Decision Tree and Instance-Based Learning for Label Ranking

Weiwei Cheng CHENG@MATHEMATIK.UNI-MARBURG.DE
Jens Hühn HUEHN@MATHEMATIK.UNI-MARBURG.DE
Eyke Hüllermeier EYKE@MATHEMATIK.UNI-MARBURG.DE
Mathematics and Computer Science, Marburg University, Hans-Meerwein-Str., 35032 Marburg, Germany

Abstract

The label ranking problem consists of learning a model that maps instances to total orders over a finite set of predefined labels. This paper introduces new methods for label ranking that complement and improve upon existing approaches. More specifically, we propose extensions of two methods that have been used extensively for classification and regression so far, namely instance-based learning and decision tree induction. The unifying element of the two methods is a procedure for locally estimating predictive probability models for label rankings.

1. Introduction

The problem to learn a mapping from instances to rankings over a finite set of predefined labels, called *label ranking*, is a natural extension of conventional classification where, instead of a ranking of all labels, only a single label is requested as a prediction. As a ranking is a special type of preference relation, label ranking is of particular interest for the emerging field of preference learning (Hüllermeier et al., 2008).

Existing methods for label ranking are typically extensions of algorithms for binary classification. Ranking by pairwise comparison is a natural extension of pairwise classification, in which binary preference models are learned for each pair of labels, and the predictions of these models are combined into a ranking of all labels (Hüllermeier et al., 2008). Two other approaches, constraint classification and log-linear models for label ranking, seek to learn a (linear) utility function for each individual label (Har-Peled et al., 2003; Dekel et al., 2004).

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

Even though these approaches have shown good performance in first experimental studies, the reduction of the complex label ranking problem to the simple problem of binary classification does not come for free. First, the representation of a "ranking-valued" mapping in terms of an aggregation (e.g. argsort) of an ensemble of simple mappings (e.g., real-valued utility functions) typically comes along with a strong bias. This is especially true for methods such as constraint classification, for which the transformation from ranking to classification problems strongly exploits linearity properties of the underlying utility functions, while being less critical for pairwise ranking, where more flexible base learners can in principle be used.

Second, a representation in terms of an ensemble of models is not always desired, mainly since single models are considered more comprehensible. This point is especially relevant for the pairwise approach, where the size of the model ensemble is quadratic in the number of class labels.

To overcome these problems, we propose extensions of two quite popular machine learning methods, namely decision tree induction (Breiman et al., 1984) and instance-based learning (Aha et al., 1991), to the label ranking setting. Both methods are based on local estimation principles and are known to have a rather weak bias. Besides, decision trees are often praised for their good interpretability. Instance-based learning (nearest neighbor estimation) is maybe not as much attractive from this point of view, though by revealing information about a query's nearest neighbors, it does still offer a natural explanation for its predictions.

The paper is organized as follows. Section 2 recalls the problem of label ranking in a more formal setting. In Section 3, we introduce a probability model that will be used for estimating (local) predictive models for rankings. Sections 4 and 5 are devoted, respectively, to the instance-based and decision tree method for label ranking. An experimental evaluation is presented in Section 6, and Section 7 concludes the paper.

2. Label Ranking

Label ranking can be seen as an extension of the conventional classification setting. Instead of associating every instance $\boldsymbol{x}$ from an instance space $\mathbb{X}$ with one among a finite set of class labels $\mathcal{Y} = \{y_1 \dots y_n\}$, we associate $\boldsymbol{x}$ with a total order of all class labels, that is, a complete, transitive, and asymmetric relation $\succ_{\boldsymbol{x}}$ on $\mathcal{Y}$, where $y_i \succ_{\boldsymbol{x}} y_j$ indicates that y_i precedes y_j. Since a ranking can be considered as a special type of preference relation, we shall also say that $y_i \succ_{\boldsymbol{x}} y_j$ indicates that y_i is *preferred* to y_j given the instance $\boldsymbol{x}$. As an illustration, suppose that instances are students (characterized by attributes such as sex, age, and major subjects in secondary school) and $\succ$ is a preference relation on a fixed set of study fields such as Math, CS, and Physics.

Formally, a total order $\succ_{\boldsymbol{x}}$ can be identified with a permutation $\pi_{\boldsymbol{x}}$ of the set $\{1 \dots n\}$. It is convenient to define $\pi_{\boldsymbol{x}}$ such that $\pi_{\boldsymbol{x}}(i) = \pi_{\boldsymbol{x}}(y_i)$ is the position of y_i in the order. This permutation encodes the (ground truth) order relation

$$y_{\pi_{\boldsymbol{x}}^{-1}(1)} \succ_{\boldsymbol{x}} y_{\pi_{\boldsymbol{x}}^{-1}(2)} \succ_{\boldsymbol{x}} \dots \succ_{\boldsymbol{x}} y_{\pi_{\boldsymbol{x}}^{-1}(n)} \ ,$$

where $\pi_{\boldsymbol{x}}^{-1}(j)$ is the index of the label put at position j. The class of permutations of $\{1 \dots n\}$ (the symmetric group of order n) is denoted by Ω. By abuse of terminology, though justified in light of the above one-to-one correspondence, we refer to elements $\pi \in \Omega$ as both permutations and rankings.

