Journal of Classification (2022) 39:78–99
https://doi.org/10.1007/s00357-021-09397-2

Comparing Boosting and Bagging for Decision Trees of Rankings

Antonella Plaia[1] · **Simona Buscemi**[1] · **Johannes Fürnkranz**[2] · **Eneldo Loza Mencía**[3]

Accepted: 10 August 2021 / Published online: 3 September 2021

Abstract
Decision tree learning is among the most popular and most traditional families of machine learning algorithms. While these techniques excel in being quite intuitive and interpretable, they also suffer from instability: small perturbations in the training data may result in big changes in the predictions. The so-called ensemble methods combine the output of multiple trees, which makes the decision more reliable and stable. They have been primarily applied to numeric prediction problems and to classification tasks. In the last years, some attempts to extend the ensemble methods to ordinal data can be found in the literature, but no concrete methodology has been provided for preference data. In this paper, we extend decision trees, and in the following also ensemble methods to ranking data. In particular, we propose a theoretical and computational definition of bagging and boosting, two of the best known ensemble methods. In an experimental study using simulated data and real-world datasets, our results confirm that known results from classification, such as that boosting outperforms bagging, could be successfully carried over to the ranking case.

Keywords Boosting · Rankings · Ensemble methods · Preference learning · Decision trees

✉ Antonella Plaia
antonella.plaia@unipa.it

Simona Buscemi
simona.buscemi@unipa.it

Johannes Fürnkranz
juffi@faw.jku.at

Eneldo Loza Mencía
eneldo@ke.tu-darmstadt.de

[1] Department of Economics, Business and Statistics, University of Palermo, Palermo, Italy

[2] Computational Data Analytics, Johannes Kepler University Linz, Linz, Austria

[3] Knowledge Engineering Group, Technische Universität Darmstadt, Darmstadt, Germany

1 Introduction

In the 1980s, Breiman et al. (1984) developed Classification and Regression Trees (CART) as alternative, non-parametric approaches to classification and regression. The tree-structured predictors are obtained by recursively partitioning the data space into a set of (usually) axis-parallel hyper-rectangles, with similar response values grouped together, so that, in the end, the resulting groups are as homogeneous as possible with respect to the considered response. The resulting decision trees suffer from high variance, i.e. the decision trees learned from different data samples drawn from the same distribution may be quite different. In other words, the learned models are unstable, i.e. small perturbations in the training set could cause large changes in the resulting predictors.

Breiman (1996) suggested to improve the stability of decision trees by learning an ensemble of diverse trees whose predictions are combined. In particular, by repeatedly perturbing the training set, learning multiple decision trees from these perturbed sets, and finally then combining their individual predictions into an overall prediction, it is possible to improve the quality of the predictions over those obtained from individual single trees. The general class of such predictors is called *perturb and combine* (P&C; see Breiman (1996) for other examples of P&C procedures), and they in turn form a special case of what is generally known as *ensemble techniques* in machine learning (Dietterich, 2000).

One of the best known P&C methods, *bagging* (Boostrap AGGregatING; Breiman 1996), perturbs the training set several times using bootstrapping, a general statistical method in which several (non-disjoint) training sets are obtained by drawing examples randomly, with replacement, from a single base dataset (Efron, 1982). The multiple predictors, which are learned from such bootstrap samples, are then combined by simple voting, in the case of classification, or by averaging for regression. In each iteration, the bootstrap samples are drawn with the same probability, which guarantees independent bootstrap training sets. Bagging without replacement was proposed as "subagging = subsample aggregating" by Bühlmann and Yu (2000, 2002) as a computational cheaper version of bagging (Bühlmann, 2003). Buja and Stuetzle (2006) analytically proved the equivalence of resampling with and without replacement in Bagging. A comparison between bagging with or without replacement can be found also in Friedman and Hall (2007).

As an alternative to bagging, Freund and Schapire (1996) and Freund and Schapire (1998) proposed *boosting*, another P&C method which aims at a faster reduction of training set errors. The key idea is to learn the ensemble of predictors sequentially, so that each predictor can focus on correcting the mistakes of the previous ones. This is obtained by increasing the probabilities for sampling examples that have been misclassified in previous iterations, so that they are more prevalent in subsequent iterations. Thus, unlike in bagging, the samples used in the boosting iterations are clearly not independent. Breiman (1998) considers boosting to be a special case of the class of *arcing* (adaptive resampling and combining) classifiers, where units that have frequently been classified as incorrect have an increased probability to be drawn in subsequent iterations.

Decision trees and ensemble methods have been widely applied to quantitative or qualitative response variables, in sociological, medical or psychological domains. For example, Austin (2012) compare inferences on the effect of in-hospital smoking cessation counselling on subsequent mortality in patients hospitalized with an acute myocardial infarction using ensemble-based methods (bagged regression trees, random forests, and boosted regression trees) to directly estimate average treatment effects by imputing potential outcomes. Stegmann et al. (2018) introduce nonlinear Longitudinal Recursive Partitioning (nLRP) and

illustrate its use with empirical data from the kindergarten cohort of the Early Childhood Longitudinal Study. Or again, recently, Grimm and Jacobucci (2020) improve the reliability of splitting functions in the case of small data samples, and illustrate their performance using data on depression and suicidal ideation from the National Longitudinal Survey of Youth. In all these works decision trees and ensemble methods are applied in situations where the outcome is a categorical or a continuous variable but, to our knowledge, there is not much work in that area that allows to cope with ranking or preference data.

Preference data may be viewed as samples in which some subjects (voters, judges), characterized by a set of features, indicate their preferences over a set of alternatives (items). In many real-world cases, items may also receive the same preference by a judge, resulting in rankings with ties. The goal is to learn a function that allows to rank these items according to the predicted preferences of a new subject, given his or her characteristics. In preference learning (Fürnkranz & Hüllermeier, 2011), this setting is referred to as *label ranking*, to distinguish it from *object ranking*, a related but different scenario which does not involve judges, but tries to predict a universal preference degree for items based on their characteristics. A typical example is to learn to rank Web pages for a search result. Freund et al. (2003) discuss a boosting approach for preference data in object ranking problems, but very few works can be found in literature on label ranking problems.

Aledo et al. (2017), taking as a basis the Label Ranking Tree (LRT)-based algorithm proposed by Hüllermeier et al. (2008) and described in (Cheng et al., 2009), design weaker tree-based models which can be learnt more efficiently and show that bagging these weak learners improves not only the LRT algorithm, but also the state-of-the-art competitors. Werbin-Ofir et al. (2019) focus mainly on the aggregation problem and again used the LRT label ranking base algorithm, which is based on the estimation of a Mallows model (Mallows, 1957) in each node of the tree. They propose to apply voting rules, typically used in the field of social choice, as the aggregation technique for label ranking ensembles. As they found that there is no single rule that consistently outperforms all other voting rules and that under different settings, different voting rules perform the best, they propose a novel aggregation method for label ranking ensembles, the voting rule selector (VRS), which learns the best voting rule to be used in a given setting. Both de Sá et al. (2017) and Zhou and Qiu (2018) use random forests (Breiman, 2001) for ensemble formation, the former considering label ranking trees (de Sá et al., 2015) as base classifiers, the latter using the top level in the ranking as the class in decision tree construction. A key, non-trivial step in ranking trees is to devise a procedure for aggregating the various rankings associated with all examples in a node into a consensus ranking. To this end, de Sá et al. (2017) use the average ranking, whereas Zhou and Qiu (2018) rely on Borda's method. Dery and Shmueli (2020) propose a boosting algorithm, BoostLR, quite similar to the one proposed in our paper. They use LRT (Hüllermeier et al., 2008) as the base algorithm and weighted Borda's count as aggregation method. We will show in Section 4.2 how and why our proposal differs from Dery and Shmueli (2020).

The purpose of this paper is to define, analytically and empirically, boosting and bagging algorithms for rankings, with or without ties, and to evaluate these algorithms on real and simulated data. We start with a brief description of ranking data and distances between rankings in Section 2. Section 3 introduces decision trees and how they can be applied to ranking data, describing the rank aggregation process. Section 4 shows how these ranking trees can be used in bagging and boosting algorithms. Finally, in Section 5, we discuss a real application and a simulation study, and, in the end, conclude with a brief conclusion.

2 Ranking and Preference Data

Ranking and classification are frequently used in order to grade various experiences in our lives. Grouping and ordering a set of elements is quite easy and meaningful. Examples include rankings of football teams, universities, and countries.

A special case of ranking data constitutes *preference data*, where some individuals (called *judges*) indicate their preferences over a set of alternatives (which, in the literature, are also referred to as options, stimuli, or items) and are asked to order them from most to least preferred.

