Mining Association Rules for Label Ranking

Cláudio Rebelo de Sá[1], Carlos Soares[1,2], Alípio Mário Jorge[1,3], Paulo Azevedo[5], and Joaquim Costa[4]

[1] LIAAD-INESC Porto L.A., Rua de Ceuta 118-6, 4050-190, Porto, Portugal
[2] Faculdade de Economia, Universidade do Porto
[3] DCC - Faculdade de Ciencias, Universidade do Porto
[4] DM - Faculdade de Ciencias, Universidade do Porto
[5] CCTC, Departamento de Informática, Universidade do Minho
claudio@liaad.up.pt, csoares@fep.up.pt, amjorge@fc.up.pt, pja@uminho.pt, jpcosta@fc.up.pt

Abstract. Recently, a number of learning algorithms have been adapted for label ranking, including instance-based and tree-based methods. In this paper, we propose an adaptation of association rules for label ranking. The adaptation, which is illustrated in this work with APRIORI Algorithm, essentially consists of using variations of the support and confidence measures based on ranking similarity functions that are suitable for label ranking. We also adapt the method to make a prediction from the possibly conflicting consequents of the rules that apply to an example. Despite having made our adaptation from a very simple variant of association rules for classification, the results clearly show that the method is making valid predictions. Additionally, they show that it competes well with state-of-the-art label ranking algorithms.

1 Introduction

Label ranking is an increasingly popular topic in the machine learning literature [12, 7, 25]. Label ranking studies the problem of learning a mapping from instances to rankings over a finite number of predefined labels. It can be considered as a variant of the conventional classification problem [7]. In contrast to a classification setting, where the objective is to assign examples to a specific class, in label ranking we are interested in assigning a complete preference order of the labels to every example.

There are two main approaches to the problem of label ranking. *Decomposition methods* decompose the problem into several simpler problems (e.g., multiple binary problems). *Direct methods* adapt existing algorithms or develop new ones to treat the rankings as target objects without any transformation. An example of the former is the ranking by pairwise comparisons [12]. Examples of algorithms that were adapted to deal with rankings as the target objects include decision trees [24, 7], k-Nearest Neighbor [5, 7] and the linear utility transformation [13, 9]. This second group of algorithms can be divided into two approaches. The first one contains methods (e.g., [7]) that are based on statistical distributions of rankings, such as Mallows [17]. The other group of methods are based on measures of similarity or correlation between rankings (e.g., [24, 2]).

In this paper, we propose an adaptation of association rules mining for label ranking based on similarity measures. Association rules mining is a very important and successful task in data mining. Although its original purpose was only descriptive, several adaptations have been proposed for predictive problems.

The paper is organized as follows: sections 2 and 3 introduce the label ranking problem and the task of association rule mining, respectively; section 4 describes the measures proposed here; section 5 presents the experimental setup and discusses the results; finally, section 6 concludes this paper.

2 Label Ranking

The formalization of the label ranking problem given here follows the one provided in [7].[6] In classification, given an instance x from the instance space $\mathbb{X}$, the goal is to predict the label (or class) λ to which x belongs, from a pre-defined set $\mathcal{L} = \{\lambda_1, \ldots, \lambda_k\}$. In label ranking the goal is to predict the ranking of the labels in $\mathcal{L}$ that are associated with x. We assume that the ranking is a total order over $\mathcal{L}$ defined on the permutation space Ω. A total order can be seen as a permutation π of the set $\{1, \ldots, k\}$, such that $\pi(a)$ is the position of λ_a in π. Let us also denote π^{-1} as the result of inverting the order in π. As in classification, we do not assume the existence of a deterministic $\mathbb{X} \to \Omega$ mapping. Instead, every instance is associated with a *probability distribution* over Ω. This means that, for each $x \in \mathbb{X}$, there exists a probability distribution $P(\cdot|x)$ such that, for every $\pi \in \Omega$, $P(\pi|x)$ is the probability that π is the ranking associated with x. The goal in label ranking is to learn the mapping $\mathbb{X} \to \Omega$. The training data is a set of instances $T = \{< x_i, \pi_i >\}, i = 1, \ldots, n$, where x_i are the independent variables describing instance i and π_i is the corresponding target ranking.