In analogy with the classification setting, 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 $\boldsymbol{x} \in \mathbb{X}$, there exists a probability distribution $\mathbf{P}(\cdot \,|\, \boldsymbol{x})$ such that, for every $\pi \in \Omega$, $\mathbf{P}(\pi \,|\, \boldsymbol{x})$ is the probability that $\pi_{\boldsymbol{x}} = \pi$.

The goal in label ranking is to learn a "label ranker" in the form of an $\mathbb{X} \to \Omega$ mapping. As training data, a label ranker uses a set of instances $\boldsymbol{x}_k$, $k = 1 \dots m$, together with information about the associated rankings $\pi_{\boldsymbol{x}_k}$. Ideally, complete rankings are given as training information. From a practical point of view, however, it is important to allow for incomplete information in the form of a ranking

$$y_{\pi_{\boldsymbol{x}}^{-1}(i_1)} \succ_{\boldsymbol{x}} y_{\pi_{\boldsymbol{x}}^{-1}(i_2)} \succ_{\boldsymbol{x}} \dots \succ_{\boldsymbol{x}} y_{\pi_{\boldsymbol{x}}^{-1}(i_k)} \ ,$$

where $\{i_1, i_2 \dots i_k\} \subset \{1 \dots n\}$ such that $1 \leq i_1 < i_2 < \dots < i_k \leq n$. For example, for an instance $\boldsymbol{x}$, it might be known that $y_2 \succ_{\boldsymbol{x}} y_1 \succ_{\boldsymbol{x}} y_5$, while no preference information is given about the labels y_3 or y_4. We denote by $\mathcal{Y}(\pi) \subseteq \mathcal{Y}$ the set of labels that are present in a possibly incomplete ranking π.

To evaluate the predictive performance of a label ranker, a suitable loss function on Ω is needed. In the statistical literature, several distance measures for rankings have been proposed. One commonly used measure is the number of discordant label pairs,

$$D(\pi, \sigma) \ = \ \#\{(i,j) \,|\, \pi(i) > \pi(j) \ \wedge \sigma(i) < \sigma(j)\} \ , \quad (1)$$

which is closely related to Kendall's tau coefficient. In fact, the latter is a normalization of (1) to the interval $[-1, +1]$ that can be interpreted as a correlation measure (it assumes the value 1 if $\sigma = \pi$ and the value -1 if σ is the reversal of π). We shall focus on Kendall's tau as a natural, intuitive, and easily interpretable measure (Mallows, 1957) throughout the paper, even though other distance measures could of course be used. A desirable property of any distance $D(\cdot)$ is its invariance toward a renumbering of the elements (renaming of labels). This property is equivalent to the *right invariance* of $D(\cdot)$, namely $D(\sigma\nu, \pi\nu) = D(\sigma, \pi)$ for all $\sigma, \pi, \nu \in \Omega$, where $\sigma\nu = \sigma \circ \nu$ denotes the permutation $i \mapsto \sigma(\nu(i))$. The distance (1) is right-invariant, and so are most other commonly used metrics on Ω.

In a sense, the label ranking problem is in-between the standard problems of classification and regression. Like in classification, the output space is discrete. However, like in regression, it is endowed with a nontrivial topological structure and, therefore, suggests using loss functions other than the simple 0/1 loss.

3. The Mallows Model

So far, no assumptions about the conditional probability measure $\mathbf{P}(\cdot \,|\, \boldsymbol{x})$ on Ω were made, despite its existence. To become more concrete, we resort to a popular and commonly used distance-based probability model introduced by Mallows (Mallows, 1957). It will provide the basis for estimating local models which, as will be seen, is a key problem in both the instance-based and decision tree method for label ranking.

The standard Mallows model is a two-parameter model that belongs to the exponential family:

$$\mathbf{P}(\sigma \,|\, \theta, \pi) = \frac{\exp(-\theta D(\pi, \sigma))}{\phi(\theta, \pi)} \quad (2)$$

The ranking $\pi \in \Omega$ is the location parameter (mode, center ranking) and $\theta \geq 0$ is a spread parameter. For right-invariant metrics, the normalization constant does not depend on π and, therefore, can be written as a function $\phi(\theta)$ of θ alone. This is due to

$$\begin{aligned} \phi(\theta, \pi) &= \sum_{\sigma \in \Omega} \exp(-\theta D(\sigma, \pi)) = \sum_{\sigma \in \Omega} \exp(-\theta D(\sigma\pi^{-1}, id)) \\ &= \sum_{\sigma' \in \Omega} \exp(-\theta D(\sigma', id)) = \phi(\theta) \ , \end{aligned}$$

where id is the identity ranking $i \mapsto i$. More specifically, it can be shown that the normalization constant is given by (Fligner & Verducci, 1986)

$$\phi(\theta) = \prod_{j=1}^{n} \frac{1 - \exp(-j\theta)}{1 - \exp(-\theta)} \ ,$$

and that the expected distance from the center is

$$\mathbf{E}\left[D(\sigma, \pi) \,|\, \theta, \pi\right] = \frac{n \exp(-\theta)}{1 - \exp(-\theta)} - \sum_{j=1}^{n} \frac{j \exp(-j\theta)}{1 - \exp(-j\theta)} \ .$$