2.1 Motivation

Preference data can be found as pairwise comparisons, when respondents are asked to select the more preferred alternative from each pair of alternatives. Note that paired comparison and ranking methods, especially when differences between choice alternatives are small, impose lower constraints on the response behaviour than rating methods. Rounds et al. (1978) say that "The method of paired comparisons ... has been frequently applied in conjunction with the law of comparative judgment to scaling psychological variables.", and they cite the seriousness of crimes, patients' mental health, visual illusions, life goals, food preferences etc. as examples of stimuli that have been scaled with these methods. Presenting the objects in pair to judges, and then producing paired comparison rankings, could be the natural experimental procedure when the objects to be ranked are really similar and the introduction of other items may be confusing (David, 1969). Of course, as the number of stimuli increases, paired comparisons become exceedingly time-consuming and laborious for the subjects, while directly ranking the stimuli involves less labour. A great deal of data, in psychological research for example, can be considered the result of a choice process (Maydeu-Olivares & Bockenholt, 2009), and if item and person characteristics are available, appropriate statistical tools can be used to identify how respondents differ in their perception and preferences for a set of choice options. Maydeu-Olivares and Bockenholt (2005) stress that the methods of ranking and paired comparison play an essential role in the measurement of preferences, attitudes, and values. They report two applications: in the first one, a Spanish university wished to investigate career preferences among its undergraduate psychology students. A pilot study was performed in which a sample of 57 psychology sophomores was asked to express their preferences for four broad psychology career areas (Academic, Clinical, Educational, and Industrial) using a ranking task. The second one has the goal to model purchasing preferences for seven compact cars among Spanish college students. In both examples, covariates may be included to explain individual differences in the evaluation of choice alternatives.

Preference data are typical of marketing analyses, but also of political and social choices science surveys and, more generally, of behavioural studies (staff selections, policy alternatives assessment, etc.). The primary aim of market segmentation, for example, is to identify relevant groups of consumers that can be addressed efficiently by marketing or advertising campaigns. But, as Müllensiefen et al. (2018) have observed, the issue whether consumer groups can be identified from background variables, that are not brand-related, and particularly how much personality vs socio-demographic variables contribute to the identification of consumer clusters is very important to address. Indeed, "the term 'psychographics' describes the collection of data on variables that reflect consumer personality, attitudes, personal values, lifestyle and other psychological constructs in order to identify and describe subpopulations of consumers and ultimately to inform marketing processes" (Müllensiefen

et al., 2018). Moreover, the authors underline how "personality information can be gathered indirectly from online data such as Facebook likes, Twitter profiles, musical preferences or spending behaviour. Thus, unlike other psychographics measures, personality has become a layer of information that can be obtained from various sources, especially online information, and that has proven to be useful for predicting a broad variety of outcome measures, such as substance use, political attitudes, or purchase satisfaction".

In the last two decades, reasoning with and learning of preferences has received increased attention in the machine learning (Fürnkranz & Hüllermeier, 2011) and artificial intelligence (Rossi et al., 2011) literature.

Consequently, many models have been proposed able to cope with ranking data, when respondent-specific variables are available, and several of them focused specifically on decision tree models. Some of the best known include the distance-based tree models proposed by Lee and Yu (2010), decision trees with ad-hoc impurity functions suggested by Yu et al. (2010) and multivariate trees for rankings based on distance measures of D'Ambrosio (2008).

2.2 Rankings

Formally, a *ranking* of m items, labeled $(1, \ldots, m)$, is a mapping a from the set of items $\{1, \ldots, m\}$ to the set of ranks $\{1, \ldots, m\}$. When all items are ranked in m distinct ranks, a complete ranking or linear ordering (Cook et al., 1986) is observed. A ranking a is, therefore, one of the $m!$ possible permutations of m elements, representing the preference ordering of one particular judge for the items involved. But if a judge fails to distinguish between two or more objects and he/she assigns equal importance to them, a *tied ranking* or *weak ordering* is obtained. Moreover, rankings may not be complete: *partial rankings* occur when only a specific subset of $q < m$ objects are ranked by judges, while *incomplete rankings* occur when judges freely rank different subsets of m objects (Cook et al., 1986). It is worth noting that different types of orderings will generate different sample spaces of rankings. With m items there are $m!$ possible complete rankings, but this number increases when ties are allowed (for the universe cardinality in the presence of ties, we refer to Good (1980) and Marcus (2013). The size of the universe of all possible rankings with ties, S^m, is equal to the following quantity:

$$S^m = \sum_{r=0}^{m} r! \left\{ {m \atop r} \right\},$$

where $\left\{ {m \atop r} \right\}$ states the Stirling number of the second kind, indicating the number of all possible ways to partition a set of m objects into r non-empty subsets, i.e. $\left\{ {m \atop r} \right\} = \frac{1}{r!} \sum_{i=0}^{r} (-1)^i \binom{r}{i} (r-i)^m$ (D'Ambrosio et al., 2017).

Zhou et al. (2014) propose a taxonomy of label ranking algorithms, where *label ranking* studies a mapping from instances to rankings over a finite number of predefined labels, in other words *ranking data.* They mention four subtypes of label ranking: (a) Multiclass classification, (b) Multilabel classification, (c) Label ranking and (d) Multilabel ranking. Case a is when each instance is associated with a single label: this is equivalent to say that in a ranking we are interested only to the first (top-1) position. Case b is when we are interested only in a subset of the labels, that share the same importance: that is to say that we are interested only to the top-k positions, equally ranked. Case c corresponds to a linear ranking — all items are ranked and without ties. Case d, finally, is a typical top-k or what we have called partial ranking (Hall & Schimek, 2012; Sampath & Verducci, 2013; Svendova

& Schimek, 2017). All these types can be handled in the framework of this paper. Further details on Label Ranking can be found in Vembu and Gärtner (2010).

2.3 Distances Between Rankings

A natural desiderata is to group subjects with similar preferences together. To this end, it is necessary to measure the spread between rankings through dissimilarity or distance measures, where a *distance d* between two rankings is a non-negative value, ranging from 0 to $D_{\max}$, where 0 is the distance between a ranking and itself. Many different distance functions between rankings have been proposed in the literature (Marcus, 2013), often defined as functions of the $m \times m$ *score matrix* (a_{ij}) (defined for the generic ranking a), whose elements are defined as:

$$a_{ij} = \begin{cases} 1 \text{ if } i \text{ is preferred to } j, \\ 0 \text{ if } i \text{ is tied to } j \text{ or } i = j, \\ -1 \text{ if } j \text{ is preferred to } i. \end{cases}$$

Kemeny & Snell (1962) introduced a metric defined on linear and weak orders, known as *Kemeny distance* (or metric), which satisfies the constraints of a distance measure suitable for rankings. Cook et al. (1986) later generalized Kemeny distances to the framework of partial orders.

In this work, we consider the possibility of ties; therefore, we assume that the geometrical space of preference rankings is the generalized permutation polytope (Heiser & D'Ambrosio, 2013; D'Ambrosio et al., 2017), for which the natural distance measure is the Kemeny distance. It is defined as follows:

$$d_K(a, b) = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} |a_{ij} - b_{ij}|, \tag{1}$$

where a_{ij} and b_{ij} are the generic elements of the $m \times m$ score matrices associated with a and b

Each distance between orderings is in a one to one correspondence to a proper correlation coefficient, through the linear transformation $c = 1 - \frac{2d}{D_{\max}}$

(Emond & Mason, 2002). The rank correlation coefficient τ_x corresponding to the Kemeny-Snell distance is defined as (Emond & Mason, 2002)

$$\tau_x(a, b) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{m} a'_{ij} b'_{ij}}{m(m-1)}, \tag{2}$$

where a'_{ij} and b'_{ij} are the generic elements of the $m \times m$ score matrices associated with a and b respectively, assuming now that the value is equal to 1 if the item i is preferred to or tied with the item j, -1 if item j is preferred over i, and 0 if $i = j$.

3 Decision Trees for Rankings

As discussed above, preference rankings can be considered as indicators of individual behaviours of a set of individuals, the judges. In the case of preference data, we also have characteristics of the individuals available, so that an important task is the identification of profiles of respondents (or judges) which exhibit a similar behaviour, i.e. which show similar preference rankings.

3.1 Decision Trees

Decision tree learning is a very popular data analysis technique, which has been independently discovered in statistics and machine learning (Murthy, 1998). The CART algorithm (Breiman et al., 1984), which we already briefly introduced in Section 1, starts from the root node that contains the whole learning sample, and uses a recursive partitioning algorithm to create a tree where each node t represents one set of the partition.

