Numerical variable reconstruction from ordinal categories based on probability distributions

J. Sánchez-Monedero[†], M. Carbonero-Ruz[‡], D. Becerra-Alonso[‡], F.J. Martínez-Estudillo[‡], P.A. Gutiérrez[†], C. Hervás-Martínez[†]
[†]*Department of Computer Science and Numerical Analysis*
Universidad de Córdoba
Email: {jsanchezm,pagutierrez,chervas}@uco.es

[‡]*Department of Management and Quantitative Methods*
ETEA
Email: {mariano,dbecerra,fjmestud}@etea.com

***Abstract*—Ordinal classification problems are an active research area in the machine learning community. Many previous works adapted state-of-art nominal classifiers to improve ordinal classification so that the method can take advantage of the ordinal structure of the dataset. However, these method improvements often rely upon a complex mathematical basis and they usually are attached to the training algorithm and model. This paper presents a novel method for generally adapting classification and regression models, such as artificial neural networks or support vector machines. The ordinal classification problem is reformulated as a regression problem by the reconstruction of a numerical variable which represents the different ordered class labels. Despite the simplicity and generality of the method, results are competitive in comparison with very specific methods for ordinal regression.**

***Keywords*-ordinal classification; ordinal regression; support vector machine; neural networks**

I. Introduction

Ordinal classification (so called ranking, sorting or ordinal regression) is a supervised learning problem of predicting categories that have an ordered arrangement. The samples are labeled by a set of ranks with an ordering amongst different categories. In contrast to the usual 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 classification and regression.

Ordinal classification problems are important, since they are common in our everyday life where many problems require classification of items into naturally ordered classes. Selecting the best route to work, where to stop, which product to buy, and where to live, are examples of daily ordinal decision-making. In this way, ordinal classification is one of the most important components in many applications. This includes multi-criteria decision-making, medicine, risk analysis, university ranking, and information retrieval and filtering.

The machine learning community has mainly focused on the learning of numerical and nominal problems. Compared with general classification problems, much less effort has been devoted to ordinal classification learning. However, in the last decade an increasing number of publications report progress in the artificial learning of ordinal concepts. A brief summary of approaches is presented below.

One simple idea is to transform the ordinal problem to a regular regression by building learners that map the ordinal scale by assigning numerical values [1]. However, an appropriate mapping may be difficult to formulate, because the underlying metric among ordinal labels is unknown for most tasks.

Other authors transform the ordinal classification problem into a nested binary classification problem. By combining these results with the results of these binary classifications we can obtain rating predictions [2], [3].

Other approaches are the commonly called threshold models. 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 in which the samples are projected and a set of thresholds that divide the direction into consecutive intervals representing ordinal categories [4], [5], [6], [7], [8].

Finally, algorithms for learning rank functions are a different way to deal with ordinal classification problems. In [9], the authors design a method for learning a ranking function maximizing a specific statistic on the training data.

In this paper we present a new approach for dealing with ordinal classification problems. The approach lies between the regular regression models and the threshold models. The regular regression is addressed by generating random values from triangular probability distributions. There is a different triangular distribution for modelling (generating) the random values of each ordinal class. For each pattern, these values are considered as the regression response variable. In this

978-1-4577-1675-1

way we turn a classification problem into a regression problem. Then, the methodology is suitable for any kind of regression model. With this problem reformulation we are imposing the class order in the *new* regression dataset, and we assume that the trained regressor should reflect this order.

Regarding the threshold model, the limits of the triangular distributions are used as thresholds for properly assigning a pattern to a class. Even thought the simplicity of this idea, experimental results show that the proposal is competitive regarding state-of-the-art ordinal classification methods, specially when considering an order sensitive performance metric.

The rest of the paper is organized as follows. Section II presents the theoretical basis of our proposal, as well as the implementation of the triangular probability distribution. Section III presents the datasets and state-of-the-art related methods used for the experiments. Finally, last section sums up some conclusions and future work.

II. Numerical Variable Reconstruction

The center of our proposal is to turn a classification problem into a regression problem so that the class structures are reflected in the regression variable. This procedure is called Numerical Variable Reconstruction (NVR). NVR has two main steps: the numerical variable reconstruction itself, i.e., the random values generation during the classifier training procedure; and the classification procedure based on the predicted values of the regression response variable.