Obviously, the Mallows model assigns the maximum probability to the center ranking π. The larger the distance $D(\sigma, \pi)$, the smaller the probability of σ becomes. The spread parameter θ determines how quickly the probability decreases, i.e., how peaked the distribution is around π. For $\theta = 0$, the uniform distribution is obtained, while for $\theta \to \infty$, the distribution converges to the one-point distribution that assigns probability 1 to π and 0 to all other rankings.

4. Instance-Based Label Ranking

Coming back to the label ranking problem and the idea of instance-based learning, i.e., local prediction based on the nearest neighbor estimation principle, consider a query instance $\boldsymbol{x} \in \mathbb{X}$ and let $\boldsymbol{x}_1 \ldots \boldsymbol{x}_k$ denote the nearest neighbors of $\boldsymbol{x}$ (according to an underlying distance measure on $\mathbb{X}$) in the training set, where $k \in \mathbb{N}$ is a fixed integer.

4.1. Learning from Complete Observations

First, we consider the case where the observed label rankings are complete. Thus, each neighbor $\boldsymbol{x}_i$, $i = 1 \ldots k$, is associated with a ranking $\sigma_i \in \Omega$. In analogy to the conventional settings of classification and regression, in which the nearest neighbor estimation principle has been applied for a long time, we assume that the probability distribution $\mathbf{P}(\cdot \,|\, \boldsymbol{x})$ on Ω is (at least approximately) *locally constant* around the query $\boldsymbol{x}$. By furthermore assuming independence of the observations, the probability to observe $\boldsymbol{\sigma} = \{\sigma_1 \ldots \sigma_k\}$ given the parameters (θ, π) becomes

$$\begin{aligned} \mathbf{P}(\boldsymbol{\sigma} \,|\, \theta, \pi) &= \prod_{i=1}^{k} \mathbf{P}(\sigma_i \,|\, \theta, \pi) = \prod_{i=1}^{k} \frac{\exp\left(-\theta D(\sigma_i, \pi)\right)}{\phi(\theta)} \\ &= \frac{\exp\left(-\theta \sum_{i=1}^{k} D(\sigma_i, \pi)\right)}{\left(\prod_{j=1}^{n} \frac{1-\exp(-j\theta)}{1-\exp(-\theta)}\right)^k} \ . \end{aligned} \tag{3}$$

The maximum likelihood estimation (MLE) of (θ, π) is then given by those parameters that maximize this probability. It is easily verified that the MLE of π is

$$\hat{\pi} \ = \ \arg\min_{\pi} \sum_{i=1}^{k} D(\sigma_i, \pi) \ , \tag{4}$$

i.e., the (generalized) median of the rankings $\sigma_1 \ldots \sigma_k$. Moreover, the MLE of θ is derived from the mean observed distance from $\hat{\pi}$, which is an estimation of the expected distance $\mathbf{E}\left[D(\sigma, \pi) | \theta, \pi\right]$:

$$\frac{1}{k} \sum_{i=1}^{k} D(\sigma_i, \hat{\pi}) = \frac{n \exp(-\theta)}{1 - \exp(-\theta)} - \sum_{j=1}^{n} \frac{j \exp(-j\theta)}{1 - \exp(-j\theta)}.$$

Since the right-hand side of this equation is monotone decreasing in θ, a standard line search quickly converges to the MLE (Fligner & Verducci, 1986).

4.2. Learning from Incomplete Observations

Now, consider the more general case of incomplete preference information, which means that a ranking σ_i does not necessarily contain all labels. The probability of σ_i is then given by

$$\mathbf{P}(E(\sigma_i)) \ = \ \sum_{\sigma \in E(\sigma_i)} \mathbf{P}(\sigma \,|\, \theta, \pi) \ ,$$

where $E(\sigma_i)$ denotes the set of all *consistent extensions* of σ_i: A permutation $\sigma \in \Omega$ is a consistent extension of σ if it ranks all labels in $\mathcal{Y}(\sigma_i)$ in the same order.

The probability of observing the neighbor rankings $\boldsymbol{\sigma} = (\sigma_1 \ldots \sigma_k)$ then becomes

$$\begin{aligned} \mathbf{P}(\boldsymbol{\sigma} \,|\, \theta, \pi) &= \prod_{i=1}^{k} \mathbf{P}(E(\sigma_i) \,|\, \theta, \pi) \\ &= \prod_{i=1}^{k} \sum_{\sigma \in E(\sigma_i)} \mathbf{P}(\sigma \,|\, \theta, \pi) \\ &= \frac{\prod_{i=1}^{k} \sum_{\sigma \in E(\sigma_i)} \exp\left(-\theta D(\sigma, \pi)\right)}{\left(\prod_{j=1}^{n} \frac{1-\exp(-j\theta)}{1-\exp(-\theta)}\right)^k} \ . \end{aligned} \tag{5}$$

Computing the MLE of (θ, π) by maximizing this probability now becomes more difficult. To solve this problem, we resort to the idea of the EM (Expectation-Maximization) algorithm (Dempster et al., 1977), viewing the missing labels in the neighbor rankings σ_i as hidden variables. More specifically, like other methods such as learning hidden Markov models or K-means clustering, we use an MM training procedure, which replaces the E-step by another maximization step and can be seen as an approximation of EM.

