)</td>
<td>700</td>
<td>9</td>
<td>5</td>
<td>(140, 60, 2)</td>
</tr>
<tr>
<td>abalone5 (AS)</td>
<td>4177</td>
<td>11</td>
<td>5</td>
<td>(836, 77, 1)</td>
</tr>
<tr>
<td>bank5 (BS)</td>
<td>8192</td>
<td>8</td>
<td>5</td>
<td>(1639, 10, 2)</td>
</tr>
<tr>
<td>bank5’ (BS5)</td>
<td>8192</td>
<td>32</td>
<td>5</td>
<td>(1639, 10, 2)</td>
</tr>
<tr>
<td>computer5 (CS)</td>
<td>8192</td>
<td>12</td>
<td>5</td>
<td>(1639, 10, 2)</td>
</tr>
<tr>
<td>computer5’ (CS5)</td>
<td>8192</td>
<td>12</td>
<td>5</td>
<td>(1639, 10, 2)</td>
</tr>
<tr>
<td>cal.housing (CHS)</td>
<td>20640</td>
<td>8</td>
<td>5</td>
<td>4128 per class</td>
</tr>
<tr>
<td>census5 (CE5)</td>
<td>22784</td>
<td>8</td>
<td>5</td>
<td>4557 per class</td>
</tr>
<tr>
<td>census5’ (CEES)</td>
<td>22784</td>
<td>16</td>
<td>5</td>
<td>4557 per class</td>
</tr>
<tr>
<td>pyrim5 (P5)</td>
<td>74</td>
<td>22</td>
<td>10</td>
<td>8 per class</td>
</tr>
<tr>
<td>machine10 (M10)</td>
<td>209</td>
<td>7</td>
<td>10</td>
<td>21 per class</td>
</tr>
<tr>
<td>housing10 (H10)</td>
<td>506</td>
<td>14</td>
<td>10</td>
<td>51 per class</td>
</tr>
<tr>
<td>stock10 (S10)</td>
<td>700</td>
<td>9</td>
<td>10</td>
<td>70 per class</td>
</tr>
<tr>
<td>abalone10 (A10)</td>
<td>4177</td>
<td>11</td>
<td>10</td>
<td>415 per class</td>
</tr>
<tr>
<td>bank10 (B10)</td>
<td>8192</td>
<td>8</td>
<td>10</td>
<td>820 per class</td>
</tr>
<tr>
<td>bank10’ (BB10)</td>
<td>8192</td>
<td>32</td>
<td>10</td>
<td>820 per class</td>
</tr>
<tr>
<td>computer10 (C10)</td>
<td>8192</td>
<td>12</td>
<td>10</td>
<td>820 per class</td>
</tr>
<tr>
<td>computer10’ (C10)</td>
<td>8192</td>
<td>21</td>
<td>10</td>
<td>820 per class</td>
</tr>
<tr>
<td>cal.housing10 (CH10)</td>
<td>20640</td>
<td>8</td>
<td>10</td>
<td>2064 per class</td>
</tr>
<tr>
<td>census10 (CE10)</td>
<td>22784</td>
<td>8</td>
<td>10</td>
<td>2279 per class</td>
</tr>
<tr>
<td>census10’ (CEE10)</td>
<td>22784</td>
<td>16</td>
<td>10</td>
<td>2279 per class</td>
</tr>
</tbody>
</table>

