Rank Aggregation for Non-stationary Data Streams

Ekhine Irurozki[1,2(✉)], Aritz Perez[2(✉)], Jesus Lobo[3(✉)], and Javier Del Ser[3,4(✉)]

[1] Telecom Paris, Institut Polytechnique de Paris, Palaiseau, France
irurozki@telecom-paris.fr
[2] Basque Center for Applied Mathematics (BCAM), Bilbao, Spain
aperez@bcamath.org
[3] TECNALIA, Basque Research and Technology Alliance (BRTA), Bizkaia, Spain
{Jesus.Lopez,javier.delser}@tecnalia.com
[4] University of the Basque Country (UPV/EHU), Bilbao, Spain

Abstract. The problem of learning over non-stationary ranking streams arises naturally, particularly in recommender systems. The rankings represent the preferences of a population, and the non-stationarity means that the distribution of preferences changes over time. We propose an algorithm that learns the current distribution of ranking in an online manner. The bottleneck of this process is a rank aggregation problem.

We propose a generalization of the Borda algorithm for non-stationary ranking streams. As a main result, we bound the minimum number of samples required to output the ground truth with high probability. Besides, we show how the optimal parameters are set. Then, we generalize the whole family of weighted voting rules (the family to which Borda belongs) to situations in which some rankings are more *reliable* than others. We show that, under mild assumptions, this generalization can solve the problem of rank aggregation over non-stationary data streams.

Keywords: Preference learning · Rank aggregation · Borda · Evolving preferences · Voting · Concept drift

1 Introduction

In the rank aggregation problem, a population of voters cast their preferences over a set of candidates by providing each a ranking of these candidates. The goal is to summarize this collection of rankings S with one single ranking π. A classical formulation of rank aggregation is to find the Kemeny ranking, the ranking that minimizes the number of discrepancies with S. Unfortunately, finding the Kemeny ranking is NP-hard.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-86523-8_18) contains supplementary material, which is available to authorized users.

N. Oliver et al. (Eds.): ECML PKDD 2021, LNAI 12977, pp. 297–313, 2021.
https://doi.org/10.1007/978-3-030-86523-8_18

When the rankings are distributed according to the Mallows distribution Kemeny is known to be the maximum likelihood estimator of one model parameter and the Borda ranking a high-quality, low-cost approximation to the same parameter. Therefore, besides in Optimization [8], rank aggregation has been studied in Computational Social Choice [4] and Machine Learning [19,26] extensively, generally in a batch setting.

In this paper, we challenge the standard batch learning setting and consider stream learning [13,18], which refers to learning problems for which the data is continuously generated over time. This feature imposes severe computational complexity constraints on the proposed learning algorithms. For instance, in contrast to batch learning, in stream learning scenarios the data can no longer be completely stored, and the learning algorithms must be computationally efficient (linear or even sub-linear), to the extreme of operating close to real-time. These computational complexity constraints usually lead to models that are incrementally updated with the arrival of new data.

Besides, another major challenge when dealing with stream learning arises when the source producing stream data evolves over time, which yields to non-stationary data distributions. This phenomenon is known as concept drift (CD) [14]. In the stream learning scenarios with CD, the models have to be able to adapt to changes (drifts) in the data distribution (concept). We focus on this scenario that we call *evolving preferences*.

Problem Statement. In this paper, we assume that the data is given as a stream of rankings over n items representing the evolving preferences of a population of voters. Rankings are i.i.d. according to a Mallows model (MM) -analogous to the Gaussian distribution defined over the permutation space-: MM are parametrized by (i) a modal ranking (representing the consensus of the population), and (ii) a parameter controlling the variance. Therefore, a stream of rankings with concept drift is naturally modelled by a sequence of MM with different modal rankings, i.e., an evolving MM. The goal is to develop an estimate of the current modal ranking of the evolving MM with low computational complexity and arbitrarily small error, that is able to deal with the CD phenomenon.

Contributions. There are three main contributions in this paper.

- We adapt the Borda algorithm to the stream learning scenario in which the distribution of the data changes, SL with CD. We denote this generalization of Borda unbalanced Borda (uBorda).
- We theoretically analyze uBorda and provide bounds on the number of samples required for recovering the last modal ranking of the evolving MM, in expectation and with arbitrary probability. Moreover, we show how to set an optimal learning parameter.

– We generalize whole families of voting rules[1] to manage evolving preferences. Moreover, we show that this is a particular case of the scenario in which some voters are more *trusted* than others. We denote this setting as *unbalanced voting* and show the empirical performance of this approach.

Related Work. Rank aggregation has been studied in Optimization [8], Computational Social Choice [4] and Machine Learning [19,26]. In the last decade, rank aggregation has been studied in on-line environments. In particular, [27] proposed a ranking reconstruction algorithm based on the pairwise comparisons of the items with near optimal relative loss bound. In [1], two efficient algorithms based on sorting procedures that predict the aggregated ranking with bounding maximal expected regret where introduced. In [6] an active learner was proposed that, querying items pair-wisely, returns the aggregate ranking with high probability.

Evolving preferences have been considered from an axiomatic perspective by modeling voters preferences with Markov decision processes [21], in fairness [12] and matching [15]. In [23], the authors analyze the behavior of the preferences of a population of voters under the plurality rule. In [22] the authors assume that each user preferences change over time. In contrast to the previous works, in this work we assume a probabilistic setting using a MM that changes over the time for modeling the evolving preferences of a population.

This paper is organized as follows: Sect. 2 includes the preliminaries of this paper. Section 3 presents the generalization of Borda and its theoretical guarantees. Section 4 introduces the adaptation of a whole family of voting rules to the case in which some voters are trusted more. Section 5 shows the empirical evaluation of the algorithms for the rank aggregation on stream learning with concept drift. Finally, Sect. 6 summarizes the main contributions of the work.

2 Preliminaries and Notation

Permutations are a bijection of the set $[n] = \{1, 2, \ldots, n\}$ onto itself and are represented as an ordered vector of $[n]$. The set of permutations of $[n]$ is denoted by S_n. Permutations will be denoted with the Greek letters σ or π and represent rankings, meaning that the ranking of element $i \in [n]$ in σ is denoted by $\sigma(i)$. Rankings are used to represent preferences and we say that item i is preferred to item j when it has a lower ranking, $\sigma(i) < \sigma(j)$.