Our algorithm works as follows (see Alg. 1). Starting from an initial center ranking $\pi \in \Omega$, each incomplete

neighbor ranking σ_i is replaced by the most probable consistent extension, i.e., by the ranking $\sigma_i^* \in E(\sigma_i)$ whose probability is maximal given $\hat{\pi}$ as a center (first M-step). Having replaced all neighbor rankings by their most probable extensions, an MLE $(\hat{\theta}, \hat{\pi})$ can be derived as described for the case of complete information above (second M-step). The center ranking π is then replaced by $\hat{\pi}$, and the whole procedure is iterated until the center does not change any more; $\hat{\pi}$ is then output as a prediction. In the following, we discuss three sub-problems of the algorithm in more detail, namely (i) the problem to find most probable extensions in the first M-step, (ii) the solution of the median problem (4) in the second M-step, and (iii) the choice of an initial center ranking.

(i) Regardless of the spread θ, a most probable extension $\sigma_i^* \in E(\sigma_i)$ of an incomplete ranking σ_i, given π, is obviously a minimizer of $D(\pi, \cdot)$. Such a ranking can be found efficiently, as shown in the following proposition (proof omitted due to space restrictions).

Proposition 1: Let π be a ranking of $Y = \{1, 2 \dots n\}$, and let σ be a ranking of a subset $C \subseteq Y$ with $|C| = m \leq n$. The ranking σ^* of Y that minimizes $D(\pi, \cdot)$ can be found as follows. First, each $i \in Y \setminus C$ is optimally inserted in σ, i.e., it is inserted between the labels on position j and $j+1$ in σ, where $j \in \{0, 1 \dots m\}$ ($j = 0$ means before the first and $j = m$ after the last label), if j is a minimizer of

$$\begin{aligned}\#\{1 \leq k \leq m \,|\, \sigma(k) \leq j \wedge \pi(k) > \pi(i)\} \\ + \#\{1 \leq k \leq m \,|\, \sigma(k) > j \wedge \pi(k) < \pi(i)\}\,.\end{aligned}$$

In the case of a tie, the position with the smallest index is chosen. Then, those $i \in Y \setminus C$ that are inserted at the same position are put in the same order as in π.

Algorithm 1 IBLR

Require: query $\boldsymbol{x} \in \mathbb{X}$, training data T, integer k
Ensure: label ranking estimation for $\boldsymbol{x}$

1: find the k nearest neighbors of $\boldsymbol{x}$ in T
2: get neighbor rankings $\boldsymbol{\sigma} = \{\sigma_1 \dots \sigma_k\}$
3: use generalized Borda count to get $\hat{\pi}$ from $\boldsymbol{\sigma}$
4: **for** every ranking $\sigma_i \in \boldsymbol{\sigma}$ **do**
5: **if** σ_i is incomplete **then**
6: $\sigma_i^* \leftarrow$ most probable extension of σ_i given $\hat{\pi}$
7: **end if**
8: **end for**
9: use Borda count to get π from $\boldsymbol{\sigma}^* = \{\sigma_1^* \dots \sigma_k^*\}$
10: **if** $\pi \neq \hat{\pi}$ **then**
11: $\hat{\pi} \leftarrow \pi$
12: go to step 4
13: **else**
14: estimate $\hat{\theta}$ given $\hat{\pi}$ and $\boldsymbol{\sigma}^*$
15: return $(\hat{\pi}, \hat{\theta})$
16: **end if**

(ii) Solving the (generalized) median problem (4) is known to be NP-hard for Kendall's tau, i.e., if the distance $D(\cdot)$ is given by the number of rank inversions (Alon, 2006). To solve this problem approximately, we make use of the fact that Kendall's tau is well approximated by Spearman's rank correlation (Coppersmith et al., 2006), and that the median can be computed for this measure (i.e., for $D(\cdot)$ given by the sum of squared rank differences) by an efficient procedure called *Borda count* (Hüllermeier et al., 2008): Given a (complete) ranking σ_i of n labels, the top-label receives n votes, the second-ranked $n-1$ votes, and so on. Given k rankings $\sigma_1 \dots \sigma_k$, the sum of the k votes are computed for each label, and the labels are then ranked according to their total votes.

(iii) The choice of the initial center ranking in the above algorithm is of course critical. To find a good initialization, we again resort to the idea of solving the problem (4) approximately using the Borda count principle. At the beginning, however, the neighbor rankings σ_k are still incomplete (and, since there is no π either, cannot be completed by an M-step). To handle this situation, we make the assumption that the completions are uniformly distributed in $E(\sigma_i)$. In other words, we start with the initial guess $\theta = 0$ (uniform distribution). Based on this assumption, we can show the following result (proof again omitted) that suggests an optimal initial center π^*.