A. Theoretical basis

Let us consider an ordinal classification problem with Q classes that we presume ordered by the class labels, i. e. $\mathcal{C}_1 \prec \mathcal{C}_2 \prec \ldots \prec \mathcal{C}_Q$, and the training set $\mathcal{D} = \left\{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in X \subset \mathbb{R}^k,\ y_i \in Y = \{\mathcal{C}_1, \ldots, \mathcal{C}_Q\}\right\}$ $(i = 1, \ldots .N)$ made by N patterns, where $\mathbf{x}$ is a characteristic random k-vector and y is the class it belongs to.

Our goal is to find a classifier g that is capable of assigning, according to the best fit possible, a pattern to its class depending on its characteristics. It should also be designed to include the information related to the ordinality of the classes. Thus, the ordered structure of Y should be used to determine g. This also implies that the order is somehow related to the distribution of patterns in the space of attributes X, and also to the topological distribution of the classes.

A simple and intuitive way to make this calculation is to assume the existence of a one-dimensional latent variable $z = \phi(\mathbf{x})$ that is a function of the characteristics observed and takes on the underlying order mentioned in the previous paragraph. The concept of order is included in that the lower values of z will most likely correspond to class $\mathcal{C}_1$, reaching a limit, say z_1, on class $\mathcal{C}_2$ and so on for the rest of the classes. Thus, the classifier will be given by

$$g(\mathbf{x}) = \mathcal{C}_q \text{ if } \phi(\mathbf{x}) \in (z_{q-1}, z_q)$$

where we allow $(z_0, z_Q) = (-\infty, \infty)$.

This is one of the approaches to this problem, obtaining different solutions depending on the kind of function considered for ϕ and the strategy used to determine the limit values for z_i. The drawback here is how to choose an optimal projection function, such that the limits can be adequately adjusted comparing the observed classifications and estimated over the set $\mathcal{D}$.

Our approach does not attempt to determine z (and therefore ϕ). Instead, we present it as a random variable according to the distribution function $u = F(z) = G(\mathbf{x}) = F(\phi(\mathbf{x}))$, where due to the inherent monotony in F, we can have some advantages with respect to the previously mentioned procedure:

- $F(z)$ is a uniform random variable in the interval $[0, 1]$, regardless of how z is a function of $\mathbf{x}$. Thus, not knowing ϕ and F is no longer inconvenient.
- Since the values for z_q determine the limits between classes, and the distribution function is monotonous, its images $u_q = F(z_q)$ will restrain the probabilities to the interval that belongs to each class. Also, since the relative size f_i of those intervals in $\mathcal{D}$ estimates its probability, the boundaries will be given by

$$0 = u_0 < \hat{u}_1 = f_1 < \hat{u}_2 = f_1 + f_2 < \ldots < u_Q = 1$$

Provided all this, each pattern in $\mathcal{D}$ is assigned to a random number. In order to do this, and assuming we are distributing pattern i that belongs to class $\mathcal{C}_q$, we generate a value in the interval $(\hat{u}_{q-1}, \hat{u}_q)$ using the F_q probability distribution, which generates values in that interval. Note there is a different F_q modelling each class $\mathcal{C}_q$. Calling this value U_i and bearing in mind the relationship between u and $\mathbf{x}$ we have a new set $\mathcal{D}' = \left\{(\mathbf{x}_i, U_i) : \mathbf{x}_i \in X \subset \mathbb{R}^k,\ U_i \in [0, 1]\right\}$ where the new response values meet two conditions:

- They indicate the class they belong to, given the way they have been assigned.
- They can be obtained theoretically using $U = G(\mathbf{x})$.

Thus, the initial classification problem becomes a regression one, expected to be easier to solve in order to estimate the function G. It also allows for the use of estimated regression, common to classification problems, as an additional reliability test.

Once the value for $\hat{G}$ is obtained (estimated from G), the classification of new patterns becomes immediate assigning class q as long as we can verify $\hat{u}_{q-1} < \hat{G}(\mathbf{x}) < \hat{u_q}$.

The training procedure, then, can be stated as follows:

1) Calculate, using the training set $\mathcal{D}$, the a priori probabilities of each one of the classes: $f_q = \frac{N_q}{N}$ where N_q and N are the sizes with respect to class $\mathcal{C}_q$ and the set $\mathcal{D}$.

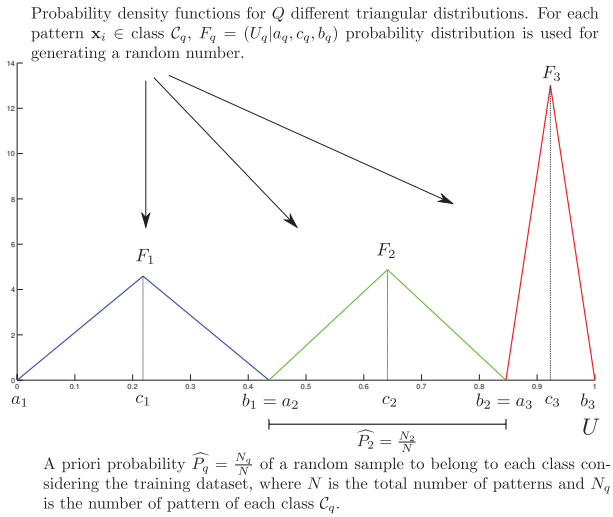

Figure 1. NVR with triangular probability distributions example.

2) Obtain the classification limits using the empirical distribution function $\hat{u}_q = \sum_{j=0}^{q} f_j$.
3) For each N_q patterns belonging to class $\mathcal{C}_q$, generate a random value U_i using the triangular symmetric distributions in the interval between $\hat{u}_{q-1}$ and $\hat{u}_q$.
4) Make a regression model considering the sample $(\mathbf{x}, U)$ where vectors $\mathbf{x}$ of each class are paired with the U values of the previous step. Let $\hat{G}$ be the resulting regressor, which is independent of the model and the training algorithm.

Finally, for every unseen pattern $\mathbf{x}$, $\hat{G}(\mathbf{x})$ is determined and assigned to class $\mathcal{C}_q$, verifying $\hat{u}_{q-1} < \hat{G}(\mathbf{x}) < \hat{u}_q$.

B. NVR using the triangular probability distribution

In order to simplify the mathematical issues regarding the underlying probability distribution for $\mathcal{U}$ modelling, we choose the triangular distribution for this model. The triangular distribution is a continuous probability distribution with lower limit a, upper limit b and mode c, where $a < b$ and $a \leq c \leq b$. Since the triangular distribution is continuous, the inverse transform sampling method can be used to generate random values from the uniform distribution [10], which is the common distribution provided by every programming language or environment.

For a Q class problem, $\mathcal{U}_q, q = 1, 2, \ldots, Q$, different random variables are used to generate values representing all the patterns $\mathbf{x} \in \mathcal{C}_q$. Therefore, Q triangular distributions can be used for modelling these $\mathcal{U}_q$ random variables which compose $\mathcal{U}$. Thus, each triangular distribution will be defined as $F_q(\mathcal{U}_q|a_q, b_q, c_q)$.

Every triangular distribution must be adapted based on the assumptions done in the previous section. Firstly, the a_q and b_q parameters must be adjusted so that the $b_q - a_q$ distance is proportional to the a priori probability N_q/N a pattern has of belonging to a class $\mathcal{C}_q$. Regarding parameter c_q, for simplicity, it is defined as the middle point between a_q and b_q. In our approach, all the triangular distributions are adjusted under the restriction of $\mathcal{U} \in [0, 1]$

Once the triangular distributions are set up, they can be used for generating random values z_i for each pattern $\mathbf{x}_i$ by using the corresponding $F_q(\mathcal{U}_q|a_q, b_q, c_q)$, therefore:

$$z_i = F_q(\mathcal{U}_q|a_q, b_q, c_q) \text{ for } \mathbf{x}_i \in \mathcal{C}_q. \quad (1)$$

At the end of this procedure the ordinal classification problem will be turned into a regression problem in the form of $\hat{G} = \phi(\mathbf{x})$, where ϕ is a regression model.

Assuming we have a trained regression model ϕ for predicting $\hat{G}$ values, these estimated values must be mapped to classes in order to perform the original classification task. In the case of the triangular distribution, this is straightforward. That is because the Q different thresholds u_q, described in Subsection II-A, correspond to the a_{q+1} and b_q parameters. Then, for classifying a pattern in class $\mathcal{C}_1$, $0 \leq \hat{G} \leq u_1$, where $u_1 = b_1 = a_2$. A pattern will be assigned to class $\mathcal{C}_2$, if $u_1 < \hat{G} \leq u_2$, where $u_2 = b_2 = a_3$, and so on. Regarding the last class $\mathcal{C}_Q$, patterns will be classified in this class if $u_{Q-1} < \hat{G} \leq 1$, where $u_{Q-1} = b_{Q-1}$. An example of the triangular distributions parameters set up can be seen in Fig. 1.

III. Experiments

A. Ordinal classification datasets and experimental design

Up to the author's knowledge, there are no public specific datasets repositories for ordinal classification. The most used dataset repository in the literature is the *ordinal regression benchmark datasets* provided by Chu et. al [11]. However, the benchmark datasets provided by Chu et. al, are not real ordinal classification datasets but regression problems. These datasets are turned from a regression problem into a classification problem by discretizing the target variable into r different bins, with equal frequency or equal width, so each bin is labeled as a different ordinal class.

We identify two problems regarding the *ordinal regression benchmark datasets*. First, the datasets are not real classification datasets, therefore issues such as class mislabel or class imbalance related to the problems nature are not present here. Second, using same width intervals for classes label generation produce an artificial class imbalance in the datasets. This artificial class imbalance creates new problems in the experimental design, so that some dataset partitions do not have patterns belonging to all the classes.

Due to the reasons mentioned above, we have collected a set of real ordinal classification datasets which are publicly available at the UCI repository [12] and at the *mldata.org* datasets repository [13] (see Table I for datasets description).

Regarding the experimental design, we have considered two different options depending on the stochastic nature

Table I
DATASETS USED FOR THE EXPERIMENTS

Dataset	Size	#Input	#Classes	Classes Distribution
automobile	205	71	6	(3,22,67,54,32,27)
balance-scale	625	4	3	(288,49,288)
ERA	1000	4	9	(92,142,181,172,158,118,88,31,18)
LEV	1000	4	5	(93,280,403,197,27)
tae	151	54	3	(49,50,52)

of the method. The proposed method (NVR) is non-deterministic because the U_i values are randomly generated using the triangular distributions and the seed used for the method determines the obtained results. For this method, we perform 10 times a holdout validation and 3 repetitions for each holdout (obtaining a total of $10 \times 3 = 30$ different results). Each holdout is a stratified random division of the data, where approximately 75% of the instances are used for the training set and 25% of them for the test set (maintaining the original distribution of classes for both sets). For the deterministic methods (all of them except NVR), we perform 30 times a stratified holdout validation using 75% of the instances for the training set and 25% of them for the generalization set, what implies a total of 30 different results. The partitions are the same for all the deterministic methods.

In this way, a total of 30 error measures has been obtained for all the methods compared, which guarantees a proper statistical significance of the results.

B. Machine learning methods used for comparison purposes

For comparison purposes, different state-of-the-art methods have been included in the experimentation. These methods are the following:

- **Gaussian Processes for Ordinal Regression (GPOR)** by Chu et. al [11], 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 [14] and [15] 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)** by Chu et. al [8][16], 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 SVOR-EX. 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 (SVOR-IM).
- **SVM-Rank** [17] applies the Extended Binary Classification (EBC) method to SVM. The EBC method can be summarized in the following three steps. First, transform all training samples into extended samples weighting these samples by using the absolute 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.
- **Support Vector Machine (SVM)** [18], [19] nominal classifier is included in the experiments in order to validate our proposal contributions. Cost Support Vector Classification (SVC) available in libSVM 3.0 [20] is used as the SVM classifier implementation.

In our approach, the Support Vector Regression (SVR) algorithm is used so the method is call SVR-NVR. The ϵ-SVR available in libSVM is used.

Regarding the algorithms' hyper-parameters, the following procedure has been applied. For the Support Vector algorithms, i.e. SVC, SVMRank, SVOR-EX, SVOR-IM and ϵ-SVR, the corresponding hyper-parameters (regularization parameter, C, and width of the Gaussian functions, γ), were adjusted using a grid search with a 10-fold cross-validation, considering the following ranges: $C \in \{10^3, 10^1, \ldots, 10^{-3}\}$ and $\gamma \in \{10^3, 10^0, \ldots, 10^{-3}\}$. Regarding ϵ-SVR, it has the additional ϵ parameter. The ϵ parameter values were $\epsilon \in \{10^3, 10^1, \ldots, 10^0\}$. For GPOR-ARD no hyper-parameters were set up since the method optimizes the associated parameters itself. All the methods were configured to use the Gaussian kernel.