As an example, given a scenario where we have financial analysts making predictions about the evolution of volatile markets, it would be advantageous to be able to predict which analysts are more profitable in a certain market context [2]. Moreover, if we could have beforehand the full ordered list of the best analysts, this would certainly increase the chances of making good investments.

Given the ranking $\hat{\pi}$ predicted by a label ranking model for an instance x, which is, in fact, associated with the true label ranking π, we need to evaluate the accuracy of the prediction. For that, we need a loss function on Ω. One such function is the number of discordant label pairs,

$$D(\pi, \hat{\pi}) = \#\{(i,j)|\pi(i) > \pi(j) \wedge \hat{\pi}(i) < \hat{\pi}(j)\}$$

which, if normalized into the interval $[-1, 1]$, is equivalent to Kendall's τ coefficient. The latter is as a correlation measure where $D(\pi, \pi) = 1$ and $D(\pi, \pi^{-1}) = -1$. We obtain a loss function by averaging this function over a set of examples. We will use it as evaluation measure in this paper, as it has been used in recent studies [7]. However, other distance measures could have been used, like Spearman's rank correlation coefficient [22].

[6] An alternative formalization can be found in [25].

3 Association Rules Mining

An association rule (AR) is an implication: $A \rightarrow C$ where $A \bigcap C = \emptyset$, $A, C \subseteq desc(\mathbb{X})$ where $desc(\mathbb{X})$ is the set of descriptors of instances in $\mathbb{X}$, typically pairs $\langle attribute, value \rangle$. We also denote $desc(x_i)$ as the set of descriptors of instance x_i.

Association rules are typically characterized by two measures, support and confidence. The support of rule $A \rightarrow C$ in T is sup if $sup\%$ of the cases in it contain A and C. Additionally, it has a confidence $conf$ in T if $conf\%$ of cases in T that contain A also contain C.

The original method for induction of AR is the APRIORI algorithm that was proposed in 1994 [1]. APRIORI identifies all AR that have a support and confidence higher than a given minimal support threshold ($minsup$) and a minimal confidence threshold ($minconf$), respectively. Thus, the model generated is a set of AR of the form $A \rightarrow C$, where $A, C \subseteq desc(\mathbb{X})$, and $sup(A \rightarrow C) \geq minsup$ and $conf(A \rightarrow C) \geq minconf$. For a more detailed description see [1].

Despite the usefulness and simplicity of APRIORI, it runs a time consuming candidate generation process and needs space and memory that is proportional to the number of possible combinations in the database. Additionally it needs multiple scans of the database and typically generates a very large number of rules. Because of this, many new pruning methods were proposed in order to avoid that. Such as the hashing [19], dynamic itemset counting [6], parallel and distributed mining [20], relational database systems integrated with mining [23].

Association rules were originally proposed for descriptive purposes. However, they have been adapted for predictive tasks such as classification (e.g., [18]). Given that label ranking is a predictive task, we describe some useful notation from an adaptation of AR for classification in Section 3.2.

3.1 Pruning

AR algorithms typically generate a large number of rules (possibly tens of thousands), some of which represent only small variations from others. This is known as the rule explosion problem [4]. It is due to the fact that the algorithm might find rules for which the confidence can be marginally improved by adding further conditions to the antecedent.

Pruning methods are usually employed to reduce the amount of rules, without reducing the quality of the model. A common pruning method is based on the improvement that a refined rule yields in comparison to the original one [4]. The *improvement* of a rule is defined as the smallest difference between the confidence of a rule and the confidence of all sub-rules sharing the same consequent. More formally, for a rule $A \rightarrow C$

$$imp(A \rightarrow C) = min\left(\forall A' \subset A, conf\left(A \rightarrow C\right) - conf\left(A' \rightarrow C\right)\right)$$

As an example, if one defines $minImp = 0.1\%$, the rule $A_1 \rightarrow C$ will be kept, if, and only if $conf\left(A_1 \rightarrow C\right) - conf\left(A \rightarrow C\right) \geq 0.001$, where $A \subset A_1$.