One of the most popular noisy models for permutations is the Mallows model (MM) [10,20]. The MM is an exponential location-scale model in which the location parameter (modal ranking) is a ranking denoted by π, and the scale

[1] Voting rules are functions that given a collections of rankings output the winner of the election and generalize the notion of rank aggregation. We generalize the Weighted Tournament solutions and Unweighted Tournament solutions families, from which Borda and Condorcet respectively are the best known members.

(or concentration parameter) is a non-negative real number denoted by θ. In general, the probability of any ranking $\sigma \in S_n$ under the MM can be written as:

$$p(\sigma) = \frac{\exp(-\theta \cdot d(\sigma, \pi))}{\psi(\theta)}, \tag{1}$$

where $d(\cdot, \cdot)$ is a distance metric and $\psi(\theta)$ is a normalization factor that depends on the location parameter θ. It has closed form expression for most interesting distances [16].

A random ranking σ drawn from a MM centered at π and with concentration parameter θ is denoted as $\sigma \sim MM(\pi, \theta)$. The expected rank of item i for $\sigma \sim MM(\pi, \theta)$ is denoted by $\mathbb{E}_\pi[\sigma(i)]$, or simply by $\mathbb{E}[\sigma(i)]$ when it is clear from the context. To model preferences, the selected distance $d(\cdot, \cdot)$ for permutations is the Kendall's-τ distance (for other choice of distances see [16,25]). This distance counts the number of pairwise disagreements in two rankings,

$$d(\sigma, \pi) = \sum_{i<j} \mathbb{I}[(\sigma(i) - \sigma(j)) * (\pi(i) - \pi(j)) < 0], \tag{2}$$

where $\mathbb{I}$ is the indicator function.

Given a sample of rankings, the maximum likelihood estimation (MLE) of the parameters of a MM is usually computed in two steps. First, the MLE of the modal ranking π is obtained. It corresponds to the Kemeny ranking [10] (see the following sections for further details). Second, given the MLE of the modal ranking, the MLE of the concentration parameter θ can be computed numerically.

2.1 Modeling Evolving Preferences: Evolving Mallows Model

In this paper, we assume that the data is a stream of preferences, i.e., a sequence of rankings and that our proposed algorithm has to update its estimate every time a new ranking is received. The rankings in the sequence may not all come from the same distribution, so in the next lines, we introduce a natural way of modeling preferences that evolve dynamically over time.

Let σ_t for $t \geq 0$ be the $t-th$ ranking in the given sequence, being σ_0 the current ranking. We assume that the stream of rankings satisfy $\sigma_t \sim MM_t(\pi_t, \theta_t)$, for $t \geq 0$. This sequence of models is denoted the evolving Mallows model (EMM).

When two consecutive models, $MM_{t+1}(\pi_{t+1}, \theta_{t+1})$ and $MM_t(\pi_t, \theta_t)$, differ on their modal ranking, $\pi_{t+1} \neq \pi_t$, we say that at time t there has been a drift in the modal ranking of the model. Under the Mallows models, the drifts in the modal ranking can be interpreted as a change in the consensus preference of a population of voters. When the concentration parameter changes over time, the drifts represent a change in the variability of the preferences of a population around the consensus ranking. For the rest of the paper, we focus on the scenario with drifts in the modal ranking.

Drifts can be classified as gradual and abrupt in terms of speed, being abrupt when a change happens suddenly between two concepts, and gradual when there is a smooth transition between both concepts. In this work, we have considered abrupt drifts in what refers to the speed of change (rankings generated from the old concept disappear suddenly and the new ones appear), and small-step/large-step (the Kendall's-τ distance between consecutive modal rankings is small/large).

3 Unbalanced Borda for Stream Ranking

In this section, we present a generalization of the well-known Borda count algorithm to rank aggregation for the context of stream data with concept drift. We want to point out that this approach is generalized in Sect. 4, for (1) the settings in which each ranking in the sample is not equally relevant (e.g., the agents have different reliabilities), and (2) for all the voting rules in the family of Weighted voting rules and Unweighted Voting rules (C1 and C2 families) [4].

The Borda algorithm [3] is one of the most popular methods for aggregating a sample of rankings, S, into one single ranking that best represents S. Borda ranks the items $[n]$ by their *Borda score* increasingly, where the Borda score $B(i)$ is the average ranking of each item i, for $i = 1, \ldots, n$:

$$B(i) = \frac{1}{|S|} \sum_{t \in S} \sigma_t(i).$$

Borda is the de-facto standard in the applied literature of rank aggregation. Firstly, it has a computational complexity of $\mathcal{O}(n \cdot (|S| + \log n))$. Secondly, it is guaranteed to be a good estimator of the modal ranking when the samples are i.i.d. according to a MM: it requires a polynomial number of samples with respect to n to return the modal ranking of the MM with high probability [7]. In general, it is a 5-approximation to the Kemeny ranking [8], which, unfortunately, has been shown to be NP-hard to compute [9].

We propose unbalanced *Borda* (*uBorda*) a generalization of Borda for the rank aggregation problem in the context of streaming preferences. In uBorda, each ranking σ_t has an associated weight ρ^t that is proportional to the relevance of the ranking in the sample. The uBorda scores correspond to the weighted average of the given rankings. In other words, uBorda is equivalent to replicating each ranking σ_t ρ^t times and applying Borda.

In stream learning, votes arrive sequentially, at time-stamp t rank σ_t is received, being $t = 0$ the most recent ranking of a possible infinite sequence of votes. In order for uBorda to adapt to the concept drifts in the sequence, we propose to weight the ranking σ_t by ρ^t for a given parameter $\rho \in [0, 1]$. This choice leads to the following Borda score for each item i.

$$B(i) \propto \sum_{t \geq 0} \rho^t \cdot \sigma_t(i) \tag{3}$$

Intuitively, using these specific weights the voting rule *pays more attention* to recent rankings since ρ^t exponentially decreases as the antiquity t of the ranking σ_t increases. The uBorda score can be incrementally computed as $B(i) \propto \sigma_0(i) + \rho \cdot B_1(i)$ where $B_1(i)$ denotes the previous uBorda score, computed using σ_t for $t \geq 1$. Thus, in this streaming scenario, the uBorda scores can be updated incrementally in linear time $\mathcal{O}(n)$, and the Borda algorithm has a computational time complexity of $\mathcal{O}(n \log n)$, in the worst case.