At each node, data are divided by choosing a split, among all possible splits, so that the resulting child nodes are the "purest", where pureness is meant in terms of homogeneity (with respect to the response) of observations in the same node. The homogeneity of child nodes is measured by a so-called *impurity function* $i(t)$, a generic function satisfying the following three properties: (a) it is minimum when the node is pure, i.e. when all data points associated with this node have the same response; (b) it is maximum when the node is the most impure; (c) its value does not change if items are renamed. Specifically, a splitting criterion is based on the reduction in impurity resulting from the split s of node t, with the best split chosen as the one maximizing the impurity reduction

$$\Delta i(s,t) = i(t) - p_L i(t_L) - p_R i(t_R),$$

where p_L and p_R are given by the proportions of units in node t assigned to the left child node t_L and to the right child node t_R, respectively, at the sth split.

3.2 Impurity Functions for Ranking Data

The key issue in adapting conventional univariate classification trees to the multivariate case is the generalization of the definition of the impurity function. In order to avoid the problems caused by the multivariate role of the ranking vector, following Sciandra et al. (2015) and Plaia and Sciandra (2019), the set of preferences are hereby valued as a unique multivariate structure. More precisely, the ranking process of m items can be seen as a permutation function from $\{1, \ldots, m\}$ into $\{1, \ldots, m\}$. Assigning a label to each permutation (with or without ties) obtained by a set of distinct items, the ranking vector can be uniquely identified by its label that can play the role of a response variable. This strategy enables us to work with the classical univariate CART methodology.

For a better understanding of our strategy, consider the following illustrative example: for ranking 4 items without ties, there are overall 4! permutations. The first permutation could represent the ranking $\langle 1, 2, 3, 4\rangle$, the second another ranking such as $\langle 2, 1, 3, 4\rangle$, the third one $\langle 1, 2, 4, 3\rangle$ and so forth. If we assign to each ranking the corresponding label "first", "second", "third" etc., the multivariate problem of rankings reduces to a univariate one, where the labels "first", "second", "third", ... represent the response variable, assuming that each one is in a one to one correspondence with a permutation.

However, the impurity function should represent the fact that some rankings are more similar to each other than others. For example, the ranking $\langle 1, 2, 3, 4\rangle$ is quite similar to the ranking $\langle 1, 2, 4, 3\rangle$ (only the order of the last two items in the ranking is reversed), but quite different from $\langle 4, 3, 2, 1\rangle$ (where all preference pairs are inverted). Thus, when the impurity is evaluated on ranking data, the impurity function $i(t)$ can be suitably modified as:

$$i(t) = \sum_{a,b \in t:\, a \neq b} d(a,b), \tag{3}$$

where d is a distance measure between orderings a and b, the sum being extended over all the pairs of orderings in node t. A decrease in node impurity at each step will be evaluated

according to all covariates and respective split points, and the best among them is chosen. Piccarreta (2010) suggests an impurity function based on dissimilarities, without proposing specific distances for preference data. In order for the impurity function to take into account the ordinal nature of rankings, following Sciandra et al. (2015), we propose to use the Kemeny distance $d_K(a, b)$ as defined in (1). Therefore, the base classifier $C_f(.)$ we will use as the main ingredient of our bagging and boosting procedure is the tree-based method for ranking data introduced by Sciandra et al. (2015) and Plaia and Sciandra (2019).

3.3 Rank Aggregation

Once the tree has been grown, we need to assign a class label or a class ranking to each leaf node. To this end, we need to compute a so-called *consensus ranking*, i.e. the ranking that mostly agrees with all the rankings in the node (D'Ambrosio et al., 2015a). Among the pool of several consensus ranking measures proposed in the literature, the median ranking will be used in this paper. The *median ranking* in a node is defined as the ranking that has the minimum sum of the distances to all rankings in the node. Equivalently, it is the ranking that maximizes the average rank correlation coefficient τ_x (2) between itself and the other rankings in the node (D'Ambrosio & Heiser, 2016). In other words, we look for the candidate, within the universe of the permutations with replacement of m elements S^m, that maximizes the rank correlation coefficient (2) or, conversely, minimizes the average Kemeny distance (1). It has been shown, as stated by Zhou and Qiu (2018), that the Kemeny optimal aggregation is the best compromise ranking.

A key problem with this approach is that the naïve computation of the median ranking is computationally prohibitive, as it essentially involves the evaluation of (in the worst case) $m!$ different candidate rankings. Thus, for an efficient computation of the median ranking, we use a branch and bound algorithm proposed by Amodio et al. (2016), implemented in the R package ConsRank (D'Ambrosio et al., 2015b). Emond and Mason (2002) suggested the initial branch and bound algorithm to find the solution in a consensus ranking problem. Concerning the Kemeny approach, D'Ambrosio et al. (2015a), D'Ambrosio et al. (2015b) and Amodio et al. (2016) proposed two accurate algorithms (QUICK and FAST) for identifying the median ranking when treating with linear, weak and partial rankings. The authors showed that the QUICK and FAST algorithms are equivalent to Emond and Mason's branch and bound, but ensure an impressive time-saving from a computational point of view, by starting with a "good" initial solution. The solution they come up with is optimal but might not find all the solutions in case of multiple solutions.

4 Ensemble Methods for Ranking Data

"The idea of ensemble learning is to build a prediction model by combining the strengths of a collection of simpler base models". (Hastie et al., 2009, ch. 16). It is well known that, by aggregating many decision trees for a categorical or continuous outcome, using methods like bagging, random forests, and boosting, the predictive performance of trees can be substantially improved (James et al., 2013). This is true with ranking data as well, as demonstrated by Dery and Shmueli (2020) who report (Figs. 3 and 4) the average improvements of ensemble methods over a single decision tree. Comparable enhancements are shown in our paper and will be illustrated more deeply in Section 5.

In this paper, we form ensembles using ranking trees as building blocks, in order to construct more powerful prediction models than a single tree. Ensemble learning can be

broken down into two tasks: developing a population of base learners (in our case decision trees) from the training data, and combining them to form the composite predictor. We will first discuss the first point, before we show in Section 4.3 the way we aggregated the trees. We briefly outline the main difference between bagging and boosting, the ensemble methods we are going to work with. Bagging (Section 4.1) learns decision trees for many datasets of the same size, randomly drawn with replacement from the training set. Thereafter, a proper predicted ranking is assigned to each unit. Boosting on the other hand (Section 4.2) combines classifiers, iteratively created from weighted versions of the learning sample, with weights adaptively modified, iteration by iteration, so that previously misclassified rankings have a higher probability of being selected in subsequent iterations. The final predicted rankings are computed by weighted combination of the intermediate rankings of the iterative process.

4.1 Bagging for Ranking Data

The simplest implementation of the idea of generating quasi-replicate training sets is bagging (Breiman, 1996), which is briefly summarized in Algorithm 1.

Algorithm 1 Bagging for ranking data.

Input: a training set T, a number of iterations B
Output: a ranker $C_f(.)$ that maps a given x to a ranking of the labels
for $b = 1$ **to** B **do**
  take a bootstrap replicate T_b of the training set T
  construct a ranking tree $C_b(.)$ from T_b
end
$C_f(x_i) = \arg\max_{y_i \in S^m} \sum_{b=1}^{B} \tau_x(C_b(x_i), y_i))$

In a few words, several training datasets T_b of the same size are sampled at random, with replacement, from the training set. Each T_b is used to train a decision tree $C_b(.)$, leading to multiple predicted rankings $C_b(x_i)$ for each data point x_i. A final ranking prediction $\hat{y}_i = C_f(x_i)$ is assigned to each unit x_i through the consensus ranking process briefly described in Section 3.3 and detailed in Section 4.3. Combining multiple classifiers decreases the expected error by reducing the variance. More classifiers in the ensemble typically lead to a greater reduction of the error (at least on the training set). The procedure allows to determine which covariates are important and discriminant: the covariates' importance can be estimated by averaging over their importance in each of the b trees.

4.2 Boosting for Rankings

Boosting also repeatedly uses a base learning algorithm on differently weighted versions of the training data, leading to a sequence of classifiers that are finally combined. However, unlike bagging, which always gives identical weights to all data points, boosting creates weighted versions of the learning sample. The example weights are adaptively adjusted at each iteration so that previously misclassified items are sampled with a higher probability. The final predictions are obtained by weighting the results of the iteratively produced predictors based on their observed error rate. There are many versions of boosting algorithms, but arguably the most frequently used is AdaBoost.M1 (Freund & Schapire, 1996), which we will adapt to ranking data in the following.

Algorithm 2 AdaBoost.R - Boosting for ranking data.