3.2 Class Association Rules

Classification Association Rules (CAR), were proposed as part of the Classification Based on AR (CBA) algorithm [18]. A class association rule (CAR) is an implication of the form: $A \rightarrow \lambda$ where $A \subseteq desc(\mathbb{X})$, and $\lambda \in \mathcal{L}$, which is the class label. A rule $A \rightarrow \lambda$ holds in T with confidence $conf$ if $conf\%$ of cases in T that contain A are labeled with class λ, and with support sup in T if $sup\%$ of the cases in it contain A and are labeled with class λ.

CBA takes a tabular data set $T = \{\langle x_i, \lambda_i \rangle\}$, where x_i is a set of items and λ_i the corresponding class, and look for all frequent *ruleitems* of the form $\langle A, \lambda \rangle$, where A is a set of items and $\lambda \in \mathcal{L}$. The algorithm aims to choose a set of high accuracy rules $\mathcal{R}_\lambda$ to match T. R_λ matches an instance $< x_i, \lambda_i > \in T$ if there is at least one rule $A \rightarrow \lambda \in \mathcal{R}_\lambda$, with $A \subseteq desc(x_i), x_i \in \mathbb{X}$, and $\lambda \in \mathcal{L}$. If the rules cannot classify all examples, a default class is given to them (e.g., the majority class in the training data).

4 Association Rules for Label Ranking

We define a *Label Ranking Association Rule* (LRAR) as a straightforward adaptation of class association rules (CAR):

$$A \rightarrow \pi$$

where $A \subseteq desc(\mathbb{X})$ and $\pi \in \Omega$. The only difference is that the label $\lambda \in \mathcal{L}$ is replaced by the ranking of the labels, $\pi \in \Omega$. Similar to what the prediction made in CBA, when an example matches the rule $A \rightarrow \pi$, the predicted ranking is π. In this regard, we can use the same basic principle of the ruleitem for CARs in LRARs, which is $\langle A, \pi \rangle$ where A is a set of items and $\pi \in \Omega$.

This approach has two important problems. First, the number of classes can be extremely large, up to a maximum of $k!$, where k is the size of the set of labels, $\mathcal{L}$. This means that the amount of data required to learn a reasonable mapping $\mathbb{X} \rightarrow \Omega$ is too big.

The second disadvantage is that this approach does not take into account the differences in nature between label rankings and classes. In classification, two examples either have the same class or not. In this regard, label ranking is more similar to regression than to classification. This property can be used in the induction of prediction models. In regression, a large number of observations with a given target value, say 5.3, increases the probability of observing similar values, say 5.4 or 5.2, but not so much for very different values, say -3.1 or 100.2. A similar reasoning can be made in label ranking. Let us consider the case of a data set in which ranking $\pi_a = \{A, B, C, D, E\}$ occurs in 1% of the examples. Treating rankings as classes would mean that $P(\pi_a) = 0.01$. Let us further consider that the rankings $\pi_b = \{A, B, C, E, D\}, \pi_c = \{B, A, C, D, E\}$ and $\pi_d = \{A, C, B, D, E\}$ occur in 50% of the examples. Taking into account the stochastic nature of these rankings [7], $P(\pi_a) = 0.01$ seems to underestimate the probability of observing π_a. In other words it is expected that the observation of

π_b, π_c and π_d increases the probability of observing π_a and vice-versa, because they are similar to each other.

This affects even rankings which are not observed in the available data. For example, even though $\pi_e = \{A, B, D, C, E\}$ is not present in the data set it would not be entirely unexpected to see it in future data.

4.1 Similarity-based Support and Confidence

To take this characteristic into account, we can argue that the support of a ranking π increases with the observation of similar rankings and that the variation is proportional to the similarity. Given a measure of similarity between rankings $s(\pi_a, \pi_b)$, we can adapt the concept of support of the rule $A \rightarrow \pi$ as follows:

$$sup_{lr}(A \rightarrow \pi) = \frac{\sum_{i:A \subseteq desc(x_i)} s(\pi_i, \pi)}{n}$$

Essentially, what we are doing is assigning a weight to each target ranking in the training, π_i, data that represents its contribution to the probability that π may be observed. Some instances $x_i \in \mathbb{X}$ give full contribution to the support count (i.e., 1), while others may give partial or even a null contribution.