By setting $\rho = 1$ we recover the classic Borda. When $0 \leq \rho < 1$, uBorda can be seen as a Borda algorithm that incorporates a forgetting mechanism to adapt the ranking aggregation process to drifts in streaming data.

In subsequent sections, we will theoretically analyze the properties of uBorda for ranking aggregation in streaming scenarios when the rankings are i.i.d. according to an EMM. First, next lemma provides some known intermediate result regarding the expected value of rankings obtained from a MM.

Theorem 1 ([11]). *Given a sample S of permutations i.i.d. according to a $MM(\pi, \theta)$, the Borda ranking is a consistent estimator of π.*

Following, we show a proof sketch to present the main ideas behind this result and introduce some new notation (see [11] for further details). The authors argue that, on average and for $\pi(i) < \pi(j)$, Borda[2] will rank i above j iff $\mathbb{E}_\pi[\sigma(i)] < \mathbb{E}_\pi[\sigma(j)]$. They reformulated the expression $\mathbb{E}_\pi[\sigma(j)] < \mathbb{E}_\pi[\sigma(i)]$ in a more convenient way:

$$\begin{aligned} \Delta_{ij} &= \mathbb{E}_\pi[\sigma(j)] - \mathbb{E}_\pi[\sigma(i)] \\ &= \sum_{\{\sigma : \sigma(i) < \sigma(j)\}} (\sigma(j) - \sigma(i))(p(\sigma) - p(\sigma\tau)), \end{aligned} \tag{4}$$

where τ is an inversion of positions i and j. Clearly, the proof of Theorem 1 is equivalent to showing that $\Delta_{ij} > 0$ for every i, j such that $\pi(i) < \pi(j)$. To see this, note that for the set of rankings $\{\sigma : \sigma(i) < \sigma(j)\}$ it holds that $(\sigma(j) - \sigma(i)) > 0$. When p is a $MM(\pi, \theta)$, due to the strong unimodality of MMs, we can also state that $(p(\sigma) - p(\sigma\tau)) > 0$. Thus, given a MM with parameters π and θ we have that $\Delta_{ij} > 0$ for every i, j such that $\pi(i) < \pi(j)$.

In the following Lemma, we provide an intermediate result that relates the expected rankings of two MMs, $MM(\sigma, \theta)$ and $MM(\sigma\tau, \theta)$, where τ represents an inversion of positions i and j.

Lemma 2. *Let σ a ranking distributed according to $MM(\pi, \theta)$. Let τ be an inversion of i and j so that $d(\pi\tau, \pi) = 1$. Using the definition on Eq. (4), we can easily see that $\mathbb{E}_\pi[\sigma]$ and $\mathbb{E}_{\pi\tau}[\sigma]$ are related as follows:*

$$\begin{aligned} \mathbb{E}_{\pi\tau}[\sigma(i)] &= \mathbb{E}_\pi[\sigma(j)] = \mathbb{E}_\pi[\sigma(i)] + \Delta_{ij} \\ \mathbb{E}_{\pi\tau}[\sigma(j)] &= \mathbb{E}_\pi[\sigma(i)] \end{aligned} \tag{5}$$

[2] Note that, with a slight abuse in the notation, we are using σ indistinctly for a ranking and for a random variable distributed according to a MM.

3.1 Sample Complexity for Returning π_0 on Average

In this section, we analyze theoretically uBorda for ranking aggregation using stream data distributed according to a EMM. We consider a possibly infinite sequence of rankings for which the current modal ranking is π_0 ant the last drift in the modal ranking has occurred m batches before. We bound the number of batches since the last drift that uBorda needs to recover the current modal ranking π_0 on average.

Theorem 3. *Let $\pi(i) < \pi(j)$, and let τ be an inversion of i and j so that $d(\pi\tau, \pi) = 1$. Let σ_t be a (possibly infinite) sequence of rankings generated by sampling an EMM, such that $\sigma_t \sim MM(\pi, \theta)$ for $t \geq m$ and $\sigma_t \sim MM(\pi\tau, \theta)$ for $m > t$. Then, uBorda returns the current ranking $\pi_0 = \pi\tau$ in expectation when*

$$m > \log_\rho 0.5.$$

Proof. The uBorda algorithm ranks the items in $k \in [n]$ w.r.t. the uBorda score $B(k) = \sum_{t\geq 0} \rho^t \sigma_t(k)$. Thus, uBorda recovers the current ranking π_0 if and only if the expression $\mathbb{E}[\sum_{t\geq 0} \rho^t \sigma_t(i)] < \mathbb{E}[\sum_{t\geq 0} \rho^t \sigma_t(j)]$ is satisfied for every pair i, j.

$$\mathbb{E}[\sum_{t\geq 0} \rho^t \sigma(i)] = \sum_{t\geq m} \rho^t \mathbb{E}_\pi[\sigma(i)] + \sum_{m>t} \rho^t \mathbb{E}_{\pi\tau}[\sigma(i)] = \sum_{t\geq 0} \rho^t \mathbb{E}_\pi[\sigma(i)] + \sum_{m>t} \rho^t \Delta_{ij}$$

$$\mathbb{E}[\sum_{t\geq 0} \rho^t \sigma(j)] = \sum_{t\geq m} \rho^t \mathbb{E}_\pi[\sigma(j)] + \sum_{m>t} \rho^t \mathbb{E}_{\pi\tau}[\sigma(j)] = \sum_{t\geq 0} \rho^t \mathbb{E}_\pi[\sigma(i)] + \sum_{t\geq m} \rho^t \Delta_{ij}$$

Therefore, the uBorda algorithm recovers the current ranking π_0 in expectation if and only if the next inequality holds:

$$\sum_{t<m} \rho^t < \sum_{t\geq m} \rho^t$$

Thus, given $m > 0$, we can satisfy the previous inequality by selecting ρ according to $m > \log_\rho 0.5$, which concludes the proof.