Proposition 2: Let a set of incomplete rankings $\sigma_1 \dots \sigma_k$ be given, and suppose the associated complete rankings $\sigma_1^* \dots \sigma_k^*$ to be distributed, respectively, uniformly in $E(\sigma_1) \dots E(\sigma_k)$. The expected sum of distances $D(\pi, \sigma_1^*) + \dots + D(\pi, \sigma_k^*)$, with D the sum of squared rank distances, becomes minimal for the ranking π^* which is obtained by a generalized Borda count, namely a Borda count with a generalized distribution of votes from incomplete rankings: If σ_i is an incomplete ranking of $m \leq n$ labels, then the label on rank $i \in \{1 \dots m\}$ receives $(m-i+1)(n+1)/(m+1)$ votes, while each missing label receives a vote of $(n+1)/2$.

As a nice feature of our approach, not shared by existing methods for label ranking, we note that it comes with a natural measure of the reliability of a prediction $\hat{\pi}$, namely the estimation of the spread θ. In fact, the larger the parameter θ, the more peaked the distribution around the center ranking and, therefore, the more reliable this ranking becomes as a prediction.

Practically, we found that a distance weighing of instances, a common procedure in nearest neighbor estimation, improves performance. Here, we used a simple weighing scheme in which the weight of the i-th nearest neighbor is given by $(d_{(k)} - d_{(i)})/(d_{(k)} - d_{(1)})$, where

$d_{(i)}$ is the distance of the i-th nearest neighbor from the query (Dudani, 1976). Besides, another form of weighing turned out to be useful: If an incomplete label ranking σ_i is extended to a complete ranking in the course of our (approximate) EM approach, the latter is arguably less reliable than a truly observed ranking. Therefore, we weigh the corresponding instance by $|\mathcal{Y}(\sigma_i)|/n$. The total weight of an instance is then given by this value multiplied with its distance weight. To make our inference procedure amenable to weighted instances, the Borda count principle is replaced by the *weighted Borda count* (Ho et al., 1994).

5. Decision Trees for Label Ranking

Decision tree induction is one of the most extensively studied methods in machine learning (Breiman et al., 1984; Quinlan, 1993), where tree-based models have been used for the purpose of classification as well as regression learning. Decision trees are competitive to other state-of-the-art methods in terms of predictive accuracy and, not less relevant, are generally considered as being more comprehensible and interpretable than most other model classes in machine learning.

In a decision tree, each leaf node represents a (typically rectangular) part of the instance space $\mathbb{X}$ and is labeled with a local model for prediction. In regression, the model is given in the form of a constant or linear function, while in classification, it is simply a class assignment. Here, we are interested in learning decision trees the leaf nodes of which are associated with (possibly incomplete) label rankings.

5.1. Tree Induction

To learn such trees, we resort to the common principle of partitioning the training data in a recursive way, using one-dimensional splits defined by thresholds for an attribute value. This approach is well-known in the machine learning field and therefore not fully detailed here. Our current implementation is restricted to binary splits and only handles numerical attributes.

The main modification of conventional decision tree learning concerns the split criterion at inner nodes and the criterion for stopping the recursive partitioning. As to the former, consider a subset $T = \{(\boldsymbol{x}_i, \sigma_i)\}$ of the training data, where the σ_i are possibly incomplete rankings. A predicate of the form $(A \leq a)$, where A is an attribute used to characterize instances and a is a value of this attribute's domain, splits T into two subsets T^+ and T^-. The goal of a split is to make these subsets as homogeneous or, say, pure as possible. In general, this can be done by fitting a local model to the examples in T^+ and T^-, respectively, and measuring the quality of these fits. In regression trees, for example, this can be realized by taking the mean μ of the output values (i.e., the constant mapping $\boldsymbol{x} \mapsto \mu$) as the model and the variance as a quality measure. Indeed, the mean and variance are optimal estimators if one assumes the data to be an i.i.d. sample from a (locally constant) normal distribution.

We can essentially apply the same principle, though instead of fitting a normal distribution to a real-valued sample, we fit the Mallows model to a set of (possibly incomplete) label rankings. In Section 4.2, we have proposed a solution for this problem. This approach delivers an estimation of the "mean value" in the form of a center ranking $\hat{\pi}$, and an estimation of the variance in terms of the spread parameter $\hat{\theta}$. As a goodness-of-split measure, we can hence use a weighted average of the within-leaf variances, just like in the case of regression trees:

$$|T|^{-1} \left(|T^+| \cdot \theta^+ + |T^-| \cdot \theta^-\right) \ , \tag{6}$$

where θ^+ (θ^-) denotes the estimated spread for the sample T^+ (T^-). Thus, a split is optimal if it maximizes (6).