### Table 5

<table>
<thead>
<tr>
<th>Abbr.</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive approaches</td>
<td></td>
</tr>
<tr>
<td>SVC1V1</td>
<td>Support Vector Classifier with OneVsOne [59]</td>
</tr>
<tr>
<td>SVC1VA</td>
<td>Support Vector Classifier with OneVsAll [59]</td>
</tr>
<tr>
<td>SVR</td>
<td>Support Vector Machines for regression [57]</td>
</tr>
<tr>
<td>CSSVC</td>
<td>Cost-Sensitive Support Vector Classifier [59]</td>
</tr>
<tr>
<td>Ordinal Binary decompositions</td>
<td></td>
</tr>
<tr>
<td>SMOP</td>
<td>Support Vector Machines with OrderedPartitions [59]</td>
</tr>
<tr>
<td>NNOP</td>
<td>Neural Network with OrderedPartitions [71]</td>
</tr>
<tr>
<td>ELMOP</td>
<td>Extreme Learning Machine with OrderedPartitions [85]</td>
</tr>
<tr>
<td>Threshold models</td>
<td></td>
</tr>
<tr>
<td>POM</td>
<td>Proportional Odds Model [80]</td>
</tr>
<tr>
<td>NNPOM</td>
<td>Neural Network based on Proportional Odd Model [18]</td>
</tr>
<tr>
<td>SVOEX</td>
<td>Support Vector Ordinal Regression with Explicit Constraints [29]</td>
</tr>
<tr>
<td>SVORIM</td>
<td>Support Vector Ordinal Regression with Implicit Constraints [29]</td>
</tr>
<tr>
<td>SVORLin</td>
<td>SVORIM using a linear kernel [29]</td>
</tr>
<tr>
<td>KDLOR</td>
<td>Kernel Discriminant Learning for Ordinal Regression [51]</td>
</tr>
<tr>
<td>GPOR</td>
<td>Gaussian Processes for Ordinal Regression [30]</td>
</tr>
<tr>
<td>REDSV</td>
<td>Reduction applied to Support Vector Machines [41]</td>
</tr>
<tr>
<td>ORBALL</td>
<td>Ordinal Regression Boosting with All margins [107]</td>
</tr>
</tbody>
</table>
sigmoid functions are used as the base classifier, where the smoothness parameter is $\gamma = 4$ \cite{107}.

**4.1.3 Performance evaluation and model selection**

Different measures can be considered for evaluating ordinal regression models \cite{119, 130, 131}. However, the most common ones are the Mean Zero-one Error (MZE) and the Mean Absolute Error (MAE). MZE is the error rate of the classifier:

$$MZE = \frac{1}{N} \sum_{i=1}^{N} \left[ I[y_i^* \neq y_i] \right] = 1 - Acc,$$

where $y_i$ is the true label, $y_i^*$ is the predicted label and $Acc$ is the accuracy of classifier. MZE values range from 0 to 1. It is related to global performance, but without considering the order. The MAE is the average deviation in absolute value of the predicted rank (O($y_i^*$)) from the true one (O($y_i$)) \cite{131}:

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |O(y_i) - O(y_i^*)|.$$  

MAE values range from 0 to $Q-1$ (maximum deviation in number of categories). In this way, MZE considers a zero-one loss for misclassification, while MAE uses an absolute cost. We consider these costs for evaluating the datasets because they are most common (for example, see \cite{29, 31}, just to cite some of them).

Multiple random splits of the datasets were considered. For discretised regression datasets, 20 random splits were done and the number of training and test patterns were those suggested in \cite{30}. For real ordinal regression problems, 30 random stratified splits with 75$\%$ and 25$\%$ of the patterns in the training and test sets were considered, respectively (as suggested in \cite{132}). All the partitions were the same for all the methods, and one model was trained and evaluated for each split. Then, MZE or MAE values were obtained, and the computational times were also gathered.

All SVM classifiers or regressors were run using the implementations available in the libsvm library (version 3.0) \cite{127}. The mnrfit function of Matlab was used for training the POM model. The authors of GPOR, SVOREX, SVORIM, RED-SVM and ORBoost provide publicly available software implementations of their methods. All the experiments were run using a common Matlab framework, with an Intel(R) Xeon(R) CPU E5405 at 2.00GHz with 8GB of RAM. This framework is available, together with all the datasets and partitions, the individual results and the detailed results of the statistical tests, on the website associated with this paper:

3. GPOR (http://www.gatsby.ucl.ac.uk/~chuwei/ordinalregression.html), SVOREX and SVORIM (http://www.gatsby.ucl.ac.uk/~chuwei/svor.html), ORBoost (http://www.work.caitech.edu/~htlin/program/orensemble/) and RED-SVM (http://home.caitech.edu/~htlin/program/libsvm/)


It is very important to consider a proper model selection process to assure a fair comparison. In this sense, all model hyperparameters were selected by using a nested five fold cross-validation over the training set. Once the lowest cross-validation error alternative was obtained, it was applied to the complete training set and test results were extracted. The criteria for selecting the best configuration were both MAE and MZE performances, depending on the measure we were interested in. The parameter configurations explored are now specified. The Gaussian kernel function was considered for all the kernel methods (SVC1V1, SVC1VA, SVR, CSSVC, SVMOP, SVOREX, SVORIM, RED-SVM and KDLOR). The following values were considered for the width of the kernel, $\sigma \in \{10^{-3}, 10^{-2}, \ldots, 10^3\}$. The cost parameter $C$ of all SVM methods (including SVORLin) was selected within the values $C \in \{10^{-3}, 10^{-2}, \ldots, 10^3\}$ and for the KDLOR within the values $C \in \{10^{-4}, 10^0, 10^1\}$, since, in this case, this parameter presents a different interpretation and, therefore, there is no need to use a larger spectrum of values. An additional parameter $u$ was also needed by KDLOR, which is intended to avoid singularities in the covariance matrices. The values considered were $u \in \{10^{-6}, 10^{-5}, \ldots, 10^{-2}\}$. The range of $\epsilon$ for $\epsilon$-SVR was $\epsilon \in \{10^0, 10^1, \ldots, 10^3\}$. For the neural network algorithms (NNOP and NNPMOP), the number of hidden neurons, $H$, was selected by considering the following values, $H \in \{5, 10, 20, 30, 40\}$. The sigmoidal activation function was considered for hidden neurons. For the iRProp+ algorithm, the number of iterations, $iter$, was also decided by cross-validation, by considering the values $iter \in \{50, 100, 150, \ldots, 500\}$. The other parameters of iRProp+ were set as in \cite{129}. For ELMOP, higher numbers of hidden neurons are considered, $H \in \{5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$, given that it relies on sufficiently informative random projections \cite{72}. With regards to the GPOR algorithm, the hyperparameters are determined by part of the optimisation process. The ORBoost process did not need any hyperparameter.

Each pair of algorithms is compared by means of the Wilcoxon test \cite{133}. A level of significance of $\alpha = 0.1$ was considered, and the corresponding correction for the number of comparisons was also included. As 16 algorithms are compared, the total number of comparisons for each dataset is 120, so the corrected level of significance was $\alpha^* = 0.1/120 = 0.00083$.

**4.2 Discretised regression datasets**

Tables 6 and 7 show the results obtained for all algorithms throughout the discretised regression datasets (when considering $Q = 5$ and $Q = 10$ bins), and also the ordinal regression ones (analysed in the following subsection). The results include the average and the standard deviation of MZE and MAE, respectively. Additionally, an analysis of the ranking of each method for each dataset was done, where this ranking is 1 for the best method and 16 for the worst one. Then, the average
values of these rankings were obtained for discretised regression problems of 5 classes ($R(D_5)$), of 10 classes ($R(D_{10})$), and for real ordinal regression problems ($R(OR)$) and for real ordinal regression ones ($R(CR)$) as a reference of the general performance. For all methods including a model selection process, Table 6 shows the results when using $MZE$ as the selection criterion, while the results in Table 7 are obtained using $MZE$ for this selection. In addition, the average rankings of the total computational time (the sum of cross-validation, training and test times) are presented in Table 8. For all these Tables, the best method for each dataset (or set of datasets) is in bold face and the second one in italics.