Any function that measures the similarity between two rankings or permutations can be used, such as Kendall's τ [16] or Spearman's ρ [22]. The function used here is of the form:

$$s(\pi_a, \pi_b) = \begin{cases} s'(\pi_a, \pi_b) & \text{if } s'(\pi_a, \pi_b) \geq \theta_{sup} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where s' is a similarity function. This general form assumes that below a given threshold, θ_{sup}, is not useful to discriminate between different similarity values, as they are so different from π_a. This means that, the support *sup* of $\langle A, \pi_a \rangle$ will have contributions from all the *ruleitems* of the form $\langle A, \pi_b \rangle$, for all π_b where $s'(\pi_a, \pi_b) > \theta_{sup}$). Again, many functions can be used as s'.

The confidence of a rule $A \rightarrow \pi$ is obtained simply by replacing the measure of support with the new one.

$$conf_{lr}\,(A \rightarrow \pi) = \frac{sup_{lr}\,(A \rightarrow \pi)}{sup\,(A \rightarrow \pi)}$$

Given that the loss function that we aim to minimize is known beforehand, it makes sense to use it to measure the similarity between rankings. Therefore, we use Kendall's τ. In this case, we think that $\theta_{sup} = 0$ would be a reasonable value, given that it separates the negative from the positive contributions. Table 1 shows an example of a label ranking dataset represented following this approach.

To present a more clear interpretation, the example given in table 1, the instance $(\{A1 = L, A2 = XL, A3 = S\})$ (TID=1) contributes to the support count of the ruleitem $\langle \{A1 = L, A2 = XL, A3 = S\}, \pi_3 \rangle$ with 1. The same instance,

Table 1. An example of a label ranking dataset to be processed by the APRIORI-LR algorithm.

				π_1	π_2	π_3
TID	A1	A2	A3	$(1,3,2)$	$(2,1,3)$	$(2,3,1)$
1	L	XL	S	0.33	0.00	1.00
2	XXL	XS	S	0.00	1.00	0.00
3	L	XL	XS	1.00	0.00	0.33

Algorithm 1 APRIORI-LR - APRIORI for Label Ranking

Require: *minsup* and *minconf*
C_k: Candidate ruleitems of size k
F_k: Frequent ruleitems of size k
$T = \{< x_i, \pi_i >\}$: Transactions in the database
$F_1 = \{\langle A, \pi\rangle : \#A = 1 \text{ AND } sup_{lr}(\langle A, \pi\rangle) \geq minsup\}$
$k = 1$
while $F_k \neq \varnothing$ **do**
  $C_{k+1} = \{cand = \langle A1 \cap A2, \pi\rangle : \langle A_1, \pi\rangle, \langle A_2, \pi\rangle \in F_k, \#(A_1 \cap A_2) = k+1\}$
  $F_{k+1} = \{c : c \in C_{k+1} \wedge sup_{lr}(c) \geq minsup\}$
  $k = k + 1$
end while
$F = \cup_{i=1}^{k} F_i$
$\mathcal{R}_\pi = \{A \rightarrow \pi : \langle A, \pi\rangle \in F \wedge conf_{lr}(A \rightarrow \pi) \geq minconf\}$
return $\mathcal{R}_\pi$

will also give a small contribution of 0.33 to the support count of the ruleitem $\langle\{A1 = L, A2 = XL, A3 = S\}, \pi_1\rangle$, given their similarity. On the other hand, no contribution to the count of the ruleitem's $\langle\{A1 = L, A2 = XL, A3 = S\}, \pi_2\rangle$ support is given, which are clearly different.

4.2 APRIORI-LR Algorithm

Using the definitions of support and confidence proposed, adaptation of any AR learning algorithm for label ranking is simple. However, for illustration purposes, we will present an adaptation of the APRIORI algorithm, called APRIORI-LR. Given a training set $T = \{< x_i, \pi_i >\}, i = 1, \ldots, n$, frequent ruleitems are generated with Algorithm 1 and transformed in LRARs.

Let $\mathcal{R}_\pi$ be the set of all the generated *label ranking association rules*. The algorithm aims to create a set of high accuracy rules $r_\pi \in \mathcal{R}_\pi$ to cover T. The classifier has the following format:

$$< r_{\pi_1}, r_{\pi_2}, \ldots, r_{\pi_n} >$$

However, if these are insufficient to rank the given examples, a $default\_ranking$ is used. The default ranking can be the average ranking [5], which is often used for this purpose.