Input: A training set T, a number of iterations B.
Output: a ranker $C_f(.)$ that maps a given x to a ranking of the labels
1 initialize $w_b(i) = 1/n \ \forall i = 1, 2, \ldots n$
2 **for** $b = 1$ **to** B **do**
3 take a sample T_b, drawn from the training set T using weights $w_b(i)$
4 fit a ranking tree $C_b(.)$ to T_b
5 $e_b = \sum_{i \in T_b} w_b(i)\left[1 - \frac{\tau_x(i)+1}{2}\right]$ where $\tau_x(i) = \tau_x(C_b(x_i), y_i))$
6 $\alpha_b = \frac{1}{2}\ln((1-e_b)/e_b)$
7 update the weights $w_{b+1}(i) = w_b(i)\exp(\alpha_b \tau_x(i))$ and normalize the
8 **end**
9 $C_f(x_i) = \arg\max_{y_i \in S^m} \sum_{b=1}^{B} \alpha_b \tau_x(C_b(x_i), y_i))$

The AdaBoost.R algorithm (Algorithm 2) is described as follows. A weight $w_1(i) = 1/n$ is initially assigned to each observation in the training set T of size n (step1). The weights w_b can be interpreted as probabilities for the corresponding example being included into the training set T_b for the bth iteration of the algorithm, in which a base classifier $C_b(.)$ is trained on T_b. The classifiers $C_b(.)$ are subsequently applied to each example in the training sample leading to a predicted ranking for each item $\tilde{y}_i^b = C_b(x_i)$ (steps 3 and 4). The ranking error e_b of the ranking tree $C_b(.)$ is estimated based on the distance between each predicted ranking $\tilde{y}_i^b$ and its real value y_i (step 5)

$$e_b = \sum_{i=1}^{n} w_b(i)\left[1 - \frac{\tau_x(\tilde{y}_i^b, y_i) + 1}{2}\right]. \tag{4}$$

An observation with a high value of $\left(1 - \frac{\tau_x(\tilde{y}_i^b, y_i)+1}{2}\right)$, i.e. a low value of $\tau_x(\tilde{y}_i^b, y_i)$, is poorly predicted.

Based on the errors e_b, a factor α_b is computed for updating the weights $w_b(i)$ (step 6). Following Freund and Schapire (1998),

$$\alpha_b = \ln((1-e_b)/e_b). \tag{5}$$

The weights $w_b(i)$ are updated after each iteration so that, at step $b+1$, the observations x_i that were misclassified by the classifier $C_b(x_i)$ have their weights increased, whereas the weights are decreased for those that were classified correctly. The new weight for the $(b+1)$th iteration (step 7) is

$$w_{b+1}(i) = w_b(i)\exp\alpha_b\tau_x(i), \tag{6}$$

normalized to sum to one.

This procedure ensures that the bigger the distance between the ranking associated with an observation and the original ranking, the higher is the probability that this observation is resampled in the new iteration (Alfaro et al., 2013). Thus, the sequence of trees tries to focus more on examples that are hard to predict. The α value can be interpreted as a local iteration-specific learning rate, calculated as a function of the error made in each iteration. Moreover, this value is also used in the final decision rule, giving more importance to the trees that made a lower error.

The iterative procedure continues until the stopping criterion (i.e. $\alpha_b > 0.5$) fires, or the maximum number of trees is reached. It is important to consider that fitting the training data too well can lead to overfitting, which increases the risk on future predictions (Hastie et al., 2009). We will return on this point in Section 5.

The importance of individual covariates can be estimated as a weighted average of their importance in each of the B trees, with weights α_b given by (5).

4.3 Rank Aggregation and Test Error Measurement

For making a final prediction, both, bagging and boosting, use rank aggregation to combine the predictions of the individual ranking trees. Recall that in each iteration b, we have obtained a tree $C_b(.)$, which predicts a ranking $\tilde{y}_i^b$ for a given example x_i.

In our experiments, we will evaluate our ensembles after each iteration, i.e. we have to compute an aggregated ranking $\hat{y}_{ib}$ and an aggregated error err(b) after each iteration b. The aggregated ranking for a generic ith observation at the bth iteration, $\hat{y}_{ib}$, will then be obtained as

$$\hat{y}_{ib} = \arg\max_{y_i \in S^m} \sum_{k=1}^{b} \alpha_k \tau_x(\tilde{y}_i^k, y_i), \qquad \text{with } b = 1, 2, \ldots, B, \tag{7}$$

where α_k, the weight related to the kth tree, is computed as in (5) for boosting, and $\alpha_k = 1$ in the case of bagging. The process is illustrated in Table 1, where the ith column shows the predictions, weights, and errors of the ith tree $C_i(.)$. The values in the last column correspond to the final lines of Algorithms 1 and 2.

Once each unit has been assigned a final ranking tree by tree, the error assigned to each sequence of trees $\{1, \ldots, b\}$ is computed as

$$\text{err}(b) = 1 - \frac{\tau_x(b) + 1}{2}, \qquad \text{with } b = 1, \ldots, B, \tag{8}$$

where $\tau_x(b) = \frac{1}{n}\sum_{i=1}^{n} \tau_x(\hat{y}_{ib}, y_i)$ is the average of τ_x of the bth tree over all the units in an example set T.

Before looking at the experimental results, we want to underline how and why our AdaBoost.R differs from Dery & Shmueli's (2020) BoostLR. First, the base classifier we use is the one introduced by Sciandra et al. (2015) and Plaia and Sciandra (2019), whereas

Table 1 Predictor matrix

Weights	α_1	α_2	...	α_b	...	...	α_B
Trees	$C_1(.)$	$C_2(.)$	...	$C_b(.)$	...	...	$C_B(.)$
1	$\hat{y}_{11}$	$\hat{y}_{11}$	...	$\hat{y}_{1b}$	...	...	$\hat{y}_{1B}$
2	$\hat{y}_{21}$	$\hat{y}_{22}$	...	$\hat{y}_{22}$	...	...	$\hat{y}_{2B}$
.	.	.	...	.	...	...	.
.	.	.	...	.	...	...	.
.	.	.	...	.	...	...	.
n	$\hat{y}_{n1}$	$\hat{y}_{n2}$	...	$\hat{y}_{nb}$	...	...	$\hat{y}_{nB}$
Error	err(1)	err(2)	...	err(b)	...	...	err(B)

Dery and Shmueli (2020) use LRT proposed by Hüllermeier et al. (2008). BoostLR uses a transformation of Kendall's τ_b coefficient to compute the "loss" (i.e. the factor multiplied by $w_b(i)$ in our (4)) for each training example: actually, the true transformation from a "correlation" to a distance is the one we use in our (4). Moreover, Emond and Mason (2002) have criticized τ_b because of its problematic way of handling ties — the all ties ranking has an undefined distance to any other ranking — and instead propose a refined version, τ_x (2), which fixes the problem assuming a different score matrix, with a'_{ij} equal to 1 (and not to 0) if the item i is tied with the item j.

Finally, Dery and Shmueli (2020) aggregate the weak models' outputs using weighted Borda, while, coherently with the previous steps in our algorithm, we use again a function of τ_x, now weighted with α_b, as shown in (7). Critiques to Borda's method can be found in Amodio et al. (2016).

5 Experimental Results

In order to evaluate the performance of the ensemble methods described in Section 4 we performed a simulation study (Section 5.1) and applied the method to three real datasets (Section 5.2).

5.1 Simulation Experiments

Following D'Ambrosio and Heiser (2016), we considered a predictor space (X_1, X_2), were X_1 and X_2 were generated according to continuous uniform distributions, with $X_1 \sim U(0, 10)$ and $X_2 \sim U(0, 6)$, which was partitioned into five regions as shown in Fig. 1. The number of datapoints (of 4 items) within each sub-partition was determined by: i) randomly drawing from a normal distribution $N(10, 2)$, ii) dividing them by their summation, iii) multiplying by the true sample size ($n = 200,\ n = 500$).

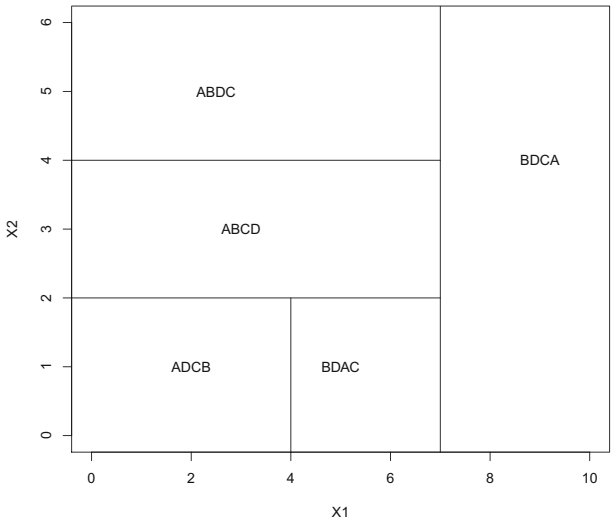

Fig. 1 Theoretical partition of the predictor space (X_1, X_2) with 4 items

The rankings (datapoints) within each sub-partition were generated from a Mallows Model (Mallows, 1957), which is one of the first probability models proposed for rankings and frequently used in both theoretical and applied studies. It is an exponential model defined by a central permutation σ_0 and a dispersion parameter θ. When $\theta > 0$, σ_0 represents the mode of the distribution, i.e. the permutation that has the highest probability. The σ_0 values for our simulation studies are shown in Fig. 1. The probability of any other ranking decays exponentially with increasing distance to the central permutation. The dispersion parameter controls the steepness of this decline. Assuming that σ is a generic ranking, the probability for this ranking is given by

$$\Pr(\sigma) = \frac{\exp(-\theta d(\sigma, \sigma_0^{-1}))}{\psi(\theta)},$$

where d is a ranking distance measure and $\psi(\theta)$ is a normalization constant. In our simulation, we generated rankings assuming the Kemeny distance and varying the dispersion parameter θ, according to three different level of noise (*low* with $\theta = 2$, *medium* with $\theta = 0.7$ and *high* with $\theta = 0.4$). Considering two levels for the sample size ($n = 200, n = 500$) the experimental design, hence, counts $3 \times 2 = 6$ different scenarios. Figure 2 shows one of the six datasets considered in our experiment (corresponding to $\theta = 2$ and $n = 500$).