As previously stated, the Wilcoxon test was applied to check the existence of significant differences. Using this test, each pair of methods was compared for each dataset and the number of statistically significant wins or losses was recorded, together with the number of draws (or absence of statistically significant differences). These results are included in Table 9 for the discretised regression datasets ($Q = 5$ and $Q = 10$). The number of wins ($w$), draws ($d$) and losses ($l$) for $15 \times 24 = 360$ comparisons are included (24 datasets and 15 methods to compare each method against). The methods are ordered by the number of statistically significant wins.

Using analysing Tables 6, 7, 8 and specially Table 9, the best performing ordinal regression methods from the families in the taxonomy can be obtained. From the naive approaches, SVC1V1 obtains better accuracy or $MZE$ while SVR is better on $MZE$. However, both improved performances imply worse results for $MZE$ and $MZE$, respectively. The computational time of SVR, higher, given that an additional parameter has to be cross-validated. In general, SVC1VA and CVSCV show worse $MZE$ and $MZE$ than SVC1VC. When considering ordinal binary decompositions, SVMOP is
Table 7

Test MAE results for each method, including the average over all the splits and the standard deviation.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>SVLV1</th>
<th>SVLV2</th>
<th>SVR</th>
<th>CNVLV</th>
<th>SVNV</th>
<th>NNPOM</th>
<th>ELMOP’T</th>
<th>SVMOP</th>
<th>SVRIn</th>
<th>LKDR</th>
<th>GCLR</th>
<th>RDSVM</th>
<th>ORDHALL</th>
</tr>
</thead>
<tbody>
<tr>
<td>P5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>M5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>H5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>SS</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>A5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>B5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>CHS</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>CS</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>CC5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>CH5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>CE5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>CEE5</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
<td>0.19</td>
</tr>
</tbody>
</table>

The best result for each dataset is in bold face and the second one in italics.
GPOR is much lower in both $MZE$ and $MAE$. SVM based threshold models are the best performing ones for both measures, SVOREX achieving the best results in $MZE$ and very close to the best performing method, REDSVM, in $MAE$. In this case, the gap of performance between SVORIM and SVORLin is higher. Discarding POM and ELMOP, the lowest computationally time is associated to SVORLin. For ordinal regression datasets, the results of ORBALL are better with respect to the rest of methods than when consider the discretised datasets. With respect to REDSVM, its computational cost is high when compared to SVOR, ORBoost and SVC methods.

### 4.4 Discussion

Several ordinal regression methods (see Tables 8 and 10) can be emphasised according to their error (GPOR, SVOREX, SVORIM, REDSVM and SVMOP) or their computational time (POM, SVORLin or ELMOP). However, there are many factors that can influence the choice of the method, and all of them should be considered.

First of all, POM is a linear model and, as such, it is very fast to train (with no associated hyperparameters), but its performance is significantly low (except for $MZE$ in discretised regression datasets). This fact is important, given that, excluding the machine learning area, the POM and its variants are the most widely used ordinal regression methods [25, 81, 84, 88].

When dealing with large datasets, we conclude that POM is a good option, given the low computational cost needed. The results achieved for $MZE$ and $MAE$ are worse than those of other alternatives, but they are good enough when computational time is a priority. SVORLin can also be a good option, although a narrower range of cross-validation for $C$ should be selected. Neural networks (NNPOM and NNOP) are generally beaten by their SVM counterparts, both in $MZE$ and $MAE$. Moreover, the training time for these methods and GPOR is generally the highest.

Our study shows that the naïve approaches can obtain competitive performance and be difficult to beat for some datasets. SVC1V1 achieves very good $MZE$ results for real ordinal regression datasets. However, SVM threshold models improves $MAE$ and $MZE$ results, as well as being simpler models. Indeed, all threshold models allow the visualisation of predicted projections together with the thresholds. This can be used for various purposes, from ranking patterns to trying to discover uncertain predictions (projections very close to class thresholds). This kind of analysis is generally more difficult with nominal models, such as SVC1V1. In general, the SVC1VA alternative has been shown to achieve worse results than SVC1V1 for the three measures evaluated (as previously shown in other studies [59]). CSSVC results are a bit better than those of SVC1VA, but still far from SVC1V1.