This approach has two problems. The first is that it can only predict rankings which were present in the training set (except when no rules apply and the predicted ranking is the default ranking). The second problem is that it solves conflicts between rankings without taking into account the "continuous" nature of rankings, which was illustrated earlier. The problem of generating a single permutation from a set of conflicting rankings has been studied in the context of *consensus rankings*.

It has been shown in [15] that a ranking obtained by ordering the average ranks of the labels across all rankings minimizes the euclidean distance to all those rankings. In other words, it maximizes the similarity according to Spearman's ρ [22]. Given m rankings π_i $(i = 1, \ldots, m)$ we aggregate them by computing for each item j $(j = 1, \ldots, k)$:

$$\overline{r_j} = \frac{\sum_{i=1}^{m} \pi_{i,j}}{m}$$

The predicted ranking $\hat{\pi}$ is obtained by ranking the items according to the value of $\overline{r_j}$.

We can take advantage of this in the ranker builder in the following way: the final predicted label ranking is the consensus of all the label rankings in the consequent of the rules r_π triggered by the test example.

To implement pruning based on improvement for LR, some adaptation is required as well. Given that the relation between target values is different from classification, as discussed in Section 4.1, we have to limit the comparison between rules with different consequents, if the similarity function $S'(\pi, \pi') \geq \theta_{imp}$.

4.3 Parameter tuning

Due to the intrinsic nature of each different dataset, or even of the pre-processing methods used to prepare the data (e.g., the discretization method), the maximum *minsup/minconf* needed to obtain a rule set $\mathcal{R}_\pi$ that matches all or at least most of the examples, may vary significantly. We used a greedy method to define the minimum confidence. As stated earlier, a rule set $\mathcal{R}_\pi$ matches an example if at least one rule $(A \rightarrow \lambda) \in \mathcal{R}_\pi$, with $A \subseteq desc(x_i), x_i \in \mathbb{X}$. Then, our goal is to obtain a rule set $\mathcal{R}_\pi$ that maximizes the number of examples that are matched, here defined as M. Additionally, we want the best rules, the rules with the highest confidence values.

The parameter tuning method (Algorithm 2) determines the $minconf$ that obtains the rule set according to those criteria. To set the step value we consider that, on one hand, a suitable $minconf$ must be found as soon as possible. On the other hand, this very same value should be as high as possible. Therefore, 5% seems a reasonable step value.

The ideal value for the *minsup*, is as close to 1% as possible. However, in some datasets, namely those with a larger number of attributes, frequent ruleitem generation can be a very time consuming task. In this case, *minsup* must be set

Table 2. Summary of the datasets

Datasets	type	#examples	#labels	#attributes
autorship	A	841	4	70
bodyfat	B	252	7	7
calhousing	B	20640	4	4
cpu-small	B	8192	5	6
elevators	B	16599	9	9
fried	B	40769	5	9
glass	A	214	6	9
housing	B	506	6	6
iris	A	150	3	4
pendigits	A	10992	10	16
segment	A	2310	7	18
stock	B	950	5	5
vehicle	A	846	4	18
vowel	A	528	11	10
wine	A	178	3	13
wisconsin	B	194	16	16

to a value larger than 1%. In this work, one such example is *authorship*, which has 70 attributes.

Algorithm 2 Parameter tuning Algorithm

$minconf = 100\%$
$minsup = 1$
while $M < 100\%$ **do**
  $minconf = minconf - 5\%$
  Run Algorithm 1 with ($minsup$,$minconf$) and determine M
end while
return minconf

This procedure has the important advantage that it does not take into account the accuracy of the rule sets generated, thus reducing the risk of overfitting.

5 Experimental Results

The data sets in this work were taken from KEBI Data Repository in the Philipps University of Marburg [7] (Table 2). Continuous variables were discretized with two distinct methods: (1) *recursive minimum entropy partitioning* criterion ([11]) with the *minimum description length* (MDL) as stopping rule, motivated by [10] and (2) *equal width* bins.