Note that the right hand side expression decreases as ρ decreases, for $\rho \in (0, 1]$, and in the limit, $\lim_{\rho\to 0} \log_\rho 0.5 = 0$. The intuitive conclusion is that as ρ decreases uBorda is more reactive, that is, in expectation it needs less samples to accommodate to the drift. In other words, when ρ decreases uBorda quickly forgets the old preferences and focuses on the most recently generated ones, so it can quickly adapt to recent drifts. One might feel tempted of lowering ρ to guarantee that uBorda recovers the last modal ranking in expectation. However, as we decrease ρ the variance on the uBorda score (see Eq. (3)) increases and, thus the variance of the ranking obtained with uBorda also increases. In the next section, we will show how the confidence of our estimated modal ranking decreases as ρ decreases. Besides, we will provide a lower bound on the number of samples required by uBorda for recovering from a drift.

3.2 Sample Complexity for Returning π_0 with High Probability

In this section we consider a possibly infinite sequence of rankings and denote by π_0 the current consensus of the model, which has generated the last m rankings. We bound the value of m that uBorda needs to return π_0 with high probability in Theorem 5 and Corollary 6.

We present an intermediate result in Lemma 4, where we bound with high probability the difference between the uBorda score and its expected value for a given sub sequence of the stream of rankings. For this intermediate result we consider that there is no drift in the sample.

Lemma 4. *Let $\sigma_r, \sigma_{r+1}, \ldots, \sigma_s$ be $m = s - r + 1$ rankings i.i.d. distributed according to $MM(\pi, \theta)$. In the absence of drifts, the absolute difference between the uBorda score for item i, $(1-\rho) \cdot \sum_{t=r}^{s} \rho^t \sigma_t(i)$, and its expectation $\mathbb{E}[(1-\rho)\sum_{t=r}^{s} \rho^t \sigma_t(i)]$ is smaller than*

$$\epsilon_r^s = (n-1)(1-\rho) \cdot \sqrt{\frac{(\rho^{2r} - \rho^{2s})}{2 \cdot (1-\rho^2)} \cdot \log \frac{2}{\delta}} \tag{6}$$

with, at least a probability of $1-\delta$,

$$P\left(\left| \sum_{t=r}^{s} \rho^t (\sigma_t(i) - \mathbb{E}[\sigma_t(i)]) \right| \leq \epsilon_r^s \right) \geq 1 - \delta$$

The proof, which is omitted, uses the Hoeffding's inequality and sums of geometric series.

The next result gives a lower bound on the number of samples required for recovering from a drift with high probability for a drift at distance 1. This result is generalized for distance d drifts afterwards. The lower bound is given as a function of the concentration of the underlying distribution θ and parameter ρ.

Theorem 5. *Let $\pi(i) < \pi(j)$ and let τ be an inversion of i and j so that $d(\pi\tau, \pi) = 1$. Let σ_t for $t \geq 0$ be a (possibly infinite) sequence of rankings generated by sampling an evolving MM, such that for $t \geq m$ then $\sigma_t \sim MM(\pi, \theta)$ and for $t < m$ then $\sigma_t \sim MM(\pi\tau, \theta)$. The number of samples that uBorda needs to return the current modal ranking $\pi_0 = \pi\tau$ in expectation with probability $1-\delta$ is at least*

$$m > \log_\rho \left(\frac{-(1-\rho)^2}{\sqrt{1-\rho^2}} \frac{n\sqrt{0.5 \log \delta^{-1}}}{\Delta_{ij}} + 0.5 \right) \tag{7}$$

where Δ_{ij} is defined in Eq. (4).

Proof. Following the idea in Theorem 3, the algorithm uBorda returns the current ranking $\pi_0 = \pi\tau$ in expectation when the expected uBorda score (see Eq. (3)) of element i is greater than of element j as

$$\sum_{t=m}^{\infty} \rho^t \mathbb{E}_\pi[\sigma(j)] + \sum_{t=0}^{m-1} \rho^t \mathbb{E}_{\pi\tau}[\sigma(j)] < \sum_{t=m}^{\infty} \rho^t \mathbb{E}_\pi[\sigma(i)] + \sum_{t=0}^{m-1} \rho^t \mathbb{E}_{\pi\tau}[\sigma(i)], \tag{8}$$

which is based on Eq. (4). Equation (8) would be an accurate measure for the sample complexity if the expected value did not deviate at all from the sum. However, according to Lemma 4 we can upper bound the difference between the uBorda score as shown in Lemma 4. Next, we make use of the definition in Eq. (6) to define the deviations of the uBorda score before the drift, ϵ_m^{∞}, and after the drift, ϵ_0^{m-1} and let ϵ_0^{∞}. Note that the next inequality holds:

$$\epsilon_m^{\infty} + \epsilon_0^{m-1} = \epsilon_0^{\infty} \tag{9}$$

Therefore, we can state that with probability $1 - \delta$ we have recovered from a drift when the m satisfies

$$\sum_{t=m}^{\infty} \rho^t \mathbb{E}_{\pi}[\sigma(j)] + \epsilon_m^{\infty} + \sum_{t=0}^{m-1} \rho^t \mathbb{E}_{\pi\tau}[\sigma(j)] + \epsilon_0^{m-1} < \sum_{t=m}^{\infty} \rho^t \mathbb{E}_{\pi}[\sigma(i)] - \epsilon_m^{\infty} + \sum_{t=0}^{m-1} \rho^t \mathbb{E}_{\pi\tau}[\sigma(i)] - \epsilon_0^{m-1},$$

which implies that with probability $1 - \delta$ uBorda will recover $\pi\tau$ when the following expression holds:

$$\Delta_{ij} \sum_{t=0}^{m-1} \rho^t - 2\epsilon_0^{m-1} > \Delta_{ij} \sum_{t=m}^{\infty} \rho^t + 2\epsilon_m^{\infty}$$

After some algebra, and using the result in Eq. (9), we obtain the lower bound for m

$$\begin{aligned} \sum_{t=0}^{m-1} \rho^t \Delta_{ij} - \sum_{t=m}^{\infty} \rho^t \Delta_{ij} &> 2\epsilon_m^{\infty} + 2\epsilon_0^{m-1} > 2\epsilon_0^{\infty} \\ \frac{\rho^m - 1}{\rho - 1} - \frac{\rho^m}{1 - \rho} &> \frac{\epsilon_0^{\infty}}{\Delta_{ij}} \\ \rho^m &> \frac{\epsilon_0^{\infty}(\rho - 1)}{\Delta_{ij}} + 0.5 \\ m > \log_{\rho} \left(\frac{(\rho - 1)(1 - \rho)}{\sqrt{1 - \rho^2}} \frac{n\sqrt{0.5 \log \delta^{-1}}}{\Delta_{ij}} + 0.5 \right) \end{aligned} \tag{10}$$

which concludes the proof.

