Mixture-Based Probabilistic Graphical Models for the Partial Label Ranking Problem

Juan C. Alfaro[1,3(✉)], Juan A. Aledo[2,3], and José A. Gámez[1,3]

[1] Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, 02071 Albacete, Spain
{JuanCarlos.Alfaro,Jose.Gamez}@uclm.es

[2] Departamento de Matemáticas, Universidad de Castilla-La Mancha, 02071 Albacete, Spain
JuanAngel.Aledo@uclm.es

[3] Laboratorio de Sistemas Inteligentes y Minería de Datos, Instituto de Investigación en Informática de Albacete, 02071 Albacete, Spain

Abstract. The *Label Ranking* problem consists in learning *preference models* from training datasets labeled with a *ranking* of 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 both, the training dataset and the prediction given as output allow *tied* labels (i.e., there is no particular preference among them), known as the *Partial Label Ranking* problem. In particular, we propose *probabilistic graphical models* to solve this problem. As far as we know, there is no probability distribution to model rankings with ties, so we transform the rankings into discrete variables to represent the precedence relations (*precedes*, *ties* and *succeeds*) among pair of class labels (*multinomial distribution*). In this proposal, we use a *Bayesian network* with *Naive Bayes* structure and a *hidden variable* as *root* to collect the interactions among the different variables (predictive and target). The inference works as follows. First, we obtain the *posterior-probability* for each pair of class labels, and then we input these probabilities to the *pair order matrix* used to solve the corresponding *rank aggregation problem*. The experimental evaluation shows that our proposals are competitive (in accuracy) with the state-of-the-art *Instance Based Partial Label Ranking* (*nearest neighbors paradigm*) and *Partial Label Ranking Trees* (*decision tree induction*) algorithms.

Keywords: Mixture-based models · Bayesian networks · Naive bayes · (Partial) Label ranking

1 Introduction

In recent years, the *non-standard supervised classification* problems have grown significantly. In particular, the *Label Ranking* (*LR*) problem [9] consists in learning *preference models* able to predict *rankings* (a.k.a. *total orders* or *permutations*) defined over a finite set of class labels. An important difference between

H. Yin et al. (Eds.): IDEAL 2021, LNCS 13113, pp. 277–288, 2021.
https://doi.org/10.1007/978-3-030-91608-4_28

this problem and other non-standard supervised classification problems (e.g., *ordinal classification*) is that the instances of the training dataset are labeled with rankings, and these rankings are used during model learning.

In this paper, we focus on the *Partial Label Ranking* problem [6]. In this problem, the rankings associated with the instances of the training dataset and the predictions given as output are *partial rankings* (a.k.a. *total orders with ties* or *bucket orders*), that is, *rankings* with (possibly) *tied* class labels.

Based on [22], we rely on the use of a *hybrid Bayesian network* [14] where the root is a *hidden discrete variable* to jointly model the probability distributions for the discrete (*multinomial*) and *continuous* (*Gaussian*) attributes, and for the rankings. Although permutations (complete rankings without ties) may be modeled with the *Mallows distribution* [20], as far as we know, there is no probability distribution for bucket orders (complete rankings with ties). Therefore, we transform the ranking global preferences into a set of pairwise local preferences (*precedes*, *ties* and *succeeds*). By doing that, these variables can be modeled by a multinomial distribution (discrete variables), and so they can be easily integrated in the hybrid Bayesian network. The prediction for these variables are used to solve the associated *rank aggregation problem* [12], so outputting a partial ranking for a given unlabeled instance.

The paper is structured as follows. In Sect. 2, we review some basic notions concerning rankings. In Sect. 3, we formally describe the proposed model. In Sect. 4, we extend this model to allow interactions between the (continuous) attributes by means of a *multivariate Gaussian distribution*. In Sect. 5, we set out the experimental evaluation conducted to assess the proposed models. Finally, in Sect. 6, we provide the conclusions and future research lines.

2 Rank Aggregation Problem

Given a set of *items* $[[n]] = \{1, \ldots, n\}$, a *ranking* π represents a precedence relation among them. In particular, rankings may be *without ties* (a.k.a. *total orders* or *permutations*) or *with ties* (a.k.a. *partial rankings*, *total orders with ties* or *bucket orders*) if there is no preference among some of the ranked items.