We applied the two ensemble methods defined in Section 4 to all six scenarios, fixing the number of trees B to 100, the depth of each tree to 4 (see Fig. 4 for different depths) and considering a training sample T of 2/3 of the observations. R code (R Core Team, 2020) was used for both the simulations and the application to real data, by opportunely modifying available functions in the R packages Consrank (D'Ambrosio et al., 2015b) and adabag (Alfaro et al., 2013).

Figure 3 compares boosting and bagging performances in all the simulated scenarios, plotting the error err(b) (8) vs. the number of trees b, both in the training and in the test set.

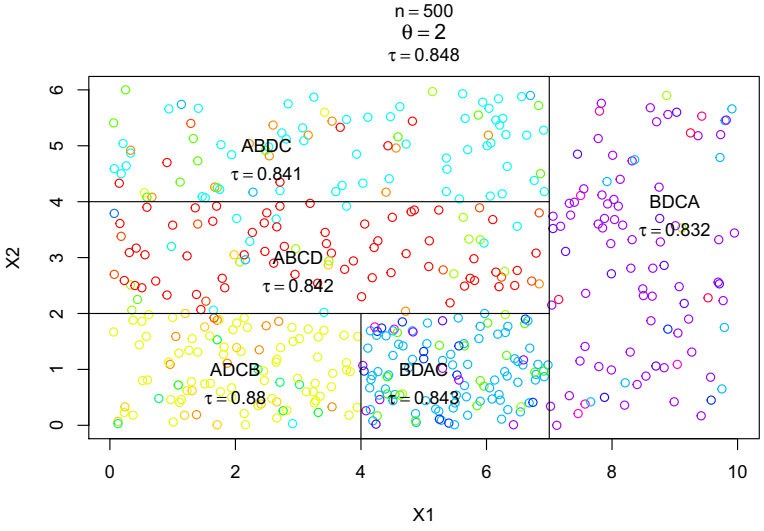

Fig. 2 Empirical partition of the predictor space, generating high homogeneous groups of rankings ($\theta = 2$), with $n = 500$. A specific colour is assigned to the generated rankings. For instance, yellow is the colour associated with the modal ranking ADCB of the bottom-left sub-partition but, because of the heterogeneity among rankings, other colours, hence other rankings, are present, too

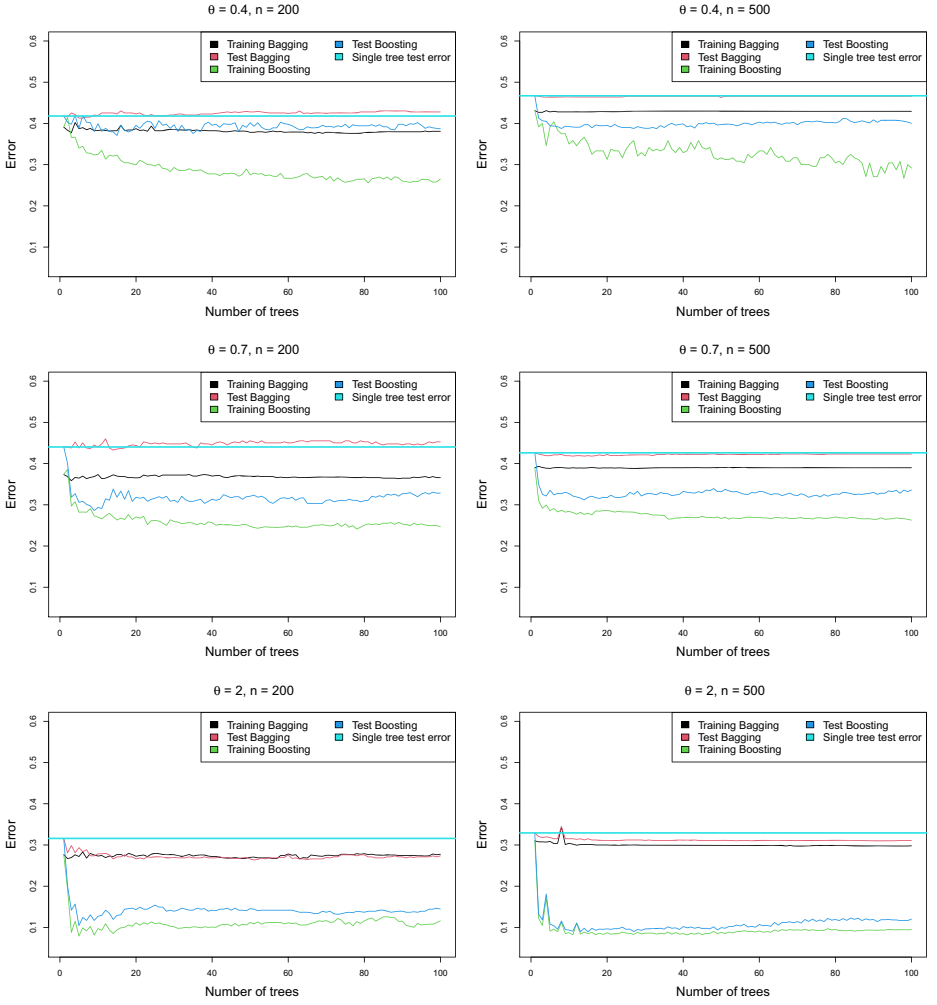

Fig. 3 Bagging and boosting for all the simulated scenarios with 100 trees: different levels of homogeneity among the rankings, $\theta = (0.4, 0.7, 2)$, and two sample sizes, $n = (200, 500)$

The starting point of each graph is the error corresponding to the first tree (x-axis), therefore the plot shows the improvement by the bagging/boosting algorithms computed both on the training (2/3 of the observations) and on the test (1/3 of the observations) samples.

Looking at the errors produced by boosting, the methodology is able to perform very well when there is a high level of heterogeneity among the rankings ($\theta = 0.4$). Expectedly the performances for $n = 200$ are worse than for $n = 500$. The training error and the test error for $n = 500$ seem to converge after approximately 20 trees. In particular, it was enough stopping the procedure at $B = 100$ trees, since the errors showed to be quite stable. Stopping at $B = 100$ trees we also avoid overfitting, that, as known, can affect boosting: looking at the top-right plot in Fig. 3, for example, we can observe that the boosting training error (green line) keeps going down, while the corresponding test error (blue line) starts to increase.