Note that, in the course of recursive partitioning, it can happen that a set of examples T is split in such a way that T^+ (or T^-) only contains examples $(\boldsymbol{x}_i, \sigma_i)$ in which a certain label is missing. In contrast to our instance-based method, we estimate a ranking for a training set T only on the subset of labels that are present in T. In fact, in the case of missing labels, the recursive partitioning scheme suggests the acquisition of further information from predecessor nodes. This idea will be detailed below.

Our current stopping criterion is rather simple. First, we stop if T is completely pure, which means that all rankings in T are consistent in the following sense: If two labels $y_i, y_j \in \mathcal{Y}$ both occur in two different rankings in T, then they are put in the same order. For the estimation of the parameters of the Mallows model, this is an exceptional case which leads to $\theta = \infty$. Second, to prevent an excessive fragmentation, we stop if the number of labels in a node becomes too small; concretely, we currently use $2|\mathcal{Y}|$ as a lower bound. This latter condition implements a kind of pre-pruning, though a very simple one. As opposed to this, we have not yet implemented a post-pruning step in which the induced tree is simplified.

One may suspect that decision trees for ranking can become undesirably large, due to the high number of different predictions that can be made. First, however, one should note that the model complexity is

not directly determined by this number. Instead, it is determined by the typically much smaller number of outputs that are really attained, as well as their distribution in the instance space. Second, the problem is arguably not worse for label ranking than it is for regression, where tree-based models offer good (approximate) models despite the potentially infinite number of different outcomes. Our experimental results will indeed show that label ranking trees are typically not much bigger than classification trees.

5.2. Prediction

Once a decision tree has been constructed, it can be used to derive predictions for new query instances. This is done as usual, namely by propagating the instance through the tree until reaching a leaf node. The prediction is then given by the label ranking σ associated with this leaf.

Here, a problem only occurs if σ is not complete. As mentioned above, this can happen if not all labels are present in a set of examples T. In such a case, we complete the prediction σ by resorting to the predecessor nodes of the leaf. More specifically, this is done as follows. The first predecessor is searched that contains more label information than the leaf; let the ranking associated with this node be π. The ranking of the leaf, σ, is then expanded to a ranking σ^* by looking for the ranking of the labels in $\mathcal{Y}(\pi)$ which is as close to π, in terms of the distance (1), while retaining the ordering of the labels in σ. This is exactly the same problem that we already faced in the instance-based approach, and that we solved on the basis of Proposition 1. Thus, the same solution can be used here. In case the expansion does still not contain all labels, the procedure is continued, i.e., σ^* is further expanded on the basis of the ranking of the next predecessor with more label information (note that the root of the tree contains full information unless there is a label that has never been observed).

6. Experimental Results

In this section, we present an empirical evaluation of instance-based label ranking (IBLR) and label ranking trees (LRT) as introduced in the previous sections. In (Hüllermeier et al., 2008), it was shown that constraint classification, log-linear models for label ranking, and ranking by pairwise comparison are quite comparable in terms of predictive accuracy. Here, we use constraint classification (CC) as a baseline to compare with, as it has the smallest number of degrees of freedom and, therefore, a comparatively straightforward implementation. Concretely, CC was implemented in its online-variant as proposed in (Har-Peled et al., 2003), using a noise-tolerant perceptron algorithm as a base learner (Khardon & Wachman, 2007).[1] For IBLR, the neighborhood size was selected through cross validation on the training set. As a distance measure on the instance space we simply used the Euclidean distance (after normalizing the attributes).

6.1. Data

In view of a lack of benchmark data for label ranking, we resorted to multi-class and regression data sets from the UCI repository and the Statlog collection and turned them into label ranking data in two different ways. (A) For classification data, we followed the procedure proposed in (Hüllermeier et al., 2008): A naive Bayes classifier is first trained on the complete data set. Then, for each example, all the labels present in the data set are ordered with respect to the predicted class probabilities (in the case of ties, labels with lower index are ranked first). (B) For regression data, a certain number of (numerical) attributes is removed from the set of predictors, and each one is considered as a label. To obtain a ranking, the attributes are standardized and then ordered by size. Given that the original attributes are correlated, the remaining predictive features will contain information about the ranking thus produced. Yet, as will be confirmed by the experimental results, this second type of data generation leads to more difficult learning problems. A summary of the data sets and their properties is given in Table 1.[2]

Table 1. Data sets and their properties (the type refers to the way in which the data has been generated).

data set	type	# inst.	# attr.	# labels
authorship	A	1513	70	4
bodyfat	B	452	7	7
calhousing	B	37152	4	4
cpu-small	B	14744	6	5
elevators	B	29871	9	9
fried	B	73376	9	5
glass	A	382	9	6
housing	B	906	6	6
iris	A	270	4	3
pendigits	A	19784	16	10
segment	A	4158	18	7
stock	B	1710	5	5
vehicle	A	1518	18	4
vowel	A	944	10	11
wine	A	314	13	3
wisconsin	B	346	16	16

[1] This algorithm is based on the "alpha-trick". We set the corresponding parameter α to 500.