The *rank aggregation problem* (*RAP*) [1] consists in obtaining a *consensus order* from a set of rankings. In particular, the *Kemeny Ranking Problem* (*KRP*) [19] is probably the most well-known, whose goal is to obtain the *consensus permutation* (a.k.a. *central permutation*) that minimizes a particular distance measure (e.g., the *Kendall distance*) with respect to a set of permutations. The KRP is usually solved with the *Borda count* algorithm because of its good trade-off between accuracy and efficiency.

In addition to the KRP, another well-known RAP is the *Optimal Bucket Order Problem* (*OBOP*) [12]. The goal of the OBOP is to obtain the *bucket matrix* B (associated with a bucket order π) that minimizes the distance D

$$D(B, P) = \sum_{u,v \in [[n]]} |B(u, v) - P(u, v)|$$

where P is the *pair order matrix* associated with a set of bucket orders (see [12] for the details).

Although there are several heuristic methods to solve the OBOP [2–4,17], we use a particular instance of the *Bucket Pivot Algorithm with least indecision assumption* [2] named LIA_G^{MP2}, because of the good trade-off it achieves between accuracy and efficiency.

3 Hidden Naive Bayes

Given that the goal is to output a partial ranking for an unlabeled instance, we have to obtain the pair order matrix C for solving the corresponding RAP. In our proposal, we use a Bayesian network to get the entries for this matrix, which codifies the preferences of a class label c_u over c_v, with $u < v$. Since $P(u,v) = 1 - P(v,u)$, we model both entries with a variable $Z_{u,v}$, and also the probability that c_u is tied with c_v. Thus, for each pair of class labels, we create the discrete variable

$$Z_{u,v} = \begin{cases} z_1, \text{ if } c_u \succ c_v \\ z_2, \text{ if } c_u \sim c_v \\ z_3, \text{ if } c_u \prec c_v \end{cases}$$

The advantage of this approach is that we only manage discrete variables. However, the complexity of the model grows quadratically with the number of labels according with $n_L = (n \cdot (n-1))/2$.

We propose a Bayesian network with Naive Bayes structure and a hidden variable to capture the interactions between the predictive variables and the target variable, and so obtain the *a-posteriori* probabilities for an unlabeled instance.

3.1 Model Definition

Figure 1 shows the *Plateau* representation of the proposed model. Note that the (hybrid) Bayesian network contains the discrete and continuous predictive variables, the discrete target variables and the discrete hidden variable H. In particular:

- *Discrete predictive variables*, denoted by X_j, $j = 1, \ldots, n_J$ with $dom(X_j) = \{x_{j_1}, \ldots, x_{j_{r_j}}\}$. They are observed both in the learning and inference stages.
- *Continuous predictive variables*, denoted by Y_k, $k = 1, \ldots, n_K$. They are observed both in the learning and inference stages.
- *Target variables*, denoted by $Z_{u,v}$, $u = 1, \ldots, n-1$ and $v = u+1, \ldots, n$, with domain $dom(Z_{u,v}) = \{z_1, z_2, z_3\}$, being $z_1 = c_u \succ c_v$, $z_2 = c_u \sim c_v$ and $z_3 = c_u \prec c_v$. They are observed in the learning stage.
- *Hidden variable*, denoted by H with $dom(H) = \{h_1, \ldots, h_{r_H}\}$, where r_H is the number of mixtures. This variable is never observed.

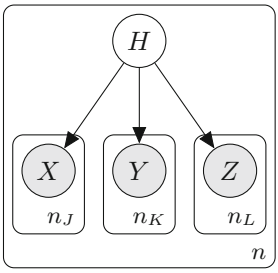

Fig. 1. Proposed HNB model

The discrete attributes, the target variables and the hidden variable follow a (conditional) multinomial distribution, and the continuous attributes follow a (conditional) Gaussian distribution. The joint probability distribution is given by

$$P(H, X_1, \ldots, X_{n_J}, Y_1, \ldots, Y_{n_K}, Z_{1,2}, \ldots, Z_{n-1,n}) = p(h_w) \cdot \prod_{j=1}^{n_J} P(X_j|H) \cdot \prod_{k=1}^{n_K} P(Y_k|H) \cdot \prod_{u=1,v=u+1}^{u=n-1,v=n} P(Z_{u,v}|H)$$

3.2 Parameter Estimation

We assume *complete* and *i.i.d.* data in both, the attributes and in the ranking variable. Therefore, we only deal with the hidden variable H, and we use the *Expectation-Maximization* (*EM*) to estimate jointly the parameters of both, the observed and hidden variables:

- **E step**: Under the assumption that the parameters of the discrete attributes $p(x_{j_i}^{h_w})$, continuous attributes $\mu_k^{h_w}, \sigma_k^{h_w}$, ranking variable $p(z_{u,v}^{h_w})$ and hidden variable $p(h_w)$, $j = 1, \ldots, n_J$, $i = 1, \ldots, r_j$, $k = 1, \ldots, n_K$, $l = 1, \ldots, n_L$, $u = 1, \ldots, n-1$, $v = u+1, \ldots, n$, $w = 1, \ldots, r_H$ are known, the probability of an instance $e_t = (x_{1,t}, \ldots, x_{n_J,t}, y_{1,t}, \ldots, y_{n_K,t}, \pi_t)$ being in a mixture is

$$P(h_w|x_{1,t}, \ldots, x_{n_J,t}, y_{1,t}, \ldots, y_{n_K,t}, \pi_t) = \frac{1}{C} \cdot p(h_w) \cdot \prod_{k=1}^{n_K} \frac{1}{\sigma_{k,t}^{h_w}\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{y_{k,t}-\mu_{k,t}^{h_w}}{\sigma}\right)^2} \cdot \prod_{j=1}^{n_J} p(x_{j,t}^{h_w}) \cdot \prod_{u=1,v=u+1}^{u=n-1,v=n} p(z_{u,v,t}^{h_w}) \quad (1)$$

  where C is a normalization constant.
- **M step**: Under the assumption that the probabilities of belonging to each mixture for all examples are known, the parameters of the model can be estimated as follows:
  - Multinomial parameters for the discrete attributes and the target variables. Each multinomial parameter is estimated by means of *maximum likelihood estimation* (*MLE*), where the count for each instance is weighted by the probability of $H = h_w$ given the instance.

- Gaussian parameters for the continuous attributes. The Gaussian parameters $\mu_k^{h_w}$ y $\sigma_k^{h_w}$ are estimated by means of MLE for each $H = h_w$, weighting each instance by the probability of it being in the mixture.

Stopping Condition: We use the *log-likelihood* of the model given the data with $\alpha = 0.001$ as convergence value. Moreover, we fix a maximum of $\beta = 100$ iterations.

3.3 Model Learning and Selection

We use the following procedure to compute the number of mixtures of the hidden variable H:

1. The dataset is divided in training Tr (0.8) and validation Tv (0.2), and the τ_X *rank correlation coefficient* is used to evaluate the models goodness in the search procedure.
2. The search for the number of mixtures is carried out greedily. First, we evaluate the model with $r_H = \{2^i\}_{i=1}^{10}$, and we select the best number of mixtures r'_H according with τ_X^{Tv}. Second, we apply a binary search in $[\frac{r'_H}{2}, r'_H]$, and we use the best number of mixtures r_H^* to train the model with the whole dataset.

Each time a new value for r_H is tried, the process starts from scratch, that is, all the parameters of the components are initialized (probabilities and weights) using the *k-means clustering algorithm* [23] with $k = r_H$ and $\gamma = 10$ different centroid seeds. Then, the EM algorithm is executed.

3.4 Inference

In the inference process, the method needs to compute the partial ranking π_t of the values in $dom(C)$ associated with an unlabeled instance e_t. Although the standard approach would select the (partial) ranking that maximizes the *a-posteriori probability* given the instance e_t, the cardinality of the search space $n!/2 \cdot \ln 2^{n+1}$, is too high, so we need to use an approximate method:

1. We compute the a-posteriori probability $P(Z_{u,v}|e_t)$ for each target variable using the Bayesian network.
2. We use these probabilities to populate the pair order matrix P_t associated with the instance e_t

$$P_t(u,v) = P(Z_{u,v} = z_1|e_t) + \tfrac{1}{2} \cdot P(Z_{u,v} = z_2|e_t)$$

$$P_t(v,u) = P(Z_{u,v} = z_3|e_t) + \tfrac{1}{2} \cdot P(Z_{u,v} = z_2|e_t) = 1 - P_t(u,v)$$

with $u < v$ and $P_t(u,v) = 0.5$ if $u = v$, and we solve the OBOP using P_t to obtain the (partial) ranking π_t for the instance e_t.

4 Gaussian Mixture Semi Naive Bayes

In this section, we assume that all the attributes are continuous, and we allow interactions among them.