The evaluation measure is Kendall's τ and the performance of the method was estimated using ten-fold cross-validation. The performance of APRIORI-LR is compared with a baseline method, the default ranking (explained earlier) and RPC [14]. For the generation of frequent ruleitems we used CAREN [3]. The base learner used in RPC is the Logistic Regression Algorithm, with the default configurations of the function *Logit* from the Stats package of R Programing Language [21].

Table 3. Results obtained with *minimum entropy* discretization and with *equal width* discretization with 3 bins for each attribute

	Minimum entropy						Equal width (3 bins)					
	τ	$\tau_{baseline}$	minsup	minconf	#rules	M	τ	$\tau_{baseline}$	minsup	minconf	#rules	M
authorship	.608	.568	20	60	3717	100%	NA	-	-	-	-	-
bodyfat	.059	-.064	1	15	3289	98%	.161	-.064	1	25	16222	100%
calhousing	.291	.048	1	35	221	97%	.139	.048	1	20	889	100%
cpu-small	.439	.234	1	35	2774	100%	.279	.234	1	30	1559	100%
elevators	.643	.289	1	60	1864	98%	.623	.289	1	60	18160	100%
fried	.774	-.005	1	35	1959	97%	.676	-.005	1	35	14493	100%
glass	.871	.684	1	85	485	99%	.794	.684	1	75	11385	100%
housing	.758	.058	1	60	2547	96%	.577	.058	1	45	5027	100%
iris	.960	.089	1	90	115	100%	.883	.089	1	80	69	100%
pendigits	NA	-	-	-	-	-	.684	.451	10	75	18590	90%
segment	.829	.372	4	85	4949	96%	.496	.372	35	75	4688	49%
stock	.890	.070	1	75	1606	100%	.836	.070	1	65	1168	100%
vehicle	.774	.179	7	80	10480	99%	.675	.179	15	80	6662	83%
vowel	.680	.195	1	70	21419	99%	.709	.195	1	70	143882	100%
wine	.844	.329	15	95	5960	100%	.910	.329	1	95	165263	100%
wisconsin	.031	-.031	1	0	1224	92%	.280	-.031	5	20	404773	100%

Additionally, we compare the performance of our algorithm with the results obtained with constraint classification (CC), instance-based label ranking (IBLR) and ranking trees (LRT), that were presented in [7]. We note that we did not run experiments with these methods and simply compared our results with the published results of the other methods. Thus, they were probably obtained with different partitions of the data and can not be compared directly. However, they provide some indication of the quality of our method, when compared to the state-of-the-art.

The value θ_{imp} was set to 0 in all experiments. This option may not be as intuitive as it is in θ_{sup}. However, since the focus of this work is the reduction of the number of generated rules, this value is suitable.

5.1 Results

Table 3 shows that the method obtains results with both discretization methods that are clearly better than the ones obtained by the baseline method. This means that the APRIORI-LR is identifying valid patterns that can predict label rankings.

Table 4 presents the results obtained with pruned rules using the same $minsup$ and $minconf$ values as in the previous experiments and compares it to RPC using as a base learner *Logistic Regression*. *Rd* represents the percentage of the number of rules reduced by pruning. The results presented clearly show that the $minImp$ constraint, set to 0.00 and 0.01, succeeded to reduce the number of rules. However, there was no improvement in accuracy, although it also did not decrease. Further tests are required to understand how this parameter affects the accuracy of the models.

Finally, table 5 compares APRIORI-LR with state of the art methods based on published results [7]. Given that the methods were not compared under the same conditions, this simply gives us a rough idea of the quality of the method proposed here. It indicates that, despite the simplicity of the adaptation, APRIORI-LR is a competitive method. We expect that the results can