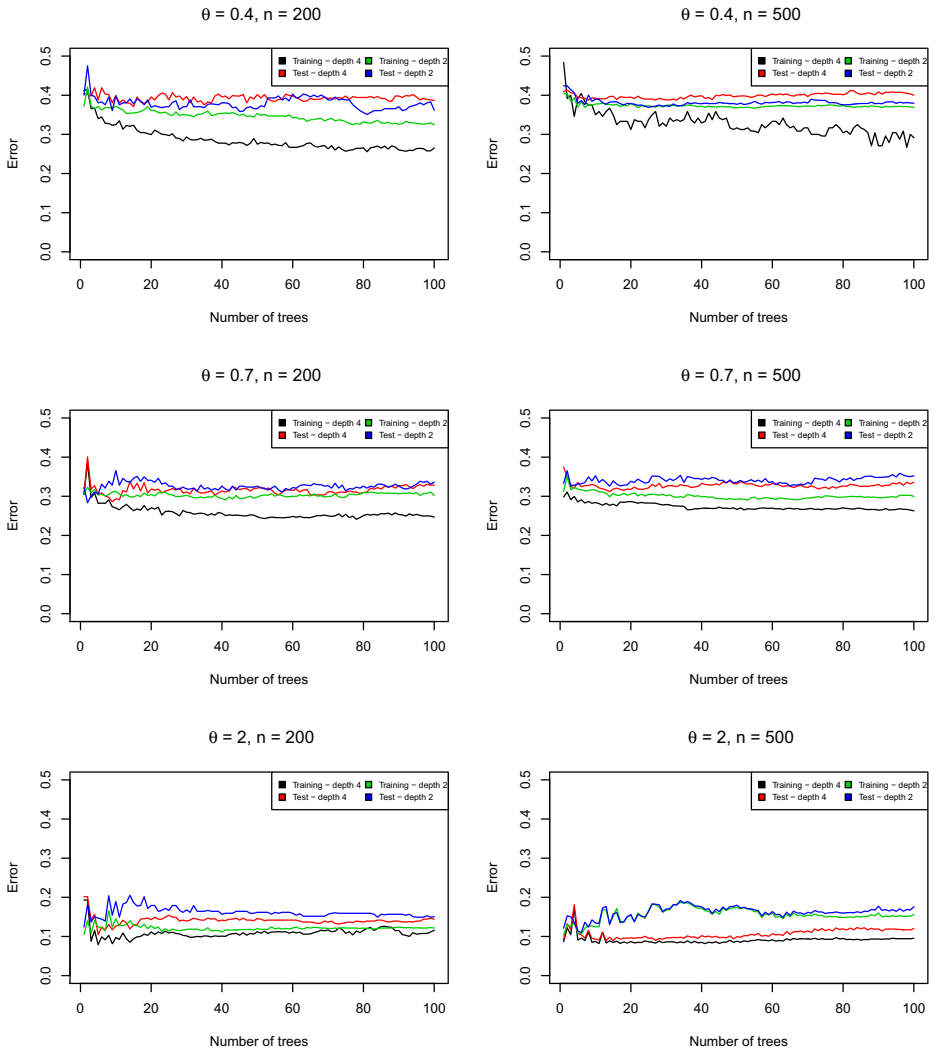

Fig. 4 Boosting built up for each simulated dataset, using two levels of splitting for the trees (2 and 4)

Finally, we also see that bagging always performs worse than boosting. It is important to highlight that, even if a depth-4 tree could seem too small for a Bagging procedure, due to the cardinality of the universe of all possible rankings with ties, even a small tree (depth 4, i.e. 5 leaves) causes a non-negligible variability in the predicted rankings $\tilde{y}_i^b$ along with the trees. Just as an example, considering $\theta = 0.7$ and $n = 200$ (i.e. left-middle plot in figure 3, which considers only depth 4 trees), the τ_x computed on the $B = 100$ predicted values for each of the $n = 200$ units

$$\tau_x(i) = \frac{1}{B}\sum_{b=1}^{B}\tau_x(\tilde{y}_{ib}, y_i)$$

ranges from 0.55 to 0.86. This means that even using small tree produces high variability.

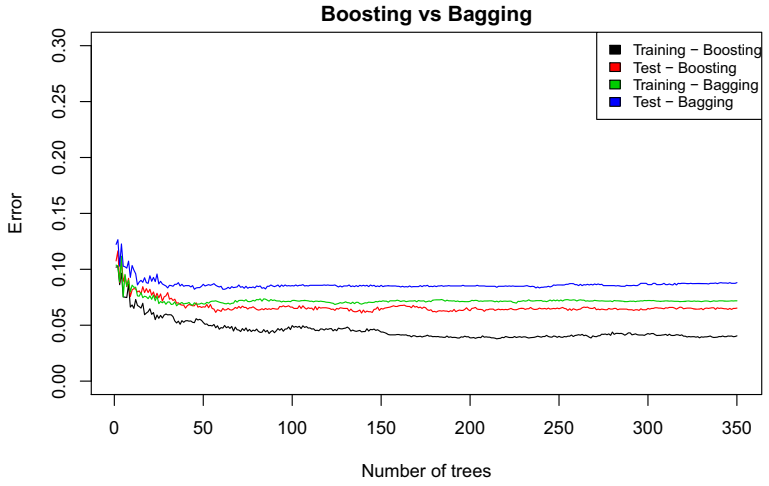

Fig. 5 Boosting and bagging applied to Vehicle dataset

In order to verify if the procedure is sensitive to the number of splits in each tree, i.e. to its depth, we consider boosting with two different tree depths (2 and 4 respectively). The results are shown in Fig. 4. It becomes clearer, going from $\theta = 0.4$ to $\theta = 2$, that the mean error computed using depth 4 (both in the training and in test sets) tends to be lower than that with depth 2. These results confirm the ones by Zhou and Mentch (2021), who found that depending on the signal-to-noise ratio of the underlying data, having smaller trees can be more beneficial to the ensemble predictions.

5.2 Real Data Applications

The first dataset is a multi-class data set (Fig. 5) from the UCI repository[1]. The dataset "vehicle" originally contains 946 units, 18 explanatory variables and four items useful for classification tasks. As the original dataset is only for classification, we followed the procedure proposed by Hüllermeier et al. (2008) to turn it into a ranking dataset: first a Naïve Bayes classifier is trained to estimate the probabilities for each class label, and then this predictd ranking is used as the target ranking for the ranking prediction task. We randomly split the data in $2/3$ training and $1/3$ test instances and learned trees with a maximum depth of 4.

Figure 5 compares the outcomes obtained applying boosting and bagging to the data for up to 350 trees. Again, boosting shows a better performance, while the error becomes stable after about 100 trees.

We also consider two more real datasets, which have also been used in previous works (de Sá et al., 2018; Werbin-Ofir et al., 2019; Dery & Shmueli, 2020). The datasets GermanElections2005 and GermanElections2009 contain socio-economic information from regions of Germany and their respective electoral results. The 413 records correspond to the administrative districts of Germany, which are described by 39 covariates, such as age and education of the population, economic indicators (e.g. GDP growth, percentage of unemployment),

[1] https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/vehicle/

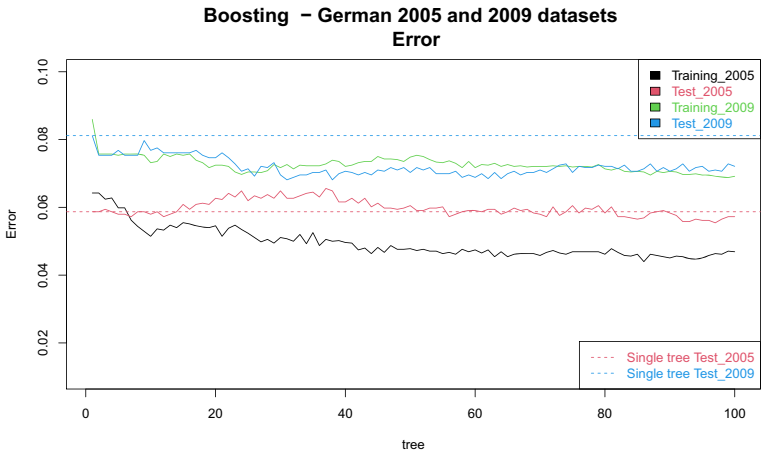

Fig. 6 Boosting applied to German Elections datasets: err(b)

indicators of the labour workforce in different sectors such as production, public service, etc. In terms of the outcome, de Sá et al. (2018) transformed the election results of the five major political parties for the federal elections in 2005 and 2009 into rankings of five items: CDU (conservative), SPD (centre-left), FDP (liberal), Green (centre-left) and Left (left-wing). Because of the better performance of boosting over bagging, observed both in the simulation study and in the previous example, we consider only boosting, limiting the number of trees B to 100 with a maximum depth of 4.

Figure 6 shows how the error varies — starting from the first tree — in both the datasets: it is interesting to observe that in 1 case, after about 15 iterations, the test error increases, i.e. on the 2005 data we seem to have a strong problem with overfitting.

We also measured $\tau_x(b) = \frac{1}{n}\sum_{i=1}^{n} \tau_x(\hat{y}_{ib}, y_i)$ for all the trees: Fig. 7 shows the average of τ_x of the bth tree over all the units in an example set T (both training or test): as expected, $\tau_x(b)$ performance is specular to err(b)'s one.