4.1 Definition

We propose a *Semi-Naive Bayes* (*SNB*) [8,15] structure to model the continuous attributes using a multivariate normal distribution, while the rest of interactions are still managed by the hidden variable H.

We assume two variants of the *Gaussian mixture model* (*GMM*) [21]: *full*, where each component has its own general covariance matrix, and *tied*, where all components share the same general covariance matrix.

4.2 Estimation

The differences of the GMSNB model with respect to the HNB model are:

- **E step**: Similarly to the HNB model, we use Eq. 1, but, instead of the product of the conditional Gaussian distributions, we use the probability density function of the multivariate normal distribution $\mathcal{MN}(\boldsymbol{y}_t|\boldsymbol{\mu}^{h_w}, \Sigma^{h_w})$, where $\boldsymbol{y}_t$ is the configuration of values for the continuous attributes in e_t, $\boldsymbol{\mu}^{h_w}$ is the vector of means and Σ^{h_w} is the covariance matrix.
- **M step**: In the same way that the parameters of the HNB model, the means and empirical covariances of the continuous attributes are weighted by $w_t^{h_w} = P(h_w|\boldsymbol{y}_t, \pi_t)$.

4.3 Learning and Inference

The learning and inference stages of the HNB model and the GMSNB model are the same, but we model the continuous attributes with the multivariate normal distribution.

5 Experimental Evaluation

In this section, we detail the datasets used, the algorithms tested, the methodology adopted and the results obtained in the evaluation of our proposal.

5.1 Datasets

Table 1 shows the main characteristics of the 15 (semi-synthetic) datasets used as *benchmark* for the PLR problem [6]. The columns #rankings y #buckets stand for the mean number of different (partial) rankings in the dataset and the mean number of buckets per ranking, respectively. The datasets (and their description) are provided at: https://www.openml.org/u/25829.

Table 1. Description of the datasets

Datasets	#Instances	#Attributes	#Labels	#Rankings	#Buckets
Authorship	841	70	4	47	3.063
Blocks	5472	10	5	116	2.337
Breast	109	9	6	62	3.925
Ecoli	336	7	8	179	4.140
Glass	214	9	6	105	4.089
Iris	150	4	3	7	2.380
Letter	20000	16	26	15014	7.033
Libras	360	90	15	356	6.889
Pendigits	10992	16	10	3327	3.397
Satimage	6435	36	6	504	3.356
Segment	2310	18	7	271	3.031
Vehicle	846	18	4	47	3.117
Vowel	528	10	11	504	5.739
Wine	178	13	3	11	2.680
Yeast	1484	8	10	1006	5.929

5.2 Algorithms

We tested the following algorithms:

- *Instance Based Partial Label Ranking* (*IBPLR*) [6]. The Euclidean distance was used to identify the k nearest neighbors, and the (partial) rankings associated with these neighbors were weighted according to the (inverse) distance. The number of nearest neighbors was adjusted with a five-fold cross validation (5-cv) over the training dataset (see [6] for details).
- *Partial Label Ranking Trees* (*PLRT*) using the four criteria described in [6].
- *HNB-PLR* (Sect. 3). We considered four alternatives: Gaussian distribution (HNB-PLR-G) for the continuous attributes and multinomial distribution for the *equal-width* (HNB-PLR-W), *equal-frequency* (HNB-PLR-F) and *entropy-based* [13] (HNB-PLR-E) discretized versions. The number of bins for the equal-width and equal-frequency binning was fixed to 5.
- *GMSNB-PLR* (Sect. 4). We used a different covariance matrix for each mixture (full, GMSNB-PLR-F) and the same covariance matrix for all the mixtures (tied, GMSNB-PLR-T).

5.3 Methodology

We decided to apply the following design decisions:

- A 5×10 cross validation method was used (standard in the PLR problem).
- The accuracy was measured with the τ_X rank correlation coefficient [11].

– We used the standard statistical analysis procedure [10,16] by using the tool `exreport` [7] to analyze the results:
  1. First, a *Friedman test* is carried out with a significance level of $\alpha = 0.05$. If the obtained p-value is less than or equal to $\alpha = 0.05$, we reject the null hypothesis H_0, and so at least one algorithm is not equal to the rest.
  2. Second, a post-hoc test using the *Holm procedure* [18] is applied to discover the outstanding methods. This method compares all the algorithms with respect to the one ranked first by the Friedman test (control algorithm).

