Integrating Bayesian network classifiers to deal with the partial label ranking problem

Juan C. Alfaro JuanCarlos.Alfaro@uclm.es
Laboratorio de Sistemas Inteligentes y Minería de Datos, Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, Albacete, Spain

Juan A. Aledo JuanAngel.Aledo@uclm.es
Laboratorio de Sistemas Inteligentes y Minería de Datos, Departamento de Matemáticas, Universidad de Castilla-La Mancha, Albacete, Spain

José A. Gámez Jose.Gamez@uclm.es
Laboratorio de Sistemas Inteligentes y Minería de Datos, Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, Albacete, Spain

Abstract

The *label ranking problem* consists in learning *preference models* from training datasets labeled with a (*possibly incomplete*) *ranking* of the class labels, and the goal is to predict a ranking for a given unlabeled instance. In this work, we focus on the particular case where the training dataset and the prediction given as output allow *tied* class labels (i.e., there is no particular preference among them), known as the *partial label ranking problem*. This paper transforms the *ranking with ties* into discrete variables representing the preference relations (*precedes*, *ties*, and *succeeds*) among pairs of class labels. We then use *Bayesian network classifiers* to model the pairwise preferences. Finally, we input the *posterior probabilities* into the *pair order matrix* used to solve the corresponding *rank aggregation problem* at inference time. The experimental evaluation shows that our proposals are competitive in accuracy with the state-of-the-art *mixture-based probabilistic graphical models* while being much faster.

Keywords: Naive Bayes; Averaged one-dependence estimators; Bayesian network classifiers; (Partial) label ranking

1. Introduction

The *label ranking* (*LR*) *problem* (Cheng et al., 2009) is a natural extension of *conventional classification*, the goal of which is to predict a *ranking* (a.k.a. *permutation*) of the *class labels* instead of a single one. It is worth noting that, although the LR problem considers *ranking with ties* (a.k.a. *partial rankings*), in general, the algorithms available in the literature are constrained to output a ranking. Therefore, we refer to this more general interpretation as the *partial label ranking* (*PLR*) *problem* (Alfaro et al., 2021a), where (*possibly incomplete*) partial rankings are allowed in the training datasets, and a partial ranking is requested as prediction.

A variety of methods exists for tackling the PLR problem. For instance, we may find methods that adapt machine learning algorithms to deal with the new target structure, such as *nearest neighbors* (Alfaro et al., 2021a), *decision trees* (Alfaro et al., 2021a), and *mixture-based algorithms* (Alfaro et al., 2021b). Moreover, *averaging ensemble methods* of

decision trees are proposed in (Alfaro et al., 2022), where the performance over a single classifier is improved.

This paper proposes transforming the PLR problem into a set of conventional classification problems, and then using *Bayesian network classifiers* to deal with the new problem structure. Thus, we introduce a *pairwise variable* with the *preference information* (*precedes*, *ties*, and *succeeds*) for each pair of class labels. We use this pairwise variable in several ways, which give rise to the methods proposed in this paper: *pairwise classifiers*, *classifier chains* (Read et al., 2009), and *bivariate classifiers*. The probabilities of the predictions obtained for the pairwise variables are then used for constructing the pair order matrix used to solve the corresponding *rank aggregation problem*, i.e., the *optimal bucket order problem* (*OBOP*) (Gionis et al., 2006; Ukkonen et al., 2009).

This paper is organized as follows. Section 2 revises the PLR problem, the OBOP, and the two base estimators used for our proposals: *naive Bayes* and *averaged one-dependence estimators* (Webb et al., 2005). Section 3 introduces the Bayesian network classifiers proposed in this paper. Section 4 presents the experimental evaluation carried out. Finally, Section 5 concludes the paper and provides some lines for future work.

2. Preliminaries

This section reviews some notions to be considered throughout the paper: *optimal bucket order problem* (Gionis et al., 2006; Ukkonen et al., 2009), *naive Bayes classifier* and *averaged one-dependence estimators* (Webb et al., 2005).

2.1 Optimal bucket order problem

The *optimal bucket order problem* (*OBOP*) (Gionis et al., 2006; Ukkonen et al., 2009) is a *rank aggregation problem* which obtains a *ranking with ties* (a.k.a. *partial ranking, bucket order*) as consensus from a set of (possibly incomplete) partial rankings.

Before formalizing the OBOP, let us revise some related concepts. A *bucket order* $\mathcal{B}$ (Fagin et al., 2004) is an ordered sequence of k disjoint subsets (*buckets*), $\mathcal{B}_1, \ldots, \mathcal{B}_k$ defined over a finite set of items $\mathcal{I} = \{1, \ldots, n\}$, where $1 \leq k \leq n$ and $\cup_{i=1}^{k} \mathcal{B}_i = \mathcal{I}$. In particular, given two buckets $\mathcal{B}_i, \mathcal{B}_j \in \mathcal{B}$, we write $\mathcal{B}_i \succ \mathcal{B}_j$ to express that $\mathcal{B}_i$ precedes $\mathcal{B}_j$ in $\mathcal{B}$. In the same way, given two items $u, v \in \mathcal{I}$, we write $u \succ_{\mathcal{B}} v$ to express that u is preferred to v in $\mathcal{B}$ and $u \sim_{\mathcal{B}} v$ if u and v are tied. A bucket order $\mathcal{B}$ may be represented with an $n \times n$ *bucket matrix* B (Ukkonen et al., 2009) such that $B(u, v) = 1$, if $u \succ_{\mathcal{B}} v$; $B(u, v) = 0$, if $v \succ_{\mathcal{B}} u$; and $B(u, v) = 0.5$, if $u \sim_{\mathcal{B}} v$, for $u, v \in \mathcal{I}$. In particular, $B(u, u) = 0.5$ for all $u \in \mathcal{I}$, and $B(u, v) + B(v, u) = 1$ for all $u, v \in \mathcal{I}$ with $u \neq v$.

A *pair order matrix* (Ukkonen et al., 2009) is an $n \times n$ matrix C, satisfying that $C(u, v) \in [0, 1]$ for all $u, v \in \mathcal{I}$, with $C(u, v) + C(v, u) = 1$ if $u \neq v$ and $C(u, u) = 0.5$. Thus, a pair order matrix C may be viewed as a precedence matrix, where the entry $C(u, v)$, for $u, v \in \mathcal{I}$, expresses the probability of item u preceding item v. If we consider the matrix distance

$$D(B, C) = \sum_{u,v \in \mathcal{I}} |B(u, v) - C(u, v)|, \tag{1}$$

where C is a pair order matrix (maybe obtained from a set of (possibly incomplete) partial rankings), the goal of the OBOP is to find the bucket matrix B (associated with a bucket order $\mathcal{B}$) which minimizes $D(B, C)$.

Several algorithms have been proposed to tackle the OBOP (Gionis et al., 2006; Ukkonen et al., 2009; Aledo et al., 2017, 2018, 2021). In this work, we use the instance named LIA_G^{MP2} of the BPA_{LIA} algorithm (Aledo et al., 2017) to balance between accuracy and computational efficiency.

2.2 Partial label ranking problem

The *partial label ranking* (*PLR*) *problem* (Alfaro et al., 2021a) can be seen as a more general interpretation of the *label ranking* (*LR*) *problem* (Cheng et al., 2009). In the standard LR scenario, every instance $\mathbf{x} = (x_1, \ldots, x_m)$ from an input space $\mathbf{X} = \text{dom}(X_1) \times \cdots \times \text{dom}(X_m)$, where m is the number of *predictive variables* (a.k.a. *features*), is associated with a *total order* defined over the values in the domain of the class variable Y, $\text{dom}(Y) = \{y_1, \ldots, y_n\}$. In the PLR problem, instead of associating every instance $\mathbf{x}$ with a total order, $\mathbf{x}$ is associated with a bucket order defined over the values in $\text{dom}(Y)$. It is worth pointing out that although it is possible to allow incomplete partial rankings in the training dataset, we deal only with complete ones in this paper.

2.3 Naive Bayes classifier

Naive Bayes classifiers are well-known *probabilistic graphical models* based on the assumption of conditional independence between every pair of features given the class variable. Naive Bayes classifiers follow the *maximum a posteriori* (*MAP*) *estimation principle*, i.e., they predict the most probable class label $y \in \text{dom}(Y)$ given an input instance $\mathbf{x} \in \mathbf{X}$ as evidence

$$y^* = \underset{y \in \text{dom}(Y)}{\arg\max}\, \mathbf{P}(y \mid \mathbf{x}) = \underset{y \in \text{dom}(Y)}{\arg\max}\, \frac{\mathbf{P}(\mathbf{x}, y)}{\mathbf{P}(\mathbf{x})} = \underset{y \in \text{dom}(Y)}{\arg\max}\, \mathbf{P}(y) \cdot \prod_{i=1}^{m} \mathbf{P}(x_i \mid y) \tag{2}$$

according to *Bayes' theorem*, the naive conditional independence assumption, and also using that $\mathbf{P}(\mathbf{x})$ is constant given the input.

The different naive Bayes classifiers vary depending on their assumptions regarding the distribution of the above conditional distributions.

2.4 Averaged one-dependence estimators classifier

Averaged one-dependence estimators (*AODE*) (Webb et al., 2005) is a probabilistic classification learning technique that seeks to address the primary concern of the naive Bayes classifiers, i.e., the conditional independence assumption. To do that, it constructs a separate ODE for each feature X_j, $1 \leq j \leq m$, such that the remaining features X_i, where $1 \leq i \leq m$ and $i \neq j$, depend on their corresponding parent feature X_j and the class variable Y. Thus, ODE estimates the joint probability for a class label $y \in \text{dom}(Y)$ given an input instance $\mathbf{x} \in \mathbf{X}$ as follows

$$\mathbf{P}(\mathbf{x}, y) = \mathbf{P}(x_j, y) \cdot \prod_{i=1}^{m} \mathbf{P}(x_i \mid x_j, y). \tag{3}$$

In this sense, ODE considers a weaker independence hypothesis than naive Bayes classifiers. Therefore, ODE creates a classifier with less bias, although it slightly increases the variance because the base probability estimates are conditioned by two variables rather than one. To reduce this variance, AODE averages the probability estimates provided by each ODE. Finally, AODE obtains the prediction using the MAP estimation principle as in naive Bayes classifiers

$$y^* = \underset{y \in \mathrm{dom}(Y)}{\arg\max} \sum_{j=1}^{m} \mathbf{P}(x_j, y) \cdot \prod_{\substack{i=1 \\ i \neq j}}^{m} \mathbf{P}(x_i \mid x_j, y). \tag{4}$$

3. Bayesian network classifiers for the partial label ranking problem

This section describes the *Bayesian network classifiers* proposed to deal with the PLR problem, mainly inspired by the *multi-label scenario*: *pairwise classifier*, *classifier chains* (Read et al., 2009), and *bivariate classifier*.