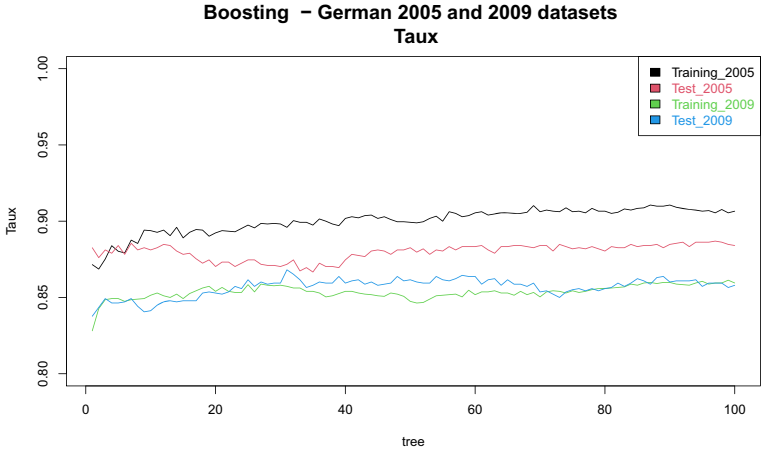

Fig. 7 Boosting applied to German Elections datasets: average τ_x

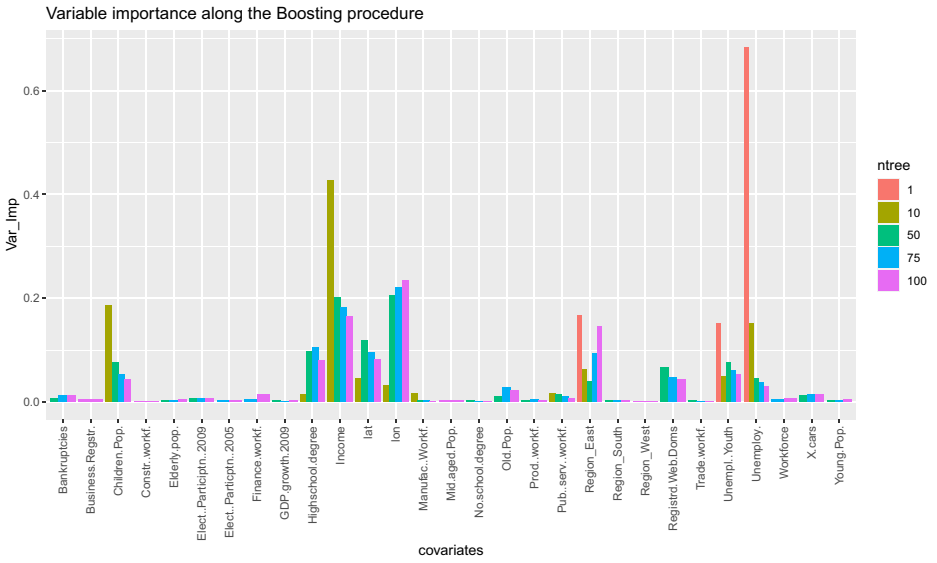

Fig. 8 Variable Importance at intermediate and final step for 2009 German Elections dataset

As explained in Section 4.2, the procedure allows to determine which covariates are important and discriminant: covariates' importance can be computed averaging over their importance resulting in each of the B trees, with weights α given by (5). Figure 8 shows the variable relative importance at some steps in the procedure (i.e. after 1, 10, 50, 75 and 100 trees). It is interesting to notice how the relative importance of each variable can change along the iterations: for example the importance of the variable Unemploy, the most important covariate at the beginning, has substantially decreased at the end of the process (i.e. at 100 trees).

6 Conclusion

In this paper, we adapted two ensemble methodologies, boosting and bagging, to ranking data to improve the prediction performance of single decision trees. Both algorithms combine the predictions of multiple decision trees, in particular the distance-based trees for ranking data proposed by Plaia and Sciandra (2019). These trees consider the Kemeny distance (Kemeny & Snell, 1962) as a measure of impurity in the splitting process, and its related rank correlation coefficient τ_x (proposed by Emond & Mason (2002)) for identifying the median ranking in the final nodes. As stated by Zhou and Qiu (2018), rank aggregation can be obtained by optimizing different rank distance measures, but it has been shown that the Kemeny optimal aggregation is the best compromise ranking. However, finding a Kemeny optimal aggregation is NP-hard even with four items, therefore, we use two algorithms QUICK and FAST (D'Ambrosio et al., 2015a; Amodio et al., 2016) for quickly approximating the median ranking. The same rank correlation coefficient τ_x is used to compute the final aggregated ranking over the predictions of every single tree. τ_x also represents

the main ingredient both in computing the error err(b) associated with each tree and the factor α_b (only in the boosting algorithm) to update the weights of each item in the bootstrap procedure.

We implemented both algorithms in R, and applied them to simulated data and real cases. The simulation study shows that boosting can perform very well when there is a high level of heterogeneity ($\theta = 0.4$) among the rankings, and its performance improves with the sample size. Moreover, depth-4 boosting outperforms depth-2 boosting with a high level of heterogeneity ($\theta = 0.4$), while they behave similarly with a low level of heterogeneity ($\theta = 2$). Bagging performs worse than boosting in all the simulated scenarios, and its error decreases when the heterogeneity decreases, independently on the sample size n. The results on real datasets were in line with the results of the simulation study. Both algorithms also allow to determine which covariates are significant and discriminant, by averaging over their importance in each of the b trees, with weights α_b in case of boosting.

An interesting research direction would be to improve the scalability of the algorithms with respect to the number of items. And in this direction, the introduction of weights, both to give different importance to some positions—say the top-positions—or to items—some items could be more relevant than others—could help (García-Lapresta & Pérez-Román, 2010; Kumar & Vassilvitskii, 2010; Can, 2014; Plaia et al., 2021): both these types of weights could help reduce the size of the universe of rankings to explore to find the consensus, the most computer intensive step of the proposed procedures. Indeed, the distance-based trees for ranking data proposed by Plaia and Sciandra (2019) already implement position weights, even if not considered in the algorithms presented in this paper, and we are now studying how item weights can be considered. Both solutions could be then considered as base decision trees within the bagging and boosting algorithms proposed in this paper.

Funding Open access funding provided by Università degli Studi di Palermo within the CRUI-CARE Agreement.

References

Aledo, J.A., Gámez, J.A., & Molina, D (2017). Tackling the supervised label ranking problem by bagging weak learners. *Information Fusion*, *35*, 38–50.

Alfaro, E., Gámez, M., & García, N (2013). Adabag: An R package for classification with boosting and bagging. *Journal of Statistical Software*, *54*(2), 1–35.

Amodio, S., D'Ambrosio, A., & Siciliano, R (2016). Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the Kemeny axiomatic approach. *European Journal of Operational Research*, *249*(2), 667–676.

Austin, P.C. (2012). Using ensemble-based methods for directly estimating causal effects: an investigation of tree-based g-computation. *Multivariate Behavioral Research*, *47*(1), 115–135.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140.