The previous result gives a bound for drifts at distance 1 and it is generalized in the following Corollary for any drift.

Corollary 6. *For τ a drift at distance d, the maximum rank difference for item i, is $|\sigma(i) - \sigma\tau(i)| < d$. For m as described in Theorem 5, after dm observed samples, uBorda recovers the current central ranking with arbitrary high probability.*

3.3 Choosing ρ Optimally

In this section how to compute the value of ρ for recovering from a drift with high probability after receiving m rankings.

Theorem 7. *Let m be the number of rankings received after the last concept drift. uBorda recovers the true ranking with probability $(1-\delta)$ by using the forgetting parameter value $\rho* = \arg\max_\rho f(\rho, m)$ that can be found numerically, where*

$$f(\rho,m) = \frac{2\rho^m - 1}{\rho - 1}\left(\sqrt{\frac{\rho^{2m}}{1-\rho^2}}\frac{1-\rho}{\rho^m} + \sqrt{\frac{\rho^{2m}-1}{\rho^2-1}}\frac{\rho-1}{\rho^m-1}\right). \tag{11}$$

The previous result provides the optimal value of ρ given the maximum delay allowed for determining a change in the consensus preference, i.e., maximize the probability of recovering the ground true ranking after m rankings. Moreover, it has a probability of success as $1-\delta$. An more basic approach to include this kind of expert knowledge is to set a window of size m and consider only the last m rankings. In this case, the probability of success can be found with classic quality results for Borda [7]. However, our proposed uBorda has different advantages over traditional window approaches. First, uBorda allows handling and infinite sequence of permutations. This means that for every pair of items that have not suffered a drift there is an infinity number of samples available. Second, this analysis can be adapted to more general settings. In other words, this paper considers the particular situation in which recent rankings are more relevant than old ones but there are situations in which a subset of the rankings are more relevant than other for different reasons. For example, because some voters provide rankings that are more trustworthy than others. In this case, uBorda can be used as a rank aggregation procedure in which expert agents have a larger weight.

4 Generalizing Voting Rules

So far we have adapted the Borda algorithm to perform rank aggregation when the rankings lose importance over time. This section generalizes that setting in every possible way. First, we consider families of algorithms (which include Borda and Condorcet). Second, we consider that each ranking has different "importance" (rather than decreasing importance with time).[3]

We propose a general framework and skip the search for statistical properties for the voting rules, since it needs assumptions on the data generating process and is context-dependent. Instead, we validate empirically the Condorcet rule in the experimental section for the evolving preference setting, showing applicability and efficiency.

[3] We can say that some voters are more *trusted* than others.

In particular, we are given a set of rankings $S = \{\sigma_v : v \in [m]\}$ representing the preferences of a set of voters, where each voter $v < m$ has a weight $w_v \in \mathbb{R}^+$ representing our confidence in σ_v. There are similar contexts in crowd learning scenarios in which the weight of each voter is related to its reliability [17].

The most relevant voting rules are those in the Unweighted Tournament Solutions family (C1 voting rules) and the Weighted Tournament Solutions family (C2 voting rules [4][4]). We generalize these voting rules by generalizing the *frequency matrix* upon which all the rules in these families are defined. The weighted frequency matrix, N, is the following summary statistic of the sample of rankings S.

$$N_{ij} = \sum_{v<m} w_v \mathbb{I}[\sigma_v(i) < \sigma_v(j)], \tag{12}$$

where w_v is the weight of the preference σ_v.[5] The family of C2 functions are given as a function of the *majority margin* matrix, wich is computed using the frequency matrix. The generalized majority matrix, M, is computed using N as follows:

$$M_{ij} = N_{ij} - N_{ji} = 2N_{ij} - \sum_{v<m} w_v, \tag{13}$$

Intuitively, M counts the difference on the weighting votes that prefer i to j and those that prefer j to i. The Kemeny ranking can be formulated using the majority margins matrix in Eq. (13) as follows

$$\sigma_K = \arg\min_{\sigma \in S_n} \sum_{i,j \in [n]:\sigma(i)>\sigma(j)} M_{ij}. \tag{14}$$

The Kemeny ranking has been shown to maintain several interesting properties, such as being a median permutation of the sample of rankings, and the MLE of the modal ranking of the sample when the rankings are i.i.d. according to a Mallows model [10]. Unfortunately, the problem of computing the Kemeny ranking given a sample of rankings S has been shown to be NP-hard [9]. Other interesting members of the family of the C2 family are a pairwise query algorithm that returns the Kemeny ranking with high probability [5], the ranked pairs method [24] and a 4/3-approximation of the Kemeny ranking [2].

The C1 family or Unweighted Voting Rules family uses the unweighted version of the graph defined by the majority margin matrix M. The most prominent member of this family is Condorcet: A Condorcet winner for a collection of rankings is an alternative i that defeats every other alternative in the strict pairwise majority sense. By stating it as a function of Eq. (13), we can define Condorcet for balanced and unbalanced settings.

[4] Condorcet and Borda belong to C1 and C2 voting rules respectively.
[5] By setting $w_v = 1$ the standard frequency matrix is obtained.