C. Ordinal classification evaluation metrics

Two evaluation metrics have been considered which quantify 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) Mean Zero-one Error (MZE) is simply the fraction of incorrect predictions on individual samples:

$$MZE = \frac{1}{N} \sum_{i=1}^{N} I\left(\hat{y}_i \neq y_i\right), \tag{2}$$

Table II
COMPARISON OF THE PROPOSED METHOD TO OTHER ORDINAL CLASSIFICATION METHODS AND SVC: RESULTS OF MZE ($MZE_{\mathrm{G}}(\%)$) AND MAE (MAE_G) ON THE GENERALIZATION SET

Dataset	Method	MZE mean	MZE std	MAE mean	MAE std
automobile	SVC	*0.3423*	*0.0659*	0.4205	0.0837
	GPOR-ARD	0.3891	0.0726	0.5942	0.1307
	SVMRank	**0.3160**	**0.0548**	**0.3929**	**0.0730**
	SVOR-EX	0.6333	0.0685	0.9276	0.1149
	SVOR-IM	0.6385	0.0552	0.9327	0.0918
	SVR-NVR	0.3449	0.0746	*0.4128*	*0.1052*
balance-scale	SVC	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	GPOR-ARD	0.0342	0.0118	0.0342	0.0118
	SVMRank	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	SVOR-EX	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	SVOR-IM	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	SVR-NVR	0.0571	0.0213	0.0597	0.0229
ERA	SVC	0.7492	0.0194	1.2575	0.0744
	GPOR-ARD	**0.7121**	**0.0270**	1.2413	0.0505
	SVMRank	0.7551	0.0239	1.2172	0.0431
	SVOR-EX	*0.7349*	*0.0305*	*1.2108*	*0.0363*
	SVOR-IM	0.7579	0.0213	1.2143	0.0343
	SVR-NVR	0.7409	0.0182	**1.1996**	**0.0528**
LEV	SVC	**0.3723**	**0.0297**	0.4080	0.0343
	GPOR-ARD	0.3877	0.0301	0.4219	0.0308
	SVMRank	0.3764	0.0238	0.4133	0.0265
	SVOR-EX	*0.3724*	*0.0218*	*0.4073*	*0.0244*
	SVOR-IM	0.3757	0.0289	0.4119	0.0318
	SVR-NVR	0.3744	0.0304	**0.4007**	**0.0308**
tae	SVC	0.5281	0.0896	0.5886	0.0834
	GPOR-ARD	0.6719	0.0407	0.8614	0.1551
	SVMRank	0.4781	0.0735	0.5149	0.0865
	SVOR-EX	*0.4421*	*0.0752*	*0.4816*	*0.0922*
	SVOR-IM	0.4711	0.0836	0.5149	0.0851
	SVR-NVR	**0.4044**	**0.0305**	**0.4360**	**0.0256**

Table III
AVERAGE MZE AND MAE RESULTS AND MEAN RANKINGS

	SVR-NVR	SVC	GPOR-ARD	SVMRank	SVOR-EX	SVOR-IM
$\overline{MZE}_{\mathrm{G}}$	**0.3843**	0.4390	0.3984	*0.3851*	0.4365	0.4486
$\overline{R}_{MZE_{\mathrm{G}}}$	3.2	4.4	*2.9*	3.5	**2.7**	4.3
$\overline{MAE}_{\mathrm{G}}$	**0.5018**	0.6306	0.5349	*0.5077*	0.6055	0.6148
$\overline{R}_{MAE_{\mathrm{G}}}$	**2.2**	5.2	3.9	3.2	*2.7*	3.8

where $I(\cdot)$ is the zero-one loss function and N is the number of patterns of the dataset.

2) Mean Absolute Error (MAE) is the average deviation of the prediction from the true targets, i.e.:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|\mathcal{O}(\hat{y}_i) - \mathcal{O}(y_i)\right|, \qquad (3)$$

where $\mathcal{O}(\mathcal{C}_k) = k, 1 \leq k \leq K$, i.e. $\mathcal{O}(y_i)$ is the order of class label y_i.

These measures are aimed to evaluate two different aspects that can be taken into account when an ordinal regression problem is considered: whether the patterns are generally well classified (MZE) and whether the classifier tends to predict a class as close to the real class as possible (MAE).

D. Results

The experiments have been carried out by following the experimental design described in Subsection III-A. Results considering mean and standard deviation in MZE and MAE are showed in Table II. The best statistical result is in bold face and the second best result in italics.

In order to compare these results, a non-parametric Friedman [21] test has been employed. The average MZE_{G} and average MAE_{G} results in the generalization sets were used for the Friedman test. The test accepted the null-hypothesis that all algorithms perform equally well when $\alpha = 0.05$. Therefore, we can conclude that our method reaches the state-of-the-art methods regarding ordinal classification. In

addition, for information purposes, Table III shows the average results in MZE_{G} and MAE_{G} for each method, as well as the average ranking for each performance metric, this is $\overline{R}_{MZE_{\mathrm{G}}}$ and $\overline{R}_{MAE_{\mathrm{G}}}$. Table III shows that SVR-NVR has the best average MZE results and the third mean MZE rank, whereas it has the best performance for average MAE results and mean ranking.