Table 4. Comparison of APRIORI-LR with RPC

	Minimum entropy					Equal width (3 bins)				
	APRIORI-LR				RPC	APRIORI-LR				RPC
	τ	mImp=0	Rd(%)	mImp=1	Log. R.	τ	mImp=0	Rd(%)	mImp=1	Log. R.
authorship	0.608	0.634	-40	0.637	0.900	NA	NA	-	NA	0.905
bodyfat	0.059	0.057	-20	0.058	0.264	0.161	0.156	-98	0.156	0.175
calhousing	0.291	0.299	-54	0.300	0.227	0.139	0.112	-83	0.110	0.132
cpu-small	0.439	0.421	-91	0.418	0.446	0.279	0.271	-97	0.271	0.286
elevators	0.643	0.647	-93	0.651	0.650	0.623	0.620	-98	0.621	0.621
fried	0.774	0.731	-71	0.730	0.827	0.676	0.674	-93	0.676	0.671
glass	0.871	0.834	-83	0.833	0.898	0.794	0.767	-98	0.776	0.846
housing	0.758	0.753	-84	0.753	0.648	0.577	0.559	-96	0.562	0.552
iris	0.960	0.961	-75	0.961	0.862	0.883	0.876	-63	0.881	0.756
pendigits	NA	NA	NA	NA	NA	0.684	0.682	-96	0.685	0.879
segment	0.829	0.828	-89	0.828	0.935	0.496	0.496	-100	0.500	0.878
stock	0.890	0.875	-75	0.874	0.795	0.836	0.822	-88	0.822	0.675
vehicle	0.774	0.781	-91	0.775	0.841	0.675	0.675	-97	0.674	0.820
vowel	0.680	0.686	-91	0.685	0.670	0.709	0.718	-97	0.721	0.571
wine	0.844	0.871	-96	0.884	0.925	0.910	0.877	-99	0.875	0.892
wisconsin	0.031	0.030	-8	0.031	0.612	0.280	0.286	-99	0.293	0.478

Table 5. Comparison of APRIORI-LR with state-of-the-art methods.

	APRIORI-LR				
	EW	ME	CC	IBLR	LRT
authorship	NA	0.608	0.920	0.936	0.882
bodyfat	0.161	0.059	0.281	0.248	0.117
calhousing	0.139	0.291	0.250	0.351	0.324
cpu-small	0.279	0.439	0.475	0.506	0.447
elevators	0.623	0.643	0.768	0.733	0.760
fried	0.676	0.774	0.999	0.935	0.890
glass	0.794	0.871	0.846	0.865	0.883
housing	0.577	0.758	0.660	0.745	0.797
iris	0.883	0.960	0.836	0.966	0.947
pendigits	0.684	NA	0.903	0.944	0.935
segment	0.496	0.829	0.914	0.959	0.949
stock	0.836	0.890	0.737	0.927	0.895
vehicle	0.675	0.774	0.855	0.862	0.827
vowel	0.709	0.680	0.623	0.900	0.794
wine	0.910	0.844	0.933	0.949	0.882
wisconsin	0.280	0.031	0.629	0.506	0.343

be significantly improved, for instance, by implementing more complex pruning methods.

6 Conclusions

In his paper we present a simple adaptation of an association rules algorithm for label ranking. This adaptation essentially consists of 1) ensuring that rules have label rankings in their consequent, 2) using variations of the support and confidence measures that are suitable for label ranking and 3) generating the model with parameters selected by a simple greedy algorithm.

These results clearly show that this is a viable label ranking method. It outperforms a simple baseline and competes well with RPC, which means that, despite its simplicity, it is inducing useful patterns.

Additionally, the results obtained indicate that the choice of the discretization method and the number of bins per attribute play an important role in the accuracy of the models. The tests indicate that the supervised discretization

method (*minimum entropy*), gives better results than *equal width* partitioning. This is, however, not the main focus of this work.

Improvement-based pruning was successfully implemented and reduced the number of rules in a substantial number. This plays an important role in generating models with higher interpretability.

The new framework proposed in this work, based on distance functions, is consistent with the classical concepts underlying association rules. Furthermore, although it was developed in the context of the label ranking task, it can also be adapted for other tasks such as regression and classification. In fact, Classification Association Rules can be regarded as a special case of distance-based AR, where the distance function is 0-1 loss.

This work uncovered several possibilities that could be better studied in order to improve the algorithm's performance. They include: improving the prediction generation method; implementing better pruning methods; developing a discretization method that is suitable for label ranking; and the choice of parameters.

For evaluation, we have used a measure that is typically used in label ranking. However, it is important to give more importance to higher ranks than to lower ones which can be done, for instance, with the weighted rank correlation coefficient [8].