Definition 8 *(Condorcet). The Condorcet winner is the item i for which $M_{ij} > 0$ for all j. For convenience, we define the number of Condorcet victories of item i is $\{|j| : M_{ij} > 0\}$.*

The Condorcet winner does not always exist, and this holds for both balanced and unbalanced settings. This is known as the *Condorcet paradox*. We show strong experimental results in the following section.

5 Experiments

In this section, we provide empirical evidence of the strengths of uBorda to deal with the ranking aggregation problem in streaming scenarios. These experiments illustrate the theoretical results for the estimation of the ground true ranking after a drift (Sect. 3.2) and the tuning of the forgetting parameter ρ (Sect. 3.3). Moreover, we show its applicability beyond the theoretical setting using different ranking models and several voting rules (see Sect. 4). For each parameter configuration, the results are run 10 times and the average is shown with a stroke line and the .95 confidence interval ($\lambda = 0.05$) shadowed.

5.1 Rank Aggregation for Dynamic Preferences with uBorda

This section analyses the performance of uBorda as a rank aggregation algorithm using streaming data of rankings. We consider a synthetic stream of 300 rankings. The larger m, the easier the learning problem from the statistical perspective and the sample size of $m = 20$ per drift is challenging even in the classical non-drift setting [7]. The stream of rankings has been sampled from an evolving **ranking model** that suffers concept drifts periodically. We have considered three types of evolving ranking models using the Mallows model, Plackett-Luce and Generalized Mallows model[6]. We consider two settings of each model, close to the uniform distribution or far from it.[7]

Between one subsequence and the next one, there has been a **drift** on the ground true ranking. The drift can be a small-step drift (drift at distance 1) or a large-step drift (the distance randomly increases in a number between 1 and n). In every case, the distance between the first and the last ground true ranking is maximal, $n * (n-1)/2$.

For the visualization of the results, each ranking in the sequence lays in the x axis, and drifts are denoted with a grey vertical line in all plots. At each point of the sequence, the uBorda ranking $\hat{\sigma}_0$ estimates the ground true ranking σ_0 with the observed rankings so far, and therefore, the y axis measures the Kendall's-τ distance $d(\sigma_0, \hat{\sigma}_0)$.[8] Therefore, the smaller, the better.

[6] In the supplementary material.

[7] For the Mallows, the expected distance (which is a function of the spread parameter) is 0.1 and 0.4 of the maximum distance. For Plackett-Luce the weights are $w_i = n - i$ and $w_i = e^{n-i}$.

[8] We follow the classical test-then-train strategy.

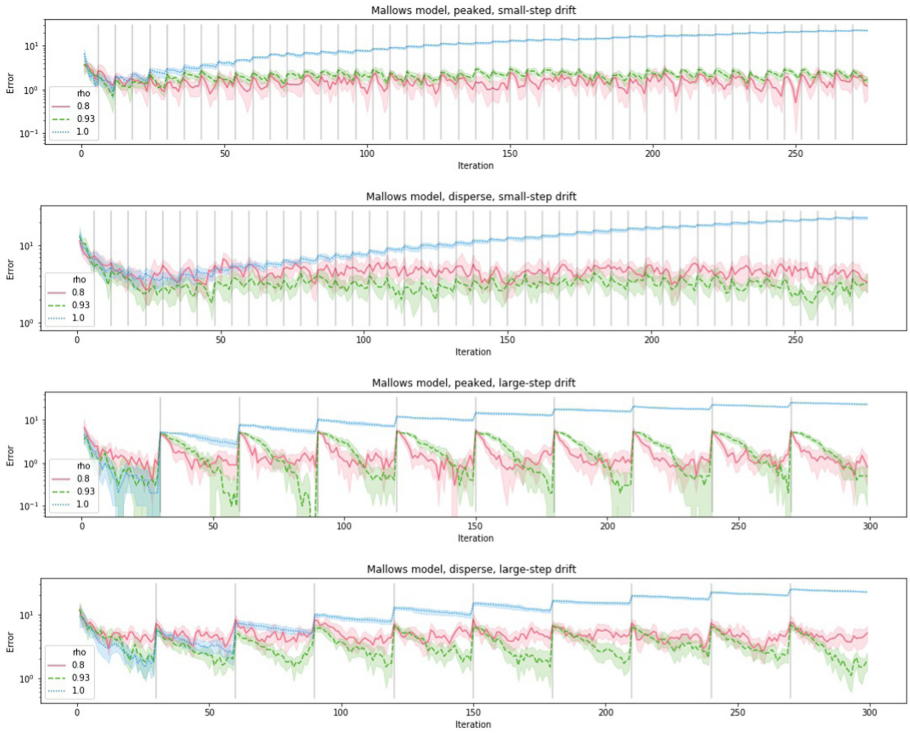

Fig. 1. uBorda algorithm for Mallows model with large and small step drifts

Simulating the inclusion of expert knowledge in the experiment, we expect the rank aggregation to recover from the drifts after $m = 20$ samples. Theorem 7 shows that finding the value for ρ that minimizes the probability of error for a given number o samples is done by solving a convex optimization problem. For our choice of $m = 20$ the optimal ρ is 0.93. For comparison, we use different fading factors $\rho \in \{0.8, 0.93, 1\}$, each corresponding to a different line in the plot.

Figure 1 shows the results of uBorda for the Mallows model and the particular configuration can be seen in the title. When a drift occurs (after each vertical grid-line), the error increases for every choice of ρ. However, differences in the value of ρ cause critical differences in the behavior of uBorda: while $\rho = 1$ (the classic Borda algorithm) gets an increasing error with a larger sample, $\rho < 1$ can adapt to the drifts, consistently with the findings in the previous sections.