IV. Conclusions

In this paper, a novel method for generally adapting classification and regression models, such as artificial neural networks or support vector machines, was presented. The ordinal classification problem is reformulated as a regression problem by the reconstruction of a numerical variable, which represents the different ordered class labels.

The NVR algorithm is implemented with the triangular probability distribution for the variable reconstruction, and with the Support Vector Regression algorithm as the regression method. Experimental results demonstrate that our method reaches the state-of-the-art related algorithms. Despite the simplicity and generality of the method, results are competitive in comparison with very specific methods for ordinal regression.

For future work, several issues may be studied. For example, it could be suitable to study how the parameters of the triangular distributions impact on the classifier performance. In addition, the relevance of the underlying probability distribution could also be studied. Finally, it may be appropriate to look at other regression methods for NVR.

Acknowledgement

Javier Sánchez-Monedero's research has been funded by the "Junta de Andalucía" Ph. D. Student Program. This work has been partially subsidized by the TIN 2008-06681-C06-03 project of the Spanish Inter-Ministerial Commission of Science and Technology (MICYT), FEDER funds and the P08-TIC-3745 project of the "Junta de Andalucía" (Spain).

References

[1] S. Kramer, G. Widmer, B. Pfahringer, and M. de Groeve, "Prediction of ordinal classes using regression trees," in *Foundations of Intelligent Systems*, ser. Lecture Notes in Computer Science, Z. Ras and S. Ohsuga, Eds. Springer Berlin / Heidelberg, 2010, vol. 1932, pp. 665–674.

[2] E. Frank and M. Hall, "A simple approach to ordinal classification," in *Proceedings of the 12th European Conference on Machine Learning*, ser. EMCL '01. London, UK: Springer-Verlag, 2001, pp. 145–156.

[3] W. Waegeman and L. Boullart, "An ensemble of weighted support vector machines for ordinal regression," *International Journal of Computer Systems Science and Engineering*, vol. 3, no. 1, 2009.

[4] P. McCullagh, "Regression models for ordinal data," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 42, no. 2, pp. 109–142, 1980.

[5] J. Verwaeren, W. Waegeman, and B. D. Baets, "Learning partial ordinal class memberships with kernel-based proportional odds models," *Computational Statistics & Data Analysis*, vol. In Press, Corrected Proof, pp. –, 2010.

[6] R. Herbrich, T. Graepel, and K. Obermayer, "Large margin rank boundaries for ordinal regression," in *Advances in Large Margin Classifiers*, A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. Cambridge, MA: MIT Press, 2000, pp. 115–132.

[7] K. Crammer and Y. Singer, "Pranking with ranking," in *Advances in Neural Information Processing Systems 14*. MIT Press, 2001, pp. 641–647.

[8] W. Chu and S. S. Keerthi, "New approaches to support vector ordinal regression," in *In ICML'05: Proceedings of the 22nd international conference on Machine Learning*, 2005, pp. 145–152.

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

[10] L. Devroye, *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, 1986.

[11] W. Chu and Z. Ghahramani, "Gaussian processes for ordinal regression," *Journal of Machine Learning Research*, vol. 6, pp. 1019–1041, 2005.

[12] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[13] D. S. Sonnenburg, "Machine Learning Data Set Repository," 2011. [Online]. Available: http://mldata.org/

[14] D. J. C. Mackay, "Bayesian methods for backpropagation networks," in *Models of Neural Networks III*, E. Domany, J. L. van Hemmen, and K. Schulten, Eds. Springer-Verlag, New York, 1994, ch. 6, pp. 211–254.

[15] R. M. Neal, *Bayesian Learning for Neural Networks*. Secaucus, NJ, USA: Springer-Verlag, New York, 1996.

[16] W. Chu and S. S. Keerthi, "Support Vector Ordinal Regression," *Neural Computation*, vol. 19, no. 3, pp. 792–815, 2007.

[17] L. Li and H.-T. Lin, "Ordinal regression by extended binary classification," *Advances in Neural Information Processing Systems*, vol. 19, pp. 865–872, 2007.

[18] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[19] V. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, 1999.

[20] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[21] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *Annals of Mathematical Statistics*, vol. 11, no. 1, pp. 86–92, 1940.