First, we define a pairwise variable, the critical point to transform the PLR problem into a set of standard classification problems (followed by an aggregation procedure to solve the OBOP). We introduce a discrete variable Z_{uv} to codify the preference relation (*precedes*, *ties*, and *succeeds*) between two class labels $y_u, y_v \in \mathrm{dom}(Y)$ such that $Z_{uv} = z_1$, if $y_u \succ y_v$; $Z_{uv} = z_2$, if $y_u \sim y_v$; and $Z_{uv} = z_3$, if $y_v \succ y_u$. We then train Bayesian network classifiers to obtain the posterior probability distribution for all the Z_{uv} variables. Finally, these values are the entries of the pair order matrix C required to solve the OBOP. Note that it is not necessary to compute all the pairwise variables Z_{uv} given that $C(u, v) = 1 - C(v, u)$. The discrete variables Z_{uv} are used in various ways by the models proposed. Let us describe each of them.

3.1 Pairwise classifier

The pairwise classifier proposed in this paper relates to the *binary relevance method* used in the *multi-label classification scenario*. In the same way that the binary relevance method decomposes the multi-label problem into several independent binary learning tasks, the pairwise classifier assumes that there are no relations among the Z_{uv} variables. Although this independence assumption among the targets is not valid for the general case, they generally lead to fast methods with good accuracy results.

Figures 1a and 1b show the representation of the pairwise classifier for both naive Bayes and AODE as base estimators. In particular, the pairwise classifier learns an estimator for each pair of class labels $y_u, y_v \in \mathrm{dom}(Y)$ using the Z_{uv} variable as a class label, where $1 \leq u < v \leq n$. Although the AODE classifier considers dependencies among the predictive features, naive Bayes and AODE assume independence among the Z_{uv} class variables.

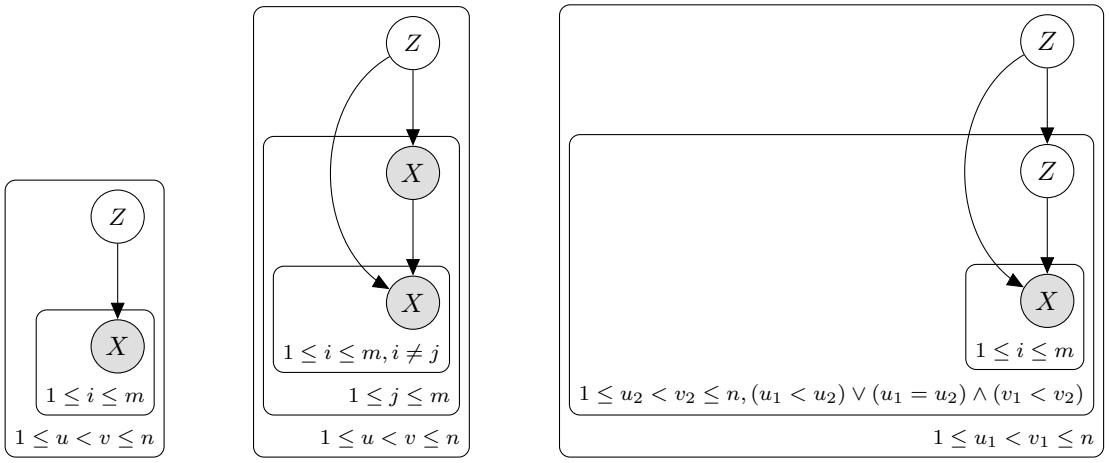

Figure 1: Representation of the Bayesian network classifiers: (a) Pairwise classifier + Naive Bayes, (b) Pairwise classifier + AODE, and (c) Bivariate classifier + Naive Bayes

3.2 Classifier chain

Classifier chains (Read et al., 2009) combine multiple conventional classification models into a single *multi-label* or *multi-dimensional* model, in a such a way that they are capable of exploiting the correlations among the class variables. To do so, a chain order among the models is defined such that each one of them is then fit to the training dataset plus the class variables that were assigned a lower order in the chain. At inference time, since the true class labels for each variable are not available, the predictions are used as features by the subsequent models in the chain.

In this paper, we transform the PLR problem into a multi-dimensional classification problem, given that each pairwise variable Z_{uv} may be seen as a class variable. Therefore, we can apply classifier chains to exploit the possible correlations among pairs of class labels $y_u, y_v \in \text{dom}(Y)$.