When $\rho = 0.93$ uBorda has the most accurate results. As shown in Theorem 7, this value maximizes the probability of recovering from a drift in 20 samples among the parameter values considered. Moreover, after these 20 samples, it recovers the ground true ranking with an error smaller than 0.1. Choosing $\rho = 0.8$ makes uBorda forget quicker the previous permutations and this can lead to a situation in which too few of the last permutations are considered to estimate the

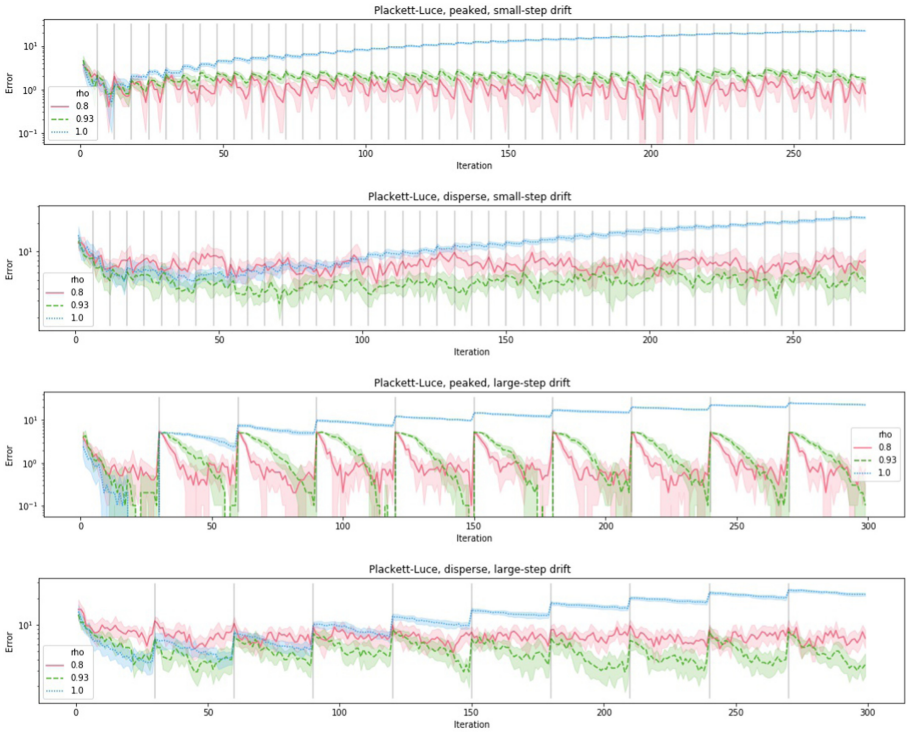

Fig. 2. uBorda algorithm for Plackett-Luce model with large and small step drifts

consensus. The more chaotic behavior of the smallest value of $\rho = 0.8$ (remind the logarithmic scale for the Y-axis) is related to this phenomenon in which few permutations are contributing to the estimation of the consensus, i.e., uBorda is aggregating a small number of rankings. Finally, for $\rho = 1$ uBorda is equivalent to Borda and does not forget the past rankings. That leads to an increasing error as new concept drifts occur because uBorda is not able to accommodate the changes in the consensus ranking of the evolving Mallows model.

Figure 2 show the results obtained with an evolving Plackett-Luce model. The conclusions are similar to those obtained for the evolving Mallows model. Since Plackett-Luce implies a transitivity pairwise matrix of the marginals, Borda is an unbiased estimator of the central ranking in a non-evolving setting [11]. For $\rho < 1$ uBorda can adapt to the changes in the ground true ranking, while the error of Borda (i.e., uBorda with $\rho = 1$) increases after each drift. The supplementary material shows similar results for other parameter setting and probability models in the probabilistic ranking literature.

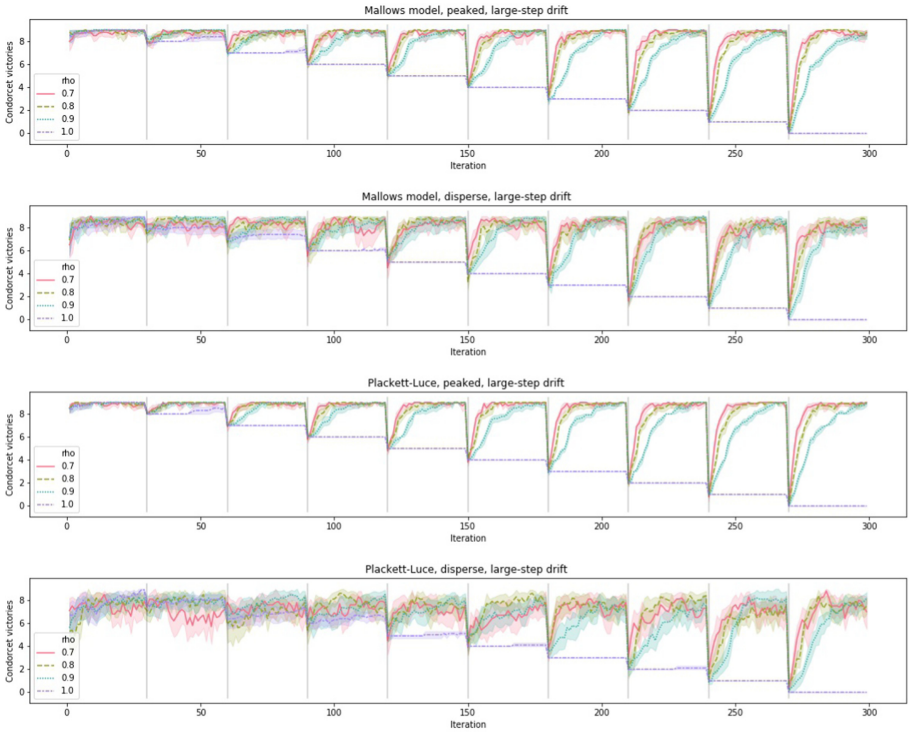

Fig. 3. Condorcet algorithm for Mallows and Plackett-Luce models

5.2 Condorcet Winner in Dynamic Preferences

Section 4 generalizes families of voting rules to the unbalanced voting (and evolving preferences) setting. In this section, we validate empirically the most relevant member of the C1 family, Condorcet (see Definition 8). In a similar setting to the previous experiments, we consider the Plackett-Luce and the Mallows models - since both possess a Condorcet winner. For each model, we consider the same peaked and disperse distributions, and the results are in Fig. 3. Drift i, represented as the i-th vertical line, is a swap between the first and the i-th item. A Condorcet winner will have $n-1=9$ Condorcet victories (y axis), therefore, the higher, the better. We show the results of 4 different settings of our algorithms, $\rho = [.7, .8, .9, 1]$.