Binary decomposition approaches are shown to be good alternatives, especially SVMOP. However, as discussed in Subsection 3.2 their theoretical analysis is more difficult, and it is necessary to decide how to
combine different binary predictions. Of all the threshold models analysed, SVOREX and SVORIM are the best. The computational time required by SVOREX is slightly lower, and it always achieves better results than SVORIM, except for MAE in discretised regression datasets. ORBALL shows a worse performance than SVOR methods. REDSVM is very competitive, but with a higher computational cost.

When comparing discretised regression datasets and real ordinal regression ones, some performance differences can be highlighted. For example, GPOR performance is seriously affected when dealing with real ordinal classification datasets. In general, SVM and ORBALL methods are more robust in the derived problems that can appear with these datasets. This is an important point, because many of the ordinal regression works in the literature make use of discretised regression sets, hiding some possible difficulties of the methods when dealing with problems such as imbalanced distributions.

When real ordinal regression datasets are considered, POM and GPOR performances decrease (both in MZE and MAE) drastically. Both models have one feature in common. They assume that their perturbation terms follow some possible difficulties of the methods when dealing with problems such as imbalanced distributions.

When real ordinal regression datasets are considered, POM and GPOR performances decrease (both in MZE and MAE) drastically. Both models have one feature in common. They assume that their perturbation terms follow some possible difficulties of the methods when dealing with problems such as imbalanced distributions.

5 Conclusions

This paper offers an exhaustive survey of the ordinal regression methods proposed in the literature. The problem setting has been clearly established and differentiated from other ranking topics. After this, a taxonomy of ordinal regression methods is proposed, dividing them into three main groups: naïve approaches, binary decompositions and threshold models. Furthermore, the most important methods of each family (a total of 16 methods) are empirically evaluated in two kinds of datasets, 24 discretised regression datasets and 17 real ordinal regression ones.

The taxonomy proposed can help the researcher or the practitioner choose the best method for a concrete problem, considering also the empirical results herein provided. It can also assist researchers in developing and proposing new methods, providing a way to classify them and to select the most similar ones. The results presented in this paper confirm that there is no single method which performs the best in all possible datasets and problem requirements. However, these results can be used to discard some of the methods, especially those clearly presenting worse performance or too high computational time. We would like to stress certain methods: 1) SVC1V1 as representative of the naïve approaches, achieving an especially good MZE because of the recursive partitioning of all pairs of classes; 2) SVMOP achieves the best results from ordinal binary decomposition methods; 3) ELMOP or POM are a good option if the computational cost is a priority; and 4) SVOREX and SVORIM can be considered the best threshold models, showing competitive accuracy, MAE and time values. Finally, there is a website [http://www.uco.es/grupos/aryna/orreview] collecting the implementations of the methods in this survey, the detailed results, the datasets and the corresponding statistical analysis.

References


Francisco Fernández-Navarro (M’13) was born in Córdoba, Spain, in 1984. He received the M.Sc. degree in computer science from the University of Córdoba, Spain, in 2008, the M.Sc. degree in artificial intelligence from the University of Málaga, Spain, in 2009 and the Ph.D. degree in computer science and artificial intelligence from the University of Málaga, in 2011. He is currently a Research Fellow in computational management with the European Space Agency, Noordwijk, The Netherlands. His current research interests include neural networks, ordinal regression, imbalanced classification and hybrid algorithms.

César Hervás-Martínez was born in Cuenca, Spain. He received the B.S. degree in Statistics and Operations Research from the “Universidad Complutense”, Madrid, Spain, in 1978, and the Ph.D. degree in Mathematics from the University of Seville, Spain, in 1986. He is currently a Professor of Computer Science and Artificial Intelligence in the Department of Computer Science and Numerical Analysis, University of Córdoba, and an Associate Professor in the Department of Quantitative Methods, School of Economics, University of Córdoba. His current research interests include neural networks, evolutionary computation, and the modelling of natural systems.