Note that the order of the chain is essential. In this sense, the first model has no information regarding the other class variables, while the last model has features indicating the presence of all the remaining ones. In our case, we consider two possible orders for the chain:

- Random. We take a random ordering of the class labels.

- Heuristic. We apply a modified version of the *Borda count algorithm* (Borda, 1770) to deal with rankings with ties, where we assign the same number of points to the tied class labels. The ordering is then considered according to the number of points, from the highest to the lowest. In the case of ties, the class label with the lowest index is selected first.

3.3 Bivariate classifier

Contrary to the pairwise classifier described in Section 3.1, the bivariate classifier considers all the combinations among every pair of pairwise variables as class labels. Therefore, one may relate the *label powerset method* of the multi-label scenario with the bivariate classifier. In this sense, the label powerset method trains a classifier for all the unique class label combinations found in the training dataset, and the bivariate classifier learns an estimator for every two pairs of class labels.

Figure 1c shows the representation of the bivariate classifier for naive Bayes as a base estimator. Specifically, the bivariate classifier learns an estimator for every two pairs of class labels $y_{u_1}, y_{v_1}, y_{u_2}, y_{v_2} \in \text{dom}(Y)$, where $1 \leq u_1 < v_1 \leq n$, $1 \leq u_2 < v_2 \leq n$, and $(u_1 < u_2) \vee (u_1 = u_2) \wedge (v_1 < v_2)$. In this case, the joint probability for a particular pair of class labels $z_k \in \text{dom}(Z_{u_1 v_1})$ and $z_l \in \text{dom}(Z_{u_2 v_2})$ given an input instance $\mathbf{x}$ is computed as follows

$$\mathbf{P}(\mathbf{x}, z_k, z_l) = \mathbf{P}(z_k, z_l) \prod_{i=1}^{m} \mathbf{P}(x_i \mid z_k, z_l). \tag{5}$$

To obtain the conditional probability distribution of each pairwise variable $Z_{u_1 v_1}$, where $1 \leq u_1 < v_1 \leq n$, we have to marginalize Equation 5 according to z_l.

It is worth pointing out that a quartic factor roughly bounds the number of base estimators that the bivariate classifier learns. Thus, although it is suitable for naive Bayes as a base estimator, we exclude AODE due to the high computational complexity. Nevertheless, we may reduce the number of pairs of pairwise classifiers using structural learning (p.e., *Markov blanquet* (Jensen and Nielsen, 2007)).

4. Experimental evaluation

This section compares the algorithms proposed in this paper in terms of accuracy and computational efficiency. Below, we describe the datasets used, the algorithms tested, the methodology adopted, and the results obtained.

4.1 Datasets

Table 1 contains a summary of the benchmark datasets used to test the proposed algorithms (see `https://www.openml.org/search?type=data&sort=runs&status=active&uploader_id=%3D_25829` for the details). In particular, they were obtained by turning standard classification datasets from the UCI repository (Dua and Graff, 2017) into PLR datasets using the procedure proposed in (Alfaro et al., 2021a). Note that the columns display the number of instances (#instances), the number of features (#attributes), the number of class labels of the standard classification problem (#labels), the number of distinct bucket orders in the PLR dataset (#rankings), the mean number of buckets per bucket order ($\overline{\text{buckets}}$), and the field of application (Domain), respectively.

4.2 Algorithms

We included the following algorithms in the experimental evaluation:

Table 1: Description of the datasets

Dataset	#instances	#attributes	#labels	#rankings	$\overline{\text{buckets}}$	Domain
authorship	841	70	4	47	3.063	Sociology
blocks	5472	10	5	116	2.337	Bioinformatics
breast	109	9	6	62	3.925	Medicine
ecoli	336	7	8	179	4.140	Biology
glass	214	9	6	105	4.089	Criminology
iris	150	4	3	7	2.380	Eugenics
letter	20000	16	26	15014	7.033	Letter classification
libras	360	90	15	356	6.889	Hand movement classification
pendigits	10992	16	10	3327	3.397	Digit classification
satimage	6435	36	6	504	3.356	Satellite image classification
segment	2310	18	7	271	3.031	Outdoor image classification
vehicle	846	18	4	47	3.117	Silhouette classification
vowel	528	10	11	504	5.739	Vowel classification
wine	178	13	3	11	2.680	Chemistry
yeast	1484	8	10	1006	5.929	Biology

- Pairwise algorithm (see Section 3.1). As base classifiers, we considered naive Bayes and AODE. We estimated the conditional probability distributions with *univariate normal distributions* for naive Bayes as a base estimator. Moreover, we binned the predictive variables into three intervals using an *equal-width discretization technique*. In this case, we estimated the parameters of the model with *univariate categorical distributions*. We followed the standard approach for AODE, i.e., we considered only categorical features.

- Classifier chain (see Section 3.2). Similar to the previous case, but also estimating the conditional probability distributions of the chain with univariate categorical distributions. Moreover, we considered both the random and heuristic ordering of the chain.

- Bivariate algorithm (see Section 3.3). We considered only naive Bayes as a base classifier due to time complexity problems with AODE.

- *Gaussian mixture* algorithm (Alfaro et al., 2021b). We considered the algorithm modeling the continuous features with a *multivariate normal distribution* and a *tied* covariance matrix, where all components share the same general covariance matrix.

We used the following procedure to decide if numerical or categorical features were employed. First, we trained a standard naive Bayes estimator with both versions (univariate normal distributions and univariate categorical distributions) and used as a class label the one ranked first in the ranking (in case of ties, the one with the lowest index was chosen). We then evaluated the performance of both estimators, and the one with higher accuracy classification score over the training dataset was selected.

4.3 Methodology

We adopted the following methodology decisions for the experimental evaluation according to the existing (P)LR literature (Cheng et al., 2009; Alfaro et al., 2021a):

- The algorithms were evaluated by a *ten-fold cross-validation* method *repeated five times*.
- The accuracy results were derived in terms of the τ_X *rank correlation coefficient*. See Section 4 in (Emond and Mason, 2002) for the details.
- The results were analyzed using the procedure described in (Demšar, 2006; García and Herrera, 2008) with the `exreport` software tool (Arias and Cózar, 2015). First, a *Friedman test* (Friedman, 1940) with the null hypothesis that all the algorithms have equal performance was applied. Second, if this hypothesis was rejected, a *post-hoc test* using *Holm's procedure* (Holm, 1979) was performed to compare all the algorithms against the one ranked first by the Friedman test. Both tests were conducted at a significance level of 5%.

4.4 Results

This section analyzes the results of the algorithms proposed in this paper in terms of accuracy and computational efficiency.

4.4.1 Accuracy

The summary of the accuracy results is shown in Table 2. Each cell contains the mean and the standard deviation of the τ_X rank correlation coefficient between the real and the predicted partial rankings averaged over the test datasets. The algorithm(s) leading to the best accuracy results for each dataset has been highlighted. Before a deeper statistical analysis, we can conclude that the chain using the heuristic order performs better than the random one. Therefore, we decide to omit this method from the statistical analysis from this point on. Moreover, the empty cells correspond to algorithms leading to excessive memory usage errors.

To analyze these results, we followed the procedure described in Section 4.3. Thus, we rejected the null hypothesis that all the algorithms have equal performance with computed p-value of 8.516×10^{-5}. The results for the post-hoc test are shown in Table 3. In particular, the rank and p-value columns contain the ranking provided by the Friedman test and the p-value adjusted by the Holm's procedure, respectively. The columns win, tie and loss refer to the number of times that the algorithm ranked first by the Friedman test wins, ties and loses with respect to the corresponding algorithm. The boldfaced values correspond to non-rejected null hypotheses. According to these results, we can conclude that:

- The Gaussian mixture algorithm is ranked first by the Friedman test. However, three algorithms with significantly less training time are not statistically different. Moreover, these three algorithms can cope with all the datasets, while the experiments of the Gaussian mixture algorithm for the `letter` dataset caused an excessive memory usage error.

Table 3: Post-hoc test results for the mean accuracy

Method	Rank	p-value	Win	Tie	Loss
Gaussian mixture algorithm	2.13	-	-	-	-
Bivariate algorithm + Naive Bayes	2.37	$\mathbf{7.327 \times 10^{-1}}$	9	0	6
Pairwise algorithm + AODE	3.43	$\mathbf{1.141 \times 10^{-1}}$	12	1	2
Pairwise algorithm + Naive Bayes	3.70	$\mathbf{6.548 \times 10^{-2}}$	11	0	4
Classifier chain + AODE	4.47	2.545×10^{-3}	13	0	2
Classifier chain + Naive Bayes	4.90	2.561×10^{-4}	12	1	2

- The classifier chain algorithms result in the worst performance, meaning that they are not capable of exploiting the pairwise correlations.
- The bivariate algorithm is ranked ahead of the pairwise algorithms, and it shows a better performance in almost all the scenarios. Hence, we can improve the performance of the algorithms by considering pairs of pairwise comparisons as possible outcomes in the classifiers.

4.4.2 Computational efficiency

Tables 4 and 5 show the CPU time for the learning and inference steps of the algorithms considered in the experimental evaluation. The CPU time for the whole procedure (learning with the training dataset and validating with the test one) is usually reported. However, since we deal with the same machine learning paradigm, we consider it more interesting to report the values separately. In light of these results, we can conclude that:

- The algorithms with naive Bayes as a base estimator give rise to the fastest classifiers.
- The bivariate algorithm is faster than the pairwise algorithm using AODE as a base classifier, except for those datasets where the number of classes is higher than the number of features (e.g., `letter` or `yeast`), because the number of classes bounds the complexity of the bivariate algorithm. In contrast, the number of features bounds the complexity of the algorithms using AODE as a base classifier.
- The CPU time of the Gaussian mixture algorithm is much higher than the one of the rest of algorithms.

Therefore, considering a balance between accuracy and computational efficiency, the bivariate algorithm becomes the best choice.

5. Conclusions

This paper explores using Bayesian network classifiers to deal with the PLR problem. The main drawback is to model the target variable, as it takes value in the set of bucket orders

Table 2: Mean accuracy for the algorithms

Classifier	Base estimator	Order	authorship	blocks	breast	ecoli	glass	iris	letter	libras	pendigits	satimage	segment	vehicle	vowel	wine	yeast
Pairwise	Naive Bayes	-	0.797 ± 0.021	0.868 ± 0.028	**0.747 ± 0.053**	0.722 ± 0.061	0.707 ± 0.055	0.901 ± 0.037	0.597 ± 0.005	0.600 ± 0.030	0.695 ± 0.006	0.727 ± 0.009	0.781 ± 0.014	0.487 ± 0.041	0.702 ± 0.019	0.845 ± 0.035	0.727 ± 0.010
	AODE	-	0.806 ± 0.023	0.888 ± 0.008	0.627 ± 0.070	0.729 ± 0.025	0.693 ± 0.056	0.872 ± 0.042	0.561 ± 0.005	**0.635 ± 0.025**	0.741 ± 0.006	0.806 ± 0.007	0.795 ± 0.011	0.602 ± 0.037	0.616 ± 0.028	0.827 ± 0.043	0.717 ± 0.011
Chain	Naive Bayes	Random	0.786 ± 0.022	0.819 ± 0.058	0.730 ± 0.060	0.636 ± 0.039	0.650 ± 0.064	0.892 ± 0.043	0.439 ± 0.008	0.553 ± 0.037	0.658 ± 0.006	0.720 ± 0.011	0.767 ± 0.013	0.510 ± 0.047	0.529 ± 0.035	0.840 ± 0.039	0.634 ± 0.020
	Naive Bayes	Heuristic	0.784 ± 0.023	0.839 ± 0.046	0.732 ± 0.059	0.610 ± 0.038	0.708 ± 0.056	0.897 ± 0.043	0.443 ± 0.007	0.569 ± 0.034	0.658 ± 0.006	0.735 ± 0.010	0.768 ± 0.016	0.523 ± 0.044	0.602 ± 0.025	0.839 ± 0.038	0.615 ± 0.018
	AODE	Random	0.790 ± 0.024	0.877 ± 0.010	0.600 ± 0.088	0.696 ± 0.034	0.697 ± 0.055	0.869 ± 0.047	-	0.595 ± 0.031	0.705 ± 0.006	0.800 ± 0.008	0.786 ± 0.012	0.604 ± 0.038	0.547 ± 0.040	0.813 ± 0.043	0.666 ± 0.013
	AODE	Heuristic	0.787 ± 0.024	0.883 ± 0.008	0.622 ± 0.079	0.690 ± 0.032	0.694 ± 0.060	0.869 ± 0.045	-	0.605 ± 0.031	0.710 ± 0.006	0.816 ± 0.008	0.792 ± 0.012	0.598 ± 0.037	0.592 ± 0.028	0.821 ± 0.047	0.662 ± 0.015
Bivariate	Naive Bayes	-	**0.819 ± 0.020**	0.842 ± 0.047	0.719 ± 0.064	0.739 ± 0.029	0.709 ± 0.050	**0.916 ± 0.033**	**0.629 ± 0.004**	0.603 ± 0.028	0.779 ± 0.005	0.804 ± 0.011	0.781 ± 0.067	0.605 ± 0.037	0.724 ± 0.019	**0.883 ± 0.034**	0.728 ± 0.010
Mixture	-	-	0.806 ± 0.023	**0.926 ± 0.006**	0.717 ± 0.076	**0.757 ± 0.028**	**0.761 ± 0.043**	0.897 ± 0.041	-	0.578 ± 0.039	**0.809 ± 0.006**	**0.875 ± 0.006**	**0.871 ± 0.012**	**0.781 ± 0.021**	**0.756 ± 0.014**	0.850 ± 0.047	**0.775 ± 0.011**