[2] The data sets, along with a description, are available at `www.uni-marburg.de/fb12/kebi/research`

6.2. Experiments and Results

Results were derived in terms of Kendall's tau coefficient from five repetitions of a ten-fold cross-validation. To model incomplete observations, we modified the training data as follows: A biased coin was flipped for every label in a ranking to decide whether to keep or delete that label; the probability for a deletion is specified by a parameter $p \in [0, 1]$. Hence, $p \times 100\%$ of the labels will be missing on average.

The summary of the results is shown in Table 2. To analyze these results, we followed the two-step procedure recommended in (Demsar, 2006), consisting of a Friedman test of the null hypothesis that all learners have equal performance and, in case this hypothesis is rejected, a Nemenyi test to compare learners in a pairwise way. Both tests are based on the average ranks (for each problem, the methods are ranked in decreasing order of performance, and the ranks thus obtained are averaged over the problems) as shown in the bottom line in Table 2. At a significance level of 10%, IBLR is better than both CC and LRT in the case of complete rankings and better than LRT for 30% missing label information, while there are no significant differences in the case of 60% missing labels. Besides, CC and LRT perform more or less on par in all three settings, without any significant differences.

As mentioned earlier, we hypothesize that, since our tree- and instance-based methods for label ranking fit local models to the data, they are especially useful for problems requiring complex decision boundaries. Some evidence in favor of this hypothesis is indeed provided by the learning curves depicting the performance as a function of the fraction of missing label information. While the learning curves of CC are often rather flat, showing a kind of saturation effect, they are much steeper for IBLR and LRT. This suggests that additional label information is still beneficial for these methods even when CC, due to a lack of flexibility, is no longer able to exploit and adapt to extra data. A typical example is the housing data, for which the learning curves are shown in Fig. 1.

For LRT, Table 2 further shows the relative size of the trees, namely the average number of leaf nodes divided by the number of leaf nodes of the tree learned by J48, the C4.5 implementation of WEKA (in its default setting), on the corresponding classification data (which is obtained from the label ranking data by keeping the top-label as class information and removing the rest of the ranking). As can be seen, there is no dramatic difference in the size of the trees, and sometimes the label ranking tree is even smaller.

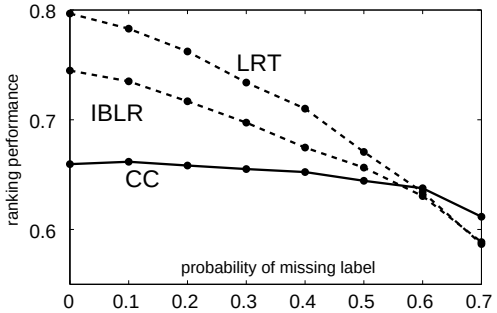

Figure 1. Ranking performance (in terms of Kendall's tau) as a function of the missing label rate.

We conclude this section with short remarks on two issues for which no empirical results are presented due to space restrictions. First, as to the computational complexity of the label ranking methods, a direct comparison is complicated by the fact that IBLR is a lazy learner, with almost no costs at training time but higher costs at prediction time. Anyway, in their current implementations, both IBLR and LRT are very efficient and quite comparable, in terms of runtime, to their corresponding counterparts for classification. This is true despite the more complex local estimation procedures, mainly because our approximate EM procedure converges very quickly.

Second, as mentioned earlier, an advantage of a local estimation method is that it delivers, as a byproduct, natural measures of the reliability of a prediction. In the case of IBLR, the estimated spread $\hat{\theta}$ is such a measure. For LRT, the $\hat{\theta}$ associated with a leaf cannot be used directly, since the leafs are deliberately constructed so as to maximize purity. Therefore, we tried a modified measure, namely the product of $\hat{\theta}$ and the fraction of examples in the leaf node. Anyway, in both cases, the reliability measures showed a very high correlation with the quality of prediction (in terms of Kendall's tau), suggesting that they are indeed reasonable indicators of the uncertainty of a prediction.

7. Conclusions and Future Work

We proposed two novel methods for label ranking, namely instance-based label ranking (IBLR) and label ranking trees (LRT). As a core component, both approaches share the problem to estimate local models, IBLR in the neighborhood of a query and LRT in a subregion of the instance space. Assuming that the probability distribution of the output is locally constant, we solve this problem by deriving (an approximation of) an ML estimation based on the Mallows model, a popular probability model for rankings.

Table 2. Performance of the label ranking methods in terms of Kendall's tau (in brackets the rank). The columns for LRT also contain the relative tree size (number to the right of performance).