Breiman, L. (1998). Arcing classifier (with discussion and a rejoinder by the author). *The Annals of Statistics*, *26*(3), 801–849.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.
Breiman, L., Friedman, J.H., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Wadsworth & Brooks: Pacific Grove.
Bühlmann, P. (2003). Bagging, subagging and bragging for improving some prediction algorithms. Tech. rep., ETH Zurich, Switzerland. ftp://ftp.stat.math.ethz.ch/Research-Reports/113.pdf.
Bühlmann, P., & Yu, B. (2000). Explaining bagging. Tech. rep., ETH Zurich, Switzerland. ftp://ess.r-project.org/pub/Research-Reports/92.pdf.
Bühlmann, P., & Yu, B. (2002). Analyzing bagging. *The Annals of Statistics*, *30*(4), 927–961.
Buja, A., & Stuetzle, W. (2006). Observations on bagging. *Statistica Sinica*, *13*, 323–351.
Can, B. (2014). Weighted distances between preferences. *Journal of Mathematical Economics*, *51*, 109–115.
Cheng, W., Huhn, J., & Hüllermeier, E (2009). Decision tree and instance-based learning for label ranking. In *Proceedings of the 26th International Conference On Machine Learning, ICML 2009, vol 382, p 21. https://www.informatik.uni-marburg.de/eyke/publications/icml09.pdf*.
Cook, W.D., Kress, M., & Seiford, LM (1986). An axiomatic approach to distance on partial orderings. *RAIRO-Operations Research*, *20*(2), 115–122.
D'Ambrosio, A. (2008). *Tree-based methods for data editing and preference rankings*. Italy: PhD thesis, Department of Mathematics and Statistics, Naples. http://www.fedoa.unina.it/2746/.
D'Ambrosio, A., & Heiser, W.J. (2016). A recursive partitioning method for the prediction of preference rankings based upon Kemeny distances. *Psychometrika*, *81*(3), 774–794.
D'Ambrosio, A., Amodio, S., & Iorio, C (2015a). Two algorithms for finding optimal solutions of the Kemeny rank aggregation problem for full rankings. *Electronic Journal of Applied Statistical Analysis*, *8*(2), 198–213.
D'Ambrosio, A., Amodio, S., & Mazzeo, G (2015b). Consrank: compute the median ranking(s) according to the Kemeny's axiomatic approach. r package version 1.0.2.
D'Ambrosio, A., Mazzeo, G., Iorio, C., & Siciliano, R (2017). A differential evolution algorithm for finding the median ranking under the Kemeny axiomatic approach. *Computers & Operations Research*, *82*, 126–138.
David, H.A. (1969). *The method of paired comparisons (2 ed.), volume 12 of Griffin's Statistical Monographs & Courses*. London: Charles Griffin & Company Limited.
Dery, L., & Shmueli, E. (2020). BoostLR: A boosting-based learning ensemble for label ranking tasks. *IEEE Access*, *8*, 176023–176032.
Dietterich, T.G. (2000). Ensemble methods in machine learning. In J. Kittler, & F. Roli (Eds.) *Multiple Classifier Systems, Springer-Verlag, pp 1–15. ftp://ftp.cs.orst.edu/pub/tgd/papers/mcs-ensembles.ps.gz*.
Efron, B. (1982). The Jackknife, the Bootstrap, and Other Resampling Plans. Society for Industrial and Applied Mathematics.
Emond, E.J., & Mason, D.W. (2002). A new rank correlation coefficient with application to the consensus ranking problem. *Journal of Multi-Criteria Decision Analysis*, *11*(1), 17–28.
Freund, Y., & Schapire, R.E. (1996). Experiments with a new boosting algorithm. In L. Saitta (Ed.) *Proceedings of the 13th International Conference on Machine Learning, Morgan Kaufmann, Bari, Italy, pp 148–156. https://cseweb.ucsd.edu/yfreund/papers/boostingexperiments.pdf*.
Freund, Y., & Schapire, R.E. (1998). Discussion: Arcing classifiers. *The Annals of Statistics*, *26*(3), 824–832.
Freund, Y., Iyer, R.D., Schapire, R.E., & Singer, Y (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, *4*, 933–969.
Friedman, J.H., & Hall, P. (2007). On bagging and nonlinear estimation. *Journal of Statistical Planning and Inference*, *137*(3), 669–683.
In J. Fürnkranz, & E. Hüllermeier (Eds.) (2011). *Preference Learning*. Berlin: Springer-Verlag.
García-Lapresta, J.L., & Pérez-Román, D. (2010). Consensus measures generated by weighted Kemeny distances on weak orders. In *2010 10th International Conference on Intelligent Systems Design and Applications, IEEE, pp 463–468. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5687220*.
Good, I. (1980). C59. the number of orderings of n candidates when ties and omissions are both allowed. *Journal of Statistical Computation and Simulation*, *10*(2), 159.
Grimm, K.J., & Jacobucci, R. (2020). Reliable trees: Reliability informed recursive partitioning for psychological data. *Multivariate Behavioral Research*, *0*(0), 1–13.
Hall, P., & Schimek, M. (2012). Moderate-deviation-based inference for random degeneration in paired rank lists. *Journal of the American Statistical Association*, *107*(498), 661–672.
Hastie, T., Tibshirani, R., & Friedman, JH. (2009). *The Elements of Statistical Learning*. Berlin: Springer-Verlag.
Heiser, W.J., & D'Ambrosio, A. (2013). Clustering and prediction of rankings within a Kemeny distance framework. In B. Lausen, & D. UA Van den Poel (Eds.) *Algorithms from and for nature and life, springer international publishing switzerland* (pp. 19–31).

Hüllermeier, E., Fürnkranz, J., Cheng, W., & Brinker, K (2008). Label ranking by learning pairwise preferences. *Artificial Intelligence*, *172*(16-17), 1897–1916.
James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning: with applications in R*. New York: NY.
Kemeny, J.G., & Snell, L. (1962). Preference ranking: an axiomatic approach. In Mathematical Models in the Social Sciences, Ginn, New York, 9–23.
Kumar, R., & Vassilvitskii, S. (2010). Generalized distances between rankings. In *Proceedings of the 19th international conference on World wide web, ACM, pp 571–580. http://theory.stanford.edu/sergei/papers/www10-metrics.pdf*.
Lee, P.H., & Yu, P.LH. (2010). Distance-based tree models for ranking data. *Computational Statistics & Data Analysis*, *54*(6), 1672–1682.
Mallows, C.L. (1957). Non-null ranking models. *Biometrika*, *44*(1–2), 114–130.
Marcus, P. (2013). Comparison of heterogeneous probability models for ranking data. Master's thesis, Leiden University. http://www.math.leidenuniv.nl/scripties/1MasterMarcus.pdf.
Maydeu-Olivares, A., & Bockenholt, U. (2005). Structural equation modeling of paired-comparison and ranking data. *Psychological Methods*, *10*(3), 285–304.
Maydeu-Olivares, A., & Bockenholt, U. (2009). Modeling preference data. In R.E. Millsap, & A. Maydeu-Olivares (Eds.) *The SAGE Handbook of Quantitative Methods in Psychology. Sage Publications Ltd, pp 264–282*.
Müllensiefen, D., Hennig, C., & Howells, H (2018). Using clustering of rankings to explain brand preferences with personality and socio-demographic variables. *Journal of Applied Statistics*, *45*(6), 1009–1029.
Murthy, S.K. (1998). Automatic construction of decision trees from data: a multi-disciplinary survey. *Data Mining and Knowledge Discovery*, *2*(4), 345–389.
Piccarreta, R. (2010). Binary trees for dissimilarity data. *Computational Statistics & Data Analysis*, *54*(6), 1516–1524.
Plaia, A., & Sciandra, M. (2019). Weighted distance-based trees for ranking data. *Advances in Data Analysis and Classification*, *13*, 427–424.
Plaia, A., Buscemi, S., & Sciandra, M (2021). Consensus among preference rankings: a new weighted correlation coefficient for linear and weak orderings. Advances in Data Analysis and Classification https://doi.org/10.1007/s11634-021-00442-x.
R Core Team (2020). R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.
Rossi, F., Venable, K.B., & Walsh, T (2011). A short introduction to preferences: Between artificial intelligence and social choice. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *5*(4), 1–102.
Rounds, J.B Jr., Miller, T.W., & Dawis, RV (1978). Comparability of multiple rank order and paired comparison methods. *Applied Psychological Measurement*, *2*(3), 415–422.
de Sá, C.R., Rebelo, C., Soares, C., Knobbe, A., & Cortez, P (2015). Distance-based decision tree algorithms for label ranking. In *Progress in Artificial Intelligence – 17th Portuguese Conference on Artificial Intelligence (EPIA), pp 525–534. https://repositorio.inesctec.pt/bitstream/123456789/3924/1/P-00G-SXT.pdf*.
de Sá, C.R., Soares, C., Knobbe, A., & Cortez, P (2017). Label ranking forests. *Expert Systems*, *34*(1), 1–8.
de Sá, C.R., Duivesteijn, W., Azevedo, P., Jorge, A.M., Soares, C., & Knobbe, A (2018). Discovering a taste for the unusual: exceptional models for preference mining. *Machine Learning*, *107*, 1775–1807.
Sampath, S., & Verducci, J. (2013). Detecting the end of agreement between two long ranked lists. *The ASA Data Science Journal*, *6*(6), 458–471.
Sciandra, M., Plaia, A., & Picone, V (2015). Recursive partitioning: an approach based on the weighted Kemeny distance. In *Proceedings of the 10th scientific meeting of the classification and data analysis group of the italian statistical society (CLADAG 2015), CUEC Editrice, pp 494–497. https://meetings3.sis-statistica.org/index.php/sis2018/50th/paper/viewFile/1377/163*.
Stegmann, G., Jacobucci, R., Serang, S., & Grimm, KJ (2018). Recursive partitioning with nonlinear models of change. *Multivariate Behavioral Research*, *53*(4), 559–570.
Svendova, V., & Schimek, M. (2017). A novel method for estimating the common signals for consensus across multiple ranked lists. *Computational Statistics & Data Analysis*, *115*, 122–135.
Vembu, S., & Gärtner, T. (2010). Label ranking algorithms: a survey. In J. Fürnkranz, & E. Hüllermeier (Eds.) *Preference Learning* (pp. 45-64). Berlin: Springer-Verlag.
Werbin-Ofir, H., Dery, L., & Shmueli, E (2019). Beyond majority: Label ranking ensembles based on voting rules. *Expert Systems with Applications*, *136*, 50–61.
Yu, P.LH., Wan, W.M., & Lee, PH In Fürnkranz j, & E. Hüllermeier (Eds.) (2010). *Decision tree modeling for ranking data*. Preference Learning: Springer-Verlag Berlin Heidelberg.

Zhou, S., & Mentch, L. (2021). Trees, forests, chickens, and eggs: When and why to prune trees in a random forest. arXiv:210316700v1.
Zhou, Y., & Qiu, G. (2018). Random forest for label ranking. *Expert Systems with Applications*, *112*, 99–109.
Zhou, Y., Liu, Y., Yang, J., He, X., & Liu, L (2014). A taxonomy of label ranking algorithms. *Journal of Computers*, *9*(3), 557–565.