Table 4: Mean CPU time (in seconds) for the learning step of the algorithms

Classifier	Base estimator	Order	authorship	blocks	breast	ecoli	glass	iris	letter	libras	pendigits	satimage	segment	vehicle	vowel	wine	yeast
Pairwise	Naive Bayes	-	**0.059 ± 0.046**	**0.075 ± 0.050**	**0.028 ± 0.007**	**0.168 ± 0.064**	**0.104 ± 0.044**	0.015 ± 0.010	**7.017 ± 0.455**	**0.749 ± 0.880**	**0.606 ± 0.138**	**0.463 ± 0.229**	**0.131 ± 0.051**	**0.046 ± 0.011**	**0.162 ± 0.051**	**0.012 ± 0.006**	**0.345 ± 0.094**
	AODE	-	11.751 ± 0.114	1.690 ± 0.193	1.115 ± 0.217	1.287 ± 0.401	0.619 ± 0.029	0.045 ± 0.025	197.367 ± 12.099	367.251 ± 12.560	33.865 ± 3.009	35.545 ± 3.049	3.514 ± 0.060	0.846 ± 0.018	6.897 ± 1.305	0.485 ± 0.083	1.968 ± 0.153
Chain	Naive Bayes	Random	0.065 ± 0.043	0.172 ± 0.062	0.094 ± 0.044	0.267 ± 0.089	0.132 ± 0.049	0.018 ± 0.010	137.006 ± 17.990	1.969 ± 0.865	1.415 ± 0.092	0.658 ± 0.263	0.249 ± 0.059	0.057 ± 0.013	1.005 ± 0.167	0.016 ± 0.012	0.721 ± 0.244
	Naive Bayes	Heuristic	0.129 ± 0.055	0.582 ± 0.234	0.085 ± 0.016	0.355 ± 0.148	0.175 ± 0.073	0.036 ± 0.023	139.060 ± 18.137	2.035 ± 0.873	2.269 ± 0.102	1.392 ± 0.507	0.367 ± 0.053	0.113 ± 0.011	0.748 ± 0.242	0.033 ± 0.018	1.009 ± 0.364
	AODE	Random	12.683 ± 0.124	1.513 ± 0.051	2.290 ± 0.775	11.992 ± 1.378	4.621 ± 0.554	0.065 ± 0.033	-	1919.447 ± 217.831	100.268 ± 4.888	29.675 ± 8.249	8.762 ± 0.131	1.091 ± 0.022	51.765 ± 15.859	0.300 ± 0.044	53.278 ± 6.029
	AODE	Heuristic	12.681 ± 0.123	3.330 ± 0.956	2.361 ± 0.900	12.058 ± 1.390	4.541 ± 0.484	0.077 ± 0.034	-	1927.792 ± 224.092	126.804 ± 49.220	44.852 ± 10.922	8.858 ± 0.121	1.155 ± 0.028	49.541 ± 8.891	0.532 ± 0.122	54.124 ± 6.344
Bivariate	Naive Bayes	-	0.163 ± 0.187	0.307 ± 0.098	0.256 ± 0.085	2.940 ± 0.645	0.982 ± 0.281	**0.009 ± 0.001**	1503.584 ± 132.230	65.513 ± 82.753	16.596 ± 1.987	5.161 ± 2.974	1.679 ± 0.655	0.145 ± 0.036	3.811 ± 0.524	0.017 ± 0.013	11.320 ± 1.366
Mixture	-	-	38.585 ± 9.388	4622.922 ± 1868.437	2.236 ± 0.847	46.714 ± 20.263	5.786 ± 2.774	2.019 ± 0.743	-	80.461 ± 16.114	7036.924 ± 1385.302	3330.594 ± 980.872	496.295 ± 152.996	77.948 ± 29.562	73.896 ± 37.405	1.656 ± 0.612	960.088 ± 288.705

Table 5: Mean CPU time (in seconds) for the inference step of the algorithms

Classifier	Base estimator	Order	authorship	blocks	breast	ecoli	glass	iris	letter	libras	pendigits	satimage	segment	vehicle	vowel	wine	yeast
Pairwise	Naive Bayes	-	**0.007 ± 0.003**	**0.018 ± 0.008**	0.008 ± 0.003	0.016 ± 0.008	0.009 ± 0.006	0.005 ± 0.008	**1.707 ± 0.140**	0.071 ± 0.031	0.187 ± 0.039	0.067 ± 0.018	**0.016 ± 0.001**	**0.005 ± 0.001**	0.048 ± 0.013	**0.003 ± 0.001**	**0.034 ± 0.011**
	AODE	-	0.525 ± 0.005	0.236 ± 0.025	0.105 ± 0.019	0.125 ± 0.046	0.056 ± 0.002	0.007 ± 0.004	21.039 ± 0.851	11.788 ± 0.391	3.599 ± 0.313	2.589 ± 0.331	0.316 ± 0.002	0.063 ± 0.001	0.569 ± 0.127	0.034 ± 0.010	0.205 ± 0.009
Chain	Naive Bayes	Random	**0.007 ± 0.002**	0.032 ± 0.015	0.010 ± 0.004	**0.014 ± 0.004**	0.007 ± 0.003	0.004 ± 0.005	11.437 ± 2.109	0.129 ± 0.030	**0.155 ± 0.011**	0.078 ± 0.020	0.025 ± 0.011	**0.005 ± 0.002**	0.082 ± 0.018	**0.003 ± 0.004**	0.042 ± 0.015
	Naive Bayes	Heuristic	**0.007 ± 0.003**	0.020 ± 0.009	0.009 ± 0.002	0.018 ± 0.010	0.009 ± 0.005	0.003 ± 0.005	11.629 ± 2.020	0.130 ± 0.030	0.159 ± 0.012	**0.053 ± 0.023**	0.021 ± 0.003	**0.005 ± 0.001**	**0.045 ± 0.010**	**0.003 ± 0.004**	0.043 ± 0.018
	AODE	Random	0.560 ± 0.004	0.195 ± 0.003	0.124 ± 0.030	0.647 ± 0.090	0.275 ± 0.064	0.009 ± 0.004	-	54.326 ± 8.391	6.738 ± 0.336	1.686 ± 0.098	0.618 ± 0.005	0.075 ± 0.001	1.697 ± 0.085	0.020 ± 0.001	2.642 ± 0.317
	AODE	Heuristic	0.558 ± 0.004	0.274 ± 0.096	0.138 ± 0.051	0.648 ± 0.080	0.270 ± 0.056	0.008 ± 0.004	-	54.368 ± 8.094	7.026 ± 0.686	2.905 ± 0.894	0.616 ± 0.005	0.076 ± 0.001	1.795 ± 0.192	0.035 ± 0.013	2.674 ± 0.323
Bivariate	Naive Bayes	-	0.021 ± 0.005	0.078 ± 0.045	0.069 ± 0.009	0.132 ± 0.100	0.033 ± 0.013	**0.002 ± 0.000**	597.227 ± 52.041	5.623 ± 0.164	7.091 ± 0.527	0.429 ± 0.103	0.158 ± 0.058	0.009 ± 0.002	1.266 ± 0.053	0.004 ± 0.003	0.498 ± 0.138
Mixture	-	-	0.016 ± 0.008	0.429 ± 0.201	**0.003 ± 0.004**	0.017 ± 0.010	**0.006 ± 0.005**	0.003 ± 0.002	-	**0.048 ± 0.015**	0.716 ± 0.218	0.482 ± 0.239	0.085 ± 0.035	0.031 ± 0.020	0.075 ± 0.042	**0.003 ± 0.003**	0.095 ± 0.036