5.4 Reproducibility

The source code is provided at: https://github.com/alfaro96/scikit-lr. The experiments were executed in computers running the CentOS Linux 7 operating system, with CPU Intel(R) Xeon(R) E5–2630 a 2.40 GHz, and 16 GB of RAM memory.

5.5 Results

In this section, we provide and analyze the accuracy and CPU time results.

Accuracy. Table 2 shows the accuracy of the mixture-based models. Each cell contains the average and standard deviation over the test datasets of the 5×10 cv for the τ_X rank correlation coefficient between the real and predicted (partial) rankings. The boldfaced values correspond to the algorithms leading to the best accuracy for each dataset.

Table 2. Accuracy for the HNB-PLR and GMSNB-PLR algorithms

Dataset	HNB-PLR-G	HNB-PLR-F	HNB-PLR-W	HNB-PLR-E	GMSNB-PLR-F	GMSNB-PLR-T
Authorship	**0.814 ± 0.020**	0.797 ± 0.023	0.793 ± 0.023	0.797 ± 0.027	0.724 ± 0.022	0.806 ± 0.023
Blocks	0.931 ± 0.005	0.926 ± 0.005	0.899 ± 0.008	**0.942 ± 0.005**	0.922 ± 0.007	0.926 ± 0.006
Breast	0.736 ± 0.057	**0.760 ± 0.055**	0.648 ± 0.088	0.729 ± 0.058	0.641 ± 0.109	0.717 ± 0.076
Ecoli	**0.758 ± 0.035**	0.728 ± 0.033	0.727 ± 0.032	0.740 ± 0.034	0.714 ± 0.035	0.757 ± 0.028
Glass	0.692 ± 0.061	0.757 ± 0.045	0.707 ± 0.055	0.759 ± 0.049	0.662 ± 0.062	**0.761 ± 0.043**
Iris	0.874 ± 0.058	0.860 ± 0.076	0.887 ± 0.043	0.802 ± 0.105	0.871 ± 0.046	**0.897 ± 0.041**
Letter Libras	0.579 ± 0.031	0.545 ± 0.027	0.558 ± 0.034	**0.613 ± 0.034**	0.289 ± 0.030	0.578 ± 0.039
Pendigits	0.807 ± 0.005	0.804 ± 0.006	0.806 ± 0.005	0.805 ± 0.005	0.793 ± 0.007	**0.809 ± 0.006**
Satimage	0.870 ± 0.006	0.857 ± 0.007	0.843 ± 0.007	0.857 ± 0.007	0.813 ± 0.009	**0.875 ± 0.006**
Segment	0.866 ± 0.013	0.867 ± 0.009	0.870 ± 0.011	**0.883 ± 0.009**	0.846 ± 0.013	0.871 ± 0.012
Vehicle	0.731 ± 0.030	0.727 ± 0.028	0.709 ± 0.029	0.697 ± 0.028	0.606 ± 0.043	**0.781 ± 0.021**
Vowel	0.707 ± 0.032	0.728 ± 0.021	0.725 ± 0.024	0.562 ± 0.063	0.596 ± 0.027	**0.756 ± 0.014**
Wine	0.835 ± 0.042	0.822 ± 0.051	0.821 ± 0.055	0.826 ± 0.047	0.824 ± 0.055	**0.850 ± 0.047**
Yeast	0.747 ± 0.017	0.740 ± 0.014	0.715 ± 0.014	0.707 ± 0.027	0.731 ± 0.017	**0.775 ± 0.011**

We used the standard statistical analysis procedure described in Sect. 5.3:

1. The p-value obtained in the Friedman test was $4.124e^{-6}$, so we rejected the null hypothesis (H_0), and, at least, one algorithm was different.
2. Table 3 shows the results of the post-hoc test, taking as control the GMSNB-PLR-T algorithm. The columns *ranking* and p-valor represent the ranking obtained by the Friedman test and the p-value adjusted by the Holm procedure, respectively. The columns *win*, *tie*, *loss* contain the number of times that the control algorithm win, tie and losses with respect to the row-wise one. The boldfaced p-values are non-rejected null hypotheses (H_0).