Additionally, it is essential to test the methods on real label ranking problems. The KEBI datasets are adapted from UCI classification problems. We plan to test our methods on other problems including algorithm selection and predicting the rankings of financial analysts [2]. In terms of real world applications, these can be adapted to rank analysts, based on their past performance and also radios, based on user's preferences.

Acknowledgments

This work was partially supported by project Rank! (PTDC/EIA/81178/2006) from FCT and Palco AdI project Palco3.0 financed by QREN and Fundo Europeu de Desenvolvimento Regional (FEDER). We thank the anonymous referees for useful comments.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: VLDB. pp. 487–499 (1994)
2. Aiguzhinov, A., Soares, C., Serra, A.P.: A similarity-based adaptation of naive bayes for label ranking: Application to the metalearning problem of algorithm recommendation. In: Discovery Science. pp. 16–26 (2010)
3. Azevedo, P.J., Jorge, A.M.: Ensembles of jittered association rule classifiers. Data Min. Knowl. Discov. 21(1), 91–129 (2010)
4. Bayardo, R., Agrawal, R., Gunopulos, D.: Constraint-based rule mining in large, dense databases. Data Mining and Knowledge Discovery 4(2), 217–240 (2000)

5. Brazdil, P., Soares, C., Costa, J.: Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results. Machine Learning 50(3), 251–277 (2003)
6. Brin, S., Motwani, R., Ullman, J.D., Tsur, S.: Dynamic itemset counting and implication rules for market basket data. Proceedings of the 1997 ACM SIGMOD international conference on Management of data - SIGMOD '97 pp. 255–264 (1997)
7. Cheng, W., Hühn, J., Hüllermeier, E.: Decision tree and instance-based learning for label ranking. In: ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning. pp. 161–168. ACM, New York, NY, USA (2009)
8. Pinto da Costa, J., Soares, C.: A weighted rank measure of correlation. Australian & New Zealand Journal of Statistics 47(4), 515–529 (2005)
9. Dekel, O., Manning, C.D., Singer, Y.: Log-linear models for label ranking. Advances in Neural Information Processing Systems (2003)
10. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Machine Learning - International Workshop Then Conference -. pp. 194–202 (1995)
11. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: IJCAI. pp. 1022–1029 (1993)
12. Fürnkranz, J., Hüllermeier, E.: Preference learning. KI 19(1), 60– (2005)
13. Har-Peled, S., Roth, D., Zimak, D.: Constraint classification: a new approach to multiclass classification. In: Proc. of the International Workshop on Algorithmic Learning Theory (ALT). pp. 135–150. Springer-Verlag (2002)
14. Hüllermeier, E., Fürnkranz, J., Cheng, W., Brinker, K.: Label ranking by learning pairwise preferences. Artif. Intell. 172(16-17), 1897–1916 (2008)
15. Kemeny, J., Snell, J.: Mathematical Models in the Social Sciences. MIT Press (1972)
16. Kendall, M., Gibbons, J.: Rank correlation methods. Griffin London (1970)
17. Lebanon, G., Lafferty, J.D.: Conditional Models on the Ranking Poset. In: NIPS. pp. 415–422 (2002)
18. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. Knowledge Discovery and Data Mining pp. 80–86 (1998)
19. Park, J.S., Chen, M.S., Yu, P.S.: An effective hash-based algorithm for mining association rules. ACM SIGMOD Record 24(2), 175–186 (May 1995)
20. Park, J.S., Chen, M.S., Yu, P.S.: Efficient parallel and data mining for association rules. In: CIKM. pp. 31–36 (1995)
21. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2010), http://www.R-project.org, ISBN 3-900051-07-0
22. Spearman, C.: The proof and measurement of association between two things. American Journal of Psychology 15, 72–101 (1904)
23. Thomas, S., Sarawagi, S.: Mining generalized association rules and sequential patterns using sql queries. In: KDD. pp. 344–348 (1998)
24. Todorovski, L., Blockeel, H., Džeroski, S.: Ranking with Predictive Clustering Trees. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) Proc. of the 13th European Conf. on Machine Learning. pp. 444–455. No. 2430 in LNAI, Springer-Verlag (2002)
25. Vembu, S., Gärtner, T.: Label Ranking Algorithms: A Survey. In: Johannes Fürnkranz, E.H. (ed.) Preference Learning. Springer–Verlag (2010)