In all the experiments, when $\rho = 1$ the Condorcet Victories of the winner item strictly decreases with each drift. When the forgetting parameter $\rho < 1$ the Condorcet Victories also decrease but recover after observing some rankings from the same distribution. This recovery is faster for smaller values of ρ, as we show theoretically for Borda in Sect. 3, and with higher variance also increases. Further experiments in the Supplementary material.

6 Conclusions

Stream learning with concept drift is an online learning setting where the distribution of the preferences changes with time. We denote as uBorda the version of Borda in which the voters have different weights and show that it can be computed efficiently, handling a possibly infinite sequence of rankings in linear space and quasi-linear time complexity. A detailed analysis leads to several contributions. First, we bound the number of samples required by uBorda to output the current ground truth modal ranking with high probability. Second, we show how to include expert knowledge to the uBorda. We generalize families of functions (including Borda and Condorcet) to the situation in which some voters are more reliable than others. We call this scenario *unbalanced voting* and allow arbitrary *reliability* values for the voters. This work, specially Sect. 4, opens natural research lines in the context of weighted voting. We plan to study statistical properties of the different voting rules in the context of expert vs non-expert voters.

Acknowledgments. This work is partially funded by the Industrial Chair "Data science & Artificial Intelligence for Digitalized Industry & Services" from Telecom Paris (France), the Basque Government through the BERC 2018–2021 and the Elkartek program (KK-2018/00096, KK-2020/00049), and by the Spanish Government excellence accreditation Severo Ochoa SEV-2013-0323 (MICIU) and the project TIN2017-82626-R (MINECO). J. Del Ser also acknowledges funding support from the Basque Government (Consolidated Research Gr. MATHMODE, IT1294-19).

References

1. Ailon, N.: Improved bounds for online learning over the permutahedron and other ranking polytopes. In: Artificial Intelligence and Statistics, pp. 29–37 (2014)
2. Ailon, N., Charikar, M., Newman, A.: Aggregating inconsistent information: ranking and clustering. J. ACM **55**, 1–27 (2008)
3. Borda, J.: Memoire sur les elections au scrutin. Histoire de l'Academie Royal des Sciences **102**, 657–665 (1781)
4. Brandt, F., Conitzer, V., Endriss, U., Lang, J., Procaccia, A.D.: Handbook of Computational Social Choice (2016)
5. Braverman, M., Mossel, E.: Noisy sorting without resampling. In: Proceedings of the 19th Annual Symposium on Discrete Algorithms (2008)
6. Busa-Fekete, R., Hüllermeier, E., Szörényi, B.: Preference-based rank elicitation using statistical models: the case of Mallows. In: Proceedings of the 31th International Conference on Machine Learning (ICML), pp. 1071–1079 (2014)
7. Caragiannis, I., Procaccia, A.D., Shah, N.: When do noisy votes reveal the truth? In: Proceedings of the 14th ACM Conference on Electronic Commerce, pp. 143–160 (2013)
8. Coppersmith, D., Fleischer, L.K., Rurda, A.: Ordering by weighted number of wins gives a good ranking for weighted tournaments. ACM Trans. Algorithms **6**(3), 1–13 (2010)
9. Dwork, C., Kumar, R., Naor, M., Sivakumar, D.: Rank aggregation methods for the Web. In: International Conference on World Wide Web (2001)

10. Fligner, M.A., Verducci, J.S.: Distance based ranking models. J. Roy. Stat. Soc. **48**(3), 359–369 (1986)
11. Fligner, M.A., Verducci, J.S.: Multistage ranking models. J. Am. Stat. Assoc. **83**(403), 892–901 (1988)
12. Freeman, R., Zahedi, S.M., Conitzer, V.: Fair social choice in dynamic settings. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI) (2017)
13. Gam, J.: Knowledge Discovery from Data Streams (2010)
14. Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: Survey on concept drift adaptation. ACM Comput. Surv. **46**(4), 44 (2014)
15. Hosseini, H., Larson, K., Cohen, R.: Matching with dynamic ordinal preferences. In: 29th AAAI Conference on Artificial Intelligence (2015)
16. Irurozki, E., Calvo, B., Lozano, J.A.: PerMallows: an R package for mallows and generalized mallows models. J. Stat. Softw. **71**, 1–30 (2019)
17. Karger, D.R., Oh, S., Shah, D.: Iterative learning for reliable crowdsourcing systems. In: Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011 (2011)
18. Krempl, G., et al.: Open challenges for data stream mining research. ACM SIGKDD Explor. Newsl. **16**, 1–10 (2014)
19. Lu, T., Boutilier, C.: Learning Mallows models with pairwise preferences. In: Proceedings of the 28th International Conference on Machine Learning, ICML 2011 (2011)
20. Mallows, C.L.: Non-null ranking models. Biometrika **44**(1–2), 114–130 (1957)
21. Parkes, D.C., Procaccia, A.D.: Dynamic social choice with evolving preferences. In: 27th AAAI Conference on Artificial Intelligence (2013)
22. Siddiqui, Z.F., Tiakas, E., Symeonidis, P., Spiliopoulou, M., Manolopoulos, Y.: xStreams: recommending items to users with time-evolving preferences. In: Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics, WIMS 2014 (2014)
23. Tal, M., Meir, R., Gal, Y.: A study of human behavior in online voting. In: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (2015)
24. Tideman, T.N.: Independence of clones as a criterion for voting rules. Soc. Choice Welf. **4**, 185–206 (1987). https://doi.org/10.1007/BF00433944
25. Vitelli, V., Sørensen, Ø., Crispino, M., Frigessi, A., Arjas, E.: Probabilistic preference learning with the Mallows rank model. J. Mach. Learn. Res. **18**(1), 1–49 (2018)
26. Xia, L.: Learning and Decision-Making from Rank Data. Synthesis Lectures on Artificial Intelligence and Machine Learning, vol. 13, no. 1, pp. 1–159 (2019)
27. Yasutake, S., Hatano, K., Takimoto, E., Takeda, M., Hoi, S.C.H., Buntine, W.: Online rank aggregation. In: Asian Conference on Machine Learning (2012)