Table 3. Results of the post-hoc test for the mean accuracy of the HNB-PLR and GMSNB-PLR algorithms

Method	Ranking	p-value	Win	Tie	Loss
GMSNB-PLR-T	1.90	–	–	–	–
HNB-PLR-G	2.47	$\mathbf{4.068e^{-1}}$	9	0	6
HNB-PLR-E	3.27	$\mathbf{9.087e^{-2}}$	10	0	5
HNB-PLR-F	3.70	$2.525e^{-2}$	13	1	1
HNB-PLR-W	4.40	$1.010e^{-3}$	15	0	0
GMSNB-PLR-F	5.27	$4.148e^{-6}$	14	0	1

In the statistical analysis, we can observe that the GMSNB-PLR-T algorithm is ranked first by the Friedman test, and it is statistically different from HNB-PLR-F, HNB-PLR-W and GMSNB-PLR-F algorithms. However, there is no statistical difference with respect to the HNB-PLR-G and HNB-PLR-E algorithms.

Let us compare the best algorithms with respect to the IBPLR and PLRT algorithms. Table 4 shows the results of this comparison.

Table 4. Mean accuracy for the IBPLR and PLRT algorithms

Conjunto de datos	IBPLR	PLRT-A	PLRT-D	PLRT-E	PLRT-G
Authorship	**0.829 ± 0.018**	0.757 ± 0.025	0.763 ± 0.019	0.780 ± 0.023	0.776 ± 0.023
Blocks	0.937 ± 0.005	0.940 ± 0.004	0.941 ± 0.005	0.944 ± 0.004	**0.946 ± 0.004**
Breast	0.751 ± 0.058	0.770 ± 0.075	**0.777 ± 0.067**	0.766 ± 0.058	0.763 ± 0.064
Ecoli	0.759 ± 0.027	0.758 ± 0.027	**0.765 ± 0.026**	0.763 ± 0.033	0.755 ± 0.031
Glass	0.756 ± 0.045	**0.764 ± 0.048**	0.761 ± 0.048	0.761 ± 0.037	0.758 ± 0.034
Iris	0.900 ± 0.051	0.912 ± 0.046	0.909 ± 0.044	**0.916 ± 0.045**	0.905 ± 0.040
Letter	**0.689 ± 0.005**	0.667 ± 0.004	0.667 ± 0.004	0.669 ± 0.005	0.670 ± 0.005
Libras	**0.648 ± 0.029**	0.584 ± 0.029	0.588 ± 0.030	0.575 ± 0.026	0.583 ± 0.029
Pendigits	**0.819 ± 0.006**	0.799 ± 0.006	0.801 ± 0.006	0.813 ± 0.006	0.811 ± 0.005
Satimage	**0.881 ± 0.005**	0.834 ± 0.006	0.839 ± 0.006	0.846 ± 0.006	0.848 ± 0.005
Segment	0.890 ± 0.008	0.886 ± 0.010	0.889 ± 0.009	0.894 ± 0.009	**0.896 ± 0.009**
Vehicle	0.739 ± 0.020	0.747 ± 0.031	0.757 ± 0.027	**0.793 ± 0.021**	0.778 ± 0.021
Vowel	**0.745 ± 0.017**	0.673 ± 0.031	0.680 ± 0.028	0.679 ± 0.023	0.682 ± 0.023
Wine	**0.845 ± 0.036**	0.841 ± 0.050	0.837 ± 0.046	0.824 ± 0.046	0.825 ± 0.060
Yeast	**0.790 ± 0.010**	0.769 ± 0.010	0.774 ± 0.009	0.775 ± 0.010	0.774 ± 0.009

For these set of algorithms, the p-value obtained by the Friedman test was $1.5e^{-2}$, so we rejected the null hypothesis H_0 and at least one algorithm is different to the rest. Table 5 shows the results of the post-hoc test adjusted with the Holm procedure, taking as control the IBPLR algorithm (ranked first by the Friedman test).

Table 5. Results of the post-hoc test for the mean accuracy of the compared algorithms

Method	Ranking	p-value	Win	Tie	Loss
IBPLR	3.07	–	–	–	–
PLRT-E	3.70	$\mathbf{5.763e^{-1}}$	8	0	7
PLRT-D	4.13	$\mathbf{5.763e^{-1}}$	9	0	6
PLRT-G	4.23	$\mathbf{5.763e^{-1}}$	9	0	6
PLRT-A	4.40	$\mathbf{5.442e^{-1}}$	9	0	6
GMSNB-PLR-T	4.63	$\mathbf{3.992e^{-1}}$	11	0	4
HNB-PLR-G	5.77	$1.523e^{-2}$	15	0	0
HNB-PLR-E	6.07	$5.574e^{-1}$	13	0	2

According with these results, we can conclude that:

- The IBPLR algorithm is ranked first by the Friedman test, without statistical difference with respect to the PLRT and GMSNB-PLR-T algorithms. These results show that the model-based methods are competitive with respect to the instance-based in the PLR problem, which is not the case for the LR problem [5,9].
- The main disadvantage of the HNB-PLR and GMSNB-PLR algorithms is the memory requirements, as they are not able to deal with datasets generating a high number of target variables. For instance, there are no results for the `letter` dataset (325 target variables and 20000 instances).

Time. Our proposals are slower than the instance-based (IBPLR) and tree-based methods (PLRT) because of the high number of mixtures (see Table 6) and so EM iterations required to properly model the joint probability distribution. Furthermore, since we have a high number of target variables (due to the ranking transformation), the EM algorithm takes too much time to converge. For instance, taking the `pendigits` dataset (10992 instances and 10 class labels, that is, 45 target variables), the GMSNB-PLR-T is 120 times slower than the IBPLR algorithm and 850 slower than the PLRT-G.

Table 6. Mean number of mixtures for each PGM

Dataset	HNB-PLR-G	HNB-PLR-F	HNB-PLR-W	HNB-PLR-E	GMSNB-PLR-F	GMSNB-PLR-T
Authorship	36.340 ± 36.988	24.58 ± 18.377	31.380 ± 21.813	40.840 ± 52.281	3.020 ± 0.141	35.420 ± 41.952
Blocks	167.440 ± 76.169	238.400 ± 168.220	81.300 ± 33.121	341.660 ± 147.979	68.520 ± 23.693	213.200 ± 97.692
Breast	15.960 ± 7.284	29.120 ± 19.157	19.800 ± 19.078	17.720 ± 12.795	4.520 ± 2.288	20.320 ± 8.110
Ecoli	31.000 ± 14.321	25.820 ± 21.930	31.140 ± 26.869	39.900 ± 20.928	12.920 ± 6.552	47.040 ± 27.871
Glass	17.220 ± 9.951	66.020 ± 32.922	27.920 ± 23.357	45.520 ± 23.603	5.180 ± 2.164	39.760 ± 16.577
Iris	17.020 ± 15.946	34.820 ± 20.457	24.180 ± 17.253	9.560 ± 4.739	7.960 ± 4.000	32.320 ± 22.709
Letter						
Libras	47.660 ± 12.967	40.940 ± 14.621	43.960 ± 13.425	121.760 ± 38.154	218.080 ± 39.608	56.200 ± 10.900
Pendigits	388.800 ± 108.298	204.680 ± 67.601	266.480 ± 95.088	261.520 ± 98.865	93.800 ± 27.355	405.840 ± 102.542
Satimage	326.660 ± 101.758	283.660 ± 96.820	198.720 ± 108.040	272.060 ± 108.377	29.620 ± 14.246	392.460 ± 110.909
Segment	140.400 ± 56.351	202.580 ± 158.267	196.300 ± 154.706	230.320 ± 141.072	42.680 ± 21.920	337.380 ± 121.548
Vehicle	73.320 ± 35.361	292.600 ± 151.414	346.300 ± 144.204	172.420 ± 126.264	12.260 ± 2.448	66.480 ± 60.716
Vowel	75.320 ± 26.250	169.260 ± 55.564	186.260 ± 49.278	95.640 ± 42.740	7.640 ± 2.884	174.580 ± 52.597
Wine	6.700 ± 9.033	11.980 ± 15.213	17.120 ± 19.256	24.960 ± 23.206	3.800 ± 1.030	14.480 ± 17.117
Yeast	103.500 ± 56.536	46.000 ± 18.553	127.660 ± 97.812	159.040 ± 113.482	30.300 ± 16.656	219.880 ± 93.535

6 Conclusions and Future Work

In this paper, we have proposed an algorithm based on Bayesian networks and rank aggregation to solve the PLR problem. In particular, we have transformed the ranking variable into several target variables to model the preferences among pair of class labels with a multinomial distribution. Our proposal is based on a SNB structure with a hidden variable as root to model the interaction between the predictive and target variables. Thus, we only need to estimate the parameters with the EM algorithm. Given an unlabeled instance, the a-posteriori probabilities for the target variables are computed and input to the pair order matrix used to solve the OBOP, and so obtain the output (partial) ranking.

From the experimental evaluation, we have concluded that the GMSNB-PLR-T algorithm is competitive with the IBPLR and PLRT algorithms. Note that, although the GMSNB-PLR-T requires more computational resources during the learning phase than the IBPLR algorithm, this is not the case during the inference phase.

As future research, we plan to reduce the problem using clustering techniques to solve the memory problems, which we expect that also reduces the CPU time required in the learning phase.

Acknowledgement. This work has been funded by the Government of Castilla-La Mancha and "ERDF A way of making Europe" through the project SBPLY/17/180501/000493. It is also part of the projects PID2019–106758GB–C33 and FPU18/00181 funded by MCIN/AEI/ 10.13039/501100011033 and "ESF Investing your future".

References

1. Aledo, J.A., Gámez, J.A., Molina, D.: Approaching the rank aggregation problem by local search-based metaheuristics. J. Comput. Appl. Math. **354**, 445–456 (2019)
2. Aledo, J.A., Gámez, J.A., Rosete, A.: Utopia in the solution of the bucket order problem. Decis. Support Syst. **97**, 69–80 (2017)

3. Aledo, J.A., Gámez, J.A., Rosete, A.: Approaching rank aggregation problems by using evolution strategies: the case of the optimal bucket order problem. Eur. J. Oper. Res. **270**, 982–998 (2018)
4. Aledo, J.A., Gámez, J.A., Rosete, A.: A highly scalable algorithm for weak rankings aggregation. Inf. Sci. **570**, 144–171 (2021)
5. Aledo, J.A., Gámez, J.A., Molina, D.: Tackling the supervised label ranking problem by bagging weak learners. Inf. Fusion **35**, 38–50 (2017)
6. Alfaro, J.C., Aledo, J.A., Gámez, J.A.: Learning decision trees for the partial label ranking problem. Int. J. Intell. Syst. **36**, 890–918 (2021)
7. Arias, J., Cózar, J.: ExReport: Fast, reliable and elegant reproducible research. http://jacintoarias.github.io/exreport 27 Oct 2021 (2016)
8. Bielza, C., Larrañaga, P.: Discrete bayesian network classifiers: a survey. ACM Comput. Surv. **47**, 1–43 (2014)
9. Cheng, W., Hühn, J., Hüllermeier, E.: Decision Tree and Instance-Based Learning for Label Ranking. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 161–168 (2009)
10. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
11. Emond, E.J., Mason, D.W.: A new rank correlation coefficient with application to the consensus ranking problem. J. Multi-Criteria Decis. Anal. **11**, 17–28 (2002)
12. Fagin, R., Kumar, R., Mahdian, M., Sivakumar, D., Vee, E.: Comparing and aggregating rankings with ties. In: Proceedings of the Twenty-third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 47–58 (2004)
13. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. Artif. Intell. **13**, 1022–1027 (1993)
14. Fernández, A., Gámez, J.A., Rumí, R., Salmerón, A.: Data clustering using hidden variables in hybrid Bayesian networks. Prog. Artif. Intell. **2**, 141–152 (2014)
15. Flores, M.J., Gámez, J. A., Martínez, A.: Supervised classification with bayesian networks: a review on models and applications. In: Intelligent Data Analysis for Real-Life Applications: Theory and Practice, pp. 72–102. IGI Global (2012)
16. García, S., Herrera, F.: An extension on "Statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. J. Mach. Learn. Res. **9**, 2677–2694 (2008)
17. Gionis, A., Mannila, H., Puolamäki, K., Ukkonen, A.: Algorithms for discovering bucket orders from data. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 561–566 (2006)
18. Holm, S.: A simple sequentially rejective multiple test procedure. Scand. J. Stat. **6**, 65–70 (1979)
19. Kemeny, J., Snell, J.: Mathematical Models in the Social Sciences. The MIT Press, Cambridge (1972)
20. Mallows, C.L.: Non-null ranking models. Biometrika **44**, 114–130 (1957)
21. Reynolds, D.: Gaussian Mixture Models. In: Li, S.Z., Jain, A. (eds.) Encyclopedia of Biometrics. Springer, Boston (2009) https://doi.org/10.1007/978-0-387-73003-5_196
22. Rodrigo, E.G., Alfaro, J.C., Aledo, J.A., Gámez, J.A.: Mixture-based probabilistic graphical models for the label ranking problem. Entropy **23**, 420 (2021)
23. Wu, X., Kumar, V.: The Top Ten Algorithms in Data Mining. Chapman and Hall, London (2009)