defined over the values in the domain of the class variable. We solve this issue by introducing a pairwise variable that codifies the preference relation between two class labels. The OBOP is solved using the pair order matrix built according to the preference information provided by the pairwise variables to obtain the prediction for an input instance.

The experimental evaluation shows that the bivariate and pairwise classifiers using naive Bayes and AODE as base estimators are comparable in accuracy to the existing Gaussian mixture algorithm. However, the classifier chain algorithms are not competitive with these algorithms. This situation may be due to the existing correlations between the target class variables and, therefore, they are not following the independent assumption among the predictive variables. Regarding the computational efficiency, the Gaussian mixture algorithm is much slower than the proposals in this paper. Moreover, the algorithms using naive Bayes as a base classifier give rise to fast classifiers in both learning and inference.

We propose two lines for future work. First, we plan to extend the algorithms proposed in this paper to cope with incomplete information in the target partial rankings, as it turns out to be important from a practical point of view. Second, we plan to use other base estimators that better approximate the predicted class probabilities, given that the aim is to use that information for the pair order matrix to solve the OBOP.

Acknowledgments

This work has been funded by MCIN/AEI/10.13039/501100011033 and "ESF Investing your future" through the projects PID2019–106758GB–C33 and FPU18/00181. It is also partially funded by the project SBPLY/21/180501/000148 funded by the Government of Castilla-La Mancha and "ERDF A way of making Europe".

References

J. A. Aledo, J. A. Gámez, and A. Rosete. Utopia in the solution of the bucket order problem. *Decision Support Systems*, 97:69–80, 2017.

J. A. Aledo, J. A. Gámez, and A. Rosete. Approaching rank aggregation problems by using evolution strategies: The case of the optimal bucket order problem. *European Journal of Operational Research*, 270:982–998, 2018.

J. A. Aledo, J. A. Gámez, and A. Rosete. A highly scalable algorithm for weak rankings aggregation. *Information Sciences*, 570:144–171, 2021.

J. C. Alfaro, J. A. Aledo, and J. A. Gámez. Learning decision trees for the partial label ranking problem. *International Journal of Intelligent Systems*, 36:890–918, 2021a.

J. C. Alfaro, J. A. Aledo, and J. A. Gámez. Mixture-based probabilistic graphical models for the partial label ranking problem. In *Proceedings of the 22nd International Conference on Intelligent Data Engineering and Automated Learning*, pages 277–288, 2021b.

J. C. Alfaro, J. A. Aledo, and J. A. Gámez. Ensemble learning for the partial label ranking problem. *Mathematical Methods in the Applied Sciences*, 2022. doi: 10.1002/mma.8489.

J. Arias and J. Cózar. exreport: Fast, reliable and elegant reproducible research, 2015. URL `https://cran.r-project.org/web/packages/exreport/index.html`.

J. Borda. *Memoire sur les elections au scrutin.* Histoire de l'Academie Royal des Sciences, 1770.

W. Cheng, J. Hühn, and E. Hüllermeier. Decision tree and instance-based learning for label ranking. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 161–168, 2009.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

D. Dua and C. Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

E. J. Emond and D. W. Mason. A new rank correlation coefficient with application to the consensus ranking problem. *Journal of Multi-Criteria Decision Analysis*, 11:17–28, 2002.

R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, and E. Vee. Comparing and aggregating rankings with ties. In *Proceedings of the Twenty-third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 47–58, 2004.

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

S. García and F. Herrera. An extension on "Statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008.

A. Gionis, H. Mannila, K. Puolamäki, and A. Ukkonen. Algorithms for discovering bucket orders from data. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 561–566, 2006.

S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6:65–70, 1979.

F. V. Jensen and T. D. Nielsen. *Bayesian networks and decision graphs.* Springer Science+Business Media, 2007.

J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. In *Proceedings of the 13th The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 254–269, 2009.

A. Ukkonen, K. Puolamäki, A. Gionis, and H. Mannila. A randomized approximation algorithm for computing bucket orders. *Information Processing Letters*, 109:356–359, 2009.

G. I. Webb, J. R. Boughton, and Z. Wang. Not so naive Bayes: Aggregating one-dependence estimators. *Machine Learning*, 58:5–24, 2005.