	complete rankings			30% missing labels			60% missing labels		
	CC	IBLR	LRT	CC	IBLR	LRT	CC	IBLR	LRT
authorship	.920(2)	.936(1)	.882(3) 1.1	.891(2)	.932(1)	.871(3) 0.9	.835(2)	.920(1)	.828(3) 0.7
bodyfat	.281(1)	.248(2)	.117(3) 1.6	.260(1)	.223(2)	.097(3) 1.7	.224(1)	.180(2)	.070(3) 1.0
calhousing	.250(3)	.351(1)	.324(2) 0.7	.249(3)	.327(1)	.307(2) 0.5	.247(3)	.289(1)	.273(2) 0.3
cpu-small	.475(2)	.506(1)	.447(3) 2.3	.474(2)	.498(1)	.405(3) 2.3	.470(2)	.480(1)	.367(3) 1.5
elevators	.768(1)	.733(3)	.760(2) 0.2	.767(1)	.719(3)	.756(2) 0.2	.765(1)	.690(3)	.742(2) 0.3
fried	.999(1)	.935(2)	.890(3) 5.5	.998(1)	.928(2)	.863(3) 5.3	.997(1)	.895(2)	.809(3) 3.0
glass	.846(3)	.865(2)	.883(1) 2.5	.835(2)	.824(3)	.850(1) 2.0	.789(2)	.771(3)	.799(1) 2.0
housing	.660(3)	.745(2)	.797(1) 2.3	.655(3)	.697(2)	.734(1) 2.4	.638(1)	.630(3)	.634(2) 1.5
iris	.836(3)	.966(1)	.947(2) 1.5	.807(3)	.945(1)	.909(2) 1.2	.743(3)	.882(1)	.794(2) 1.5
pendigits	.903(3)	.944(1)	.935(2) 6.2	.902(3)	.924(1)	.914(2) 3.2	.900(1)	.899(2)	.871(3) 2.2
segment	.914(3)	.959(1)	.949(2) 3.8	.911(3)	.934(1)	.933(2) 3.8	.902(2)	.902(3)	.903(1) 2.3
stock	.737(3)	.927(1)	.895(2) 1.5	.735(3)	.904(1)	.877(2) 1.6	.724(3)	.858(1)	.827(2) 1.1
vehicle	.855(2)	.862(1)	.827(3) 0.8	.839(2)	.842(1)	.819(3) 0.9	.810(1)	.791(2)	.764(3) 0.5
vowel	.623(3)	.900(1)	.794(2) 4.6	.615(3)	.824(1)	.718(2) 3.6	.598(3)	.722(1)	.615(2) 3.2
wine	.933(2)	.949(1)	.882(3) 0.8	.911(2)	.941(1)	.862(3) 1.1	.853(1)	.789(2)	.752(3) 0.8
wisconsin	.629(1)	.506(2)	.343(3) 1.6	.617(1)	.484(2)	.284(3) 1.5	.566(1)	.438(2)	.251(3) 1.6
average rank	2.25	1.44	2.31	2.19	1.50	2.31	1.75	1.88	2.38

As mentioned previously, both methods complement existing approaches in a reasonable way and do have some advantages. As shown by our empirical studies, IBLR is particularly strong in terms of predictive accuracy. Label ranking trees, while being at least competitive to state-of-the-art label ranking methods in this regard, are especially appealing from a comprehensibility point of view. Besides, both methods provide natural measures of the reliability of a prediction.

Despite the already strong performance of both methods, there is of course scope for further improvement. In the case of IBLR, for example, extensions known to be beneficial for classification learning, such as instance selection, feature weighing, or locally adaptive metrics, are likely to improve performance in label ranking, too. The most important extension of LRT, to be addressed in future work, is an effective pruning strategy to simplify a learned tree and prevent from a possible overfitting of the data.

References

Aha, D., Kibler, D., & Albert, M. (1991). Instance-based learning algorithms. *Mach. Learn.*, *6*, 37–66.

Alon, N. (2006). Ranking tournaments. *SIAM Journal on Discrete Mathematics*, *20(1)*, 134–142.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth International Group.

Coppersmith, D., Fleischer, L., & Rudra, A. (2006). Ordering by weighted number of wins gives a good ranking for weighted tournaments. *ACM-SIAM Symposium on Discrete Algorithms* (pp. 776–782).

Dekel, O., Manning, C., & Singer, Y. (2004). Log-linear models for label ranking. *Advances in Neural Information Processing Systems 16* (pp. 497–504). MIT Press.

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. of the Royal Stat. Soc. B*, *39*, 1–38.

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

Dudani, S. (1976). The distance-weighted k-nearest-neighbor rule. *IEEE Trans. SMC*, *6*, 325–327.

Fligner, M., & Verducci, J. (1986). Distance based ranking models. *J. Royal Stat. Soc.*, *48*, 359–369.

Har-Peled, S., Roth, D., & Zimak, D. (2003). Constraint classification for multiclass classification and ranking. *Advances in Neural Information Processing Systems 15* (pp. 785–792). MIT Press.

Ho, T., Hull, J., & Srihari, S. (1994). Decision combination in multiple classifier systems. *IEEE Trans. Pattern Anal. and Machine Intell.*, *16*, 66-75.

Hüllermeier, E., Fürnkranz, J., Cheng, W., & Brinker, K. (2008). Label ranking by learning pairwise preferences. *Artificial Intelligence*, *172*, 1897–1916.

Khardon, R., & Wachman, G. (2007). Noise tolerant variants of the perceptron algorithm. *Journal of Machine Learning Research*, *8*, 227–248.

Mallows, C. (1957). Non-null ranking models. *Biometrika*, *44(1)*, 114–130.

Quinlan, J. (1993). *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufmann.