ARCHIVIO ISTITUZIONALE
DELLA RICERCA

Alma Mater Studiorum Università di Bologna
Archivio istituzionale della ricerca

Exploration Trade-offs in Web Recommender Systems

This is the final peer-reviewed author's accepted manuscript (postprint) of the following publication:

Published Version:
Baeza-Yates R., Delnevo G. (2022). Exploration Trade-offs in Web Recommender Systems. Institute of Electrical and Electronics Engineers Inc. [10.1109/BigData55660.2022.10325847].

Availability:
This version is available at: https://hdl.handle.net/11585/954179 since: 2024-01-29

Published:
DOI: http://doi.org/10.1109/BigData55660.2022.10325847

Terms of use:

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (https://cris.unibo.it/).
When citing, please refer to the published version.

(Article begins on next page)

11 November 2026

Exploration Trade-offs in Web Recommender Systems

Ricardo Baeza-Yates[1] and Giovanni Delnevo[2]

[1]Institute for Experiential AI, Northeastern University, Silicon Valley Campus, CA, USA.
[2]Department of Computer Science and Engineering, University of Bologna, Bologna, Italy.

Contributing authors: rbaeza@ieee.org; giovanni.delnevo2@unibo.it;

Abstract

One of the main problems of web recommender systems is exposure bias, due to the fact that the web system itself is partly generating its own future, as users can only click on items shown to them. This bias not only creates popularity bias for products but also is one of the main challenges for recommender systems that deal with a very dynamic environment, where new items and users appear frequently and also user preferences change (or the market changes as happened with the coronavirus pandemic). The main paradigm to deal with these changes is to explore and exploit, avoiding the filter bubble effect. However, too much exploration also reduces short-term revenue and hence is usually traffic bounded. In this work, we present a counterfactual analysis that shows that web recommender systems could improve their long-term revenue if significantly more exploration is performed. This is good for the web recommender system but also for everyone as it creates more fair and healthy digital markets. This also improves the web user experience so is a double win-win for the e-commerce platform, the sellers, the users, and ultimately society.

Keywords: keyword1, Keyword2, Keyword3, Keyword4

1 Introduction

Most web recommender systems are optimized by using implicit user feedback, that is, clicks or other traceable user interactions. However, those interactions are biased to

the choices that such systems offer to their users, as clicks can only be done on things that are shown to them. Hence, these closed feedback loops are tainted by exposure or presentation bias [1], which is probably the bias that has most impact on search and recommender systems, as it affects all users. The most well-known problem of this type is related to personalization and is called the filter bubble [2]. That is, we cannot offer something new to a user as is not in their past interaction data. This problem is partially solved by using collaborative filtering and recommending things that are liked by similar users. However, this does not cover long tail users that have little data and/or new users or items.

However, there is a second problem, not so well-known, that is even more important than the filter bubble for users. As these recommender systems are usually based on (deep) machine learning, the closed feedback loop yields self-fulfilled prophecies and/or sub-optimal solutions, leading to an echo chamber for the system itself. Even more, if there is no single optimization function, sometimes subsystems compete among themselves, learning also from each other rather than real user behavior. The typical case is when two separate teams have different goals that conflict with each other, such as user engagement and monetization. In this case, the user experience is also harmed. This problem is usually addressed by the explore and exploit paradigm [3, 4] to learn how the environment is changing by using Multi-Armed Bandit (MAB) methods. However, most of the time exploration is bounded by design, because it implies a loss of short-term revenue. That means that the recommender system might be in a local optimum and possibly will never achieve a long-term revenue optimization. This problem is worsened by the fact that the world is highly dynamic, where new users and items appear, other users and items disappear or the users' preferences change. These changes can also be drastic due to external changes such as the current coronavirus pandemic, where new needs suddenly appear (*e.g.*, masks and sanitizers) as well as many new users registering due to lockdown measures.

The previous problem has important consequences in the fairness of the digital market behind the recommender system such as sellers in an e-commerce platform or advertisers in search engines and social media sites. Fairness in these two-sided marketplaces is one of the current challenges in recommender systems [5]. Indeed, exposure bias discriminates against market players in the long tail creating unhealthy inequalities and threatening the continuity of smaller and/or niche-based players. This in turn threatens the life-cycle of the market itself as competition decreases. One solution is to explore as much as possible all users and items to learn their preferences and intrinsic values, respectively. As it is well known, more information implies better rational decisions. The same is true for a recommender system, more knowledge of the environment/world implies that in the long run should increase its revenue even though temporally, due to increased exploration, revenue may decrease. However, this is only possible if the traffic that sustains the recommender system is large enough to learn well how user preferences and item popularity are changing.

Understanding the vicious cycle above and the implications to all the stakeholders: e-commerce platforms, providers and consumers, is key for the evolution of digital markets. In fact, there are many side effects [6] that have economic, legal, and societal impact.

So, in the context of web recommender systems that have an exploration module, our contributions are threefold:

- A new problem formulation that, to the best of our knowledge, ties exploration and knowledge of the users preferences in recommendation systems that considers the impact of exposure bias and a proxy measure for fairness to providers, including its impact in digital markets as well as society at large;
- A comparison of well-known exploration algorithms in the context of this new formulation, showing the current trade-offs between revenue and knowledge of the environment/world where the web recommender system operates. This comparison is done in a counterfactual approach as doing A/B testing in a real system is usually not feasible; and
- Defining a methodology to mimic a dynamic world scenario which captures some of the complexities of the real world.

The remainder of this paper is structured as follows. In Section 2, we present the background and related work while in Section 3 we formally present the problem and its complexity. In Section 4, we detail the data set used and exploration algorithms chosen while Section 5 presents and discusses the experimental results. Finally, Section 6 provides some conclusions and future work.

2 Background and Related Work

This trade-off between information acquisition via exploration (which is forward looking) and the exploitation of the latter for immediate reward optimization (which is more myopic in nature) is fundamental in many problem areas, and is one of the main reasons MAB problems have been so widely studied, since their implicit inception in the seminal papers of Thompson [7] and Robbins [8]. This trade-off is natural in recommender systems and hence MABs have been profusely used for the information acquisition part [4, 9], including the cold-start problem [10].

On the other hand, the closed feedback loop of recommendations and user choices may degenerate in user bubbles [2, 9], especially when personalization is used. This also produces popularity bias as very few items are recommended and the Matthew effect takes place, so the rich get richer and poor gets poorer, amplifying bias [11]. Many authors have addressed popularity bias which is a consequence of exposure bias [12, 13]. Due to this, there is the need to mitigate/control this bias [14–16] as well as choose evaluation measures that do not add other biases [17].

Regarding exposure bias, only recently researchers have focused on the problem and the multi-sided aspects as it affects providers, as well as consumers [18, 19]. One important aspect is the discrimination/fairness of tail items/sellers that we also address in this work [20, 21]. Hence, there are several ways on how to achieve fairness in two-sided marketplaces [22], including diversity [23] and evaluation aspects [24], among others.

Considering the complexity involved in the understanding and fair evaluation of these systems, counterfactual approaches are very effective as you can assume certain future conditions and work back to see the implications of those conditions in the

present [25]. In that sense, the closest work to ours is the trade-off between relevance, fairness, and satisfaction in recommender systems using a counterfactual approach done by [26]. In our case, we consider revenue and market fairness, although the latter in a generic and aggregated way. Hence, both approaches are not comparable.

Finally, for more information on bias on recommender systems and the challenges ahead, see [27, 28].

3 The Exploration Trade-off Problem

To exemplify the problem at hand, let's consider the exploration module of an abstract web recommender system where there are N users and M items, with item i having an associated revenue r_i. Let's consider that items have a probabilistic preference, that is, users partially like (or dislike) them. Hence, it is possible to assume that, for user u, the Click-Through Rate (CTR) for item i is $c_{u,i}$, being 0 when users completely dislike them. If there is no knowledge of the world, which is the case at the beginning (cold-start problem) as well as when there are new unique users or items, if item i is shown to a random user u, the expected revenue would be $r_i c_{u,i}$. However, the system needs first to learn the values for c for all user-item pairs. Later, in the steady state, it needs to learn the changes of our world, and hence the incoming traffic in a certain period of time t should be at least enough to learn them in the same period with a certain desired error. That is:

$$Traffic_t \geq Approximate(\Delta World_t, \epsilon(CTR))$$

where *Approximate* is the sample size needed to approximate the world for a given CTR value and ϵ will be a parameter to specify how well it is possible to do it. For example, $\epsilon(CTR)$ could be a relative error of 10% for CTRs of at least a certain value and a fixed absolute error for smaller ones. For comparison, it will be assumed that there is an Oracle that knows the complete world (real CTRs) all the time.

The change for a given user-item pair is $k_{u,i} = \Delta c_{u,i}$, assuming that the system knows all the time the revenue associated with all items as well the users that appear and disappear, plus the items that disappear. In the case of new items and new users, $k_{u,i}$ is exactly the new knowledge that the system needs to learn. To simplify, the assumption is that users change their preference (this includes new users) as well as their CTRs slowly in time for $m << M$ new items. Hence, the total change of knowledge in the system is

$$\Delta K \approx \sum_{u=1}^{N} \sum_{i=1}^{M} k_{u,i} = O(NM) \ .$$

Now, there is no need to learn the exact value of $k_{u,i}$, as that is not practical and also unfeasible if the environment is quite dynamic. So, let's assume that $\epsilon(CTR)$ is used as a fixed absolute error ϵ in our estimations (for a relative error the analysis is similar). Therefore, to learn $k_{u,i}$ there is the need for a traffic to our recommender system of $O(k_{u,i}/\epsilon^2)$ for a fixed confidence interval [29]. Hence, to learn all the changes there

is the need for a traffic T of at most $O(\Delta K/\epsilon^2) = O(NM/\epsilon^2)$ in the time period P considered, assuming that all users and items are completely independent of each other (in other words, we cannot learn preferences of a user from the behavior of other users). Hence, if we have enough traffic, the recommender system can learn the long-term optimum. Otherwise, a partial optimality may be achieved or, if the traffic is much lower, it may survive in a local optimum or even collapse. The important observation here is that to lower the error linearly, the traffic needs to grow quadratically.

In our analysis ahead we use two different $\epsilon(CTR)$:

Fair: A 10% relative error is used until a threshold $\epsilon = 0.01$ is reached. Then, such a value is used as an absolute error (achieving small relative errors for improbable events is neither efficient nor effective). This is our *Fair* distribution.

PB: There is less error in the head than in the tail of the distribution, so is *Popularity Biased* (PB). The error for a binomial distribution is applied using a 95% confidence interval and the Agresti-Coull technique, which does not diverge for CTR values close to 0 [30].

4 Experimental Setup

4.1 Data Set

In our experiments, we used two different data sets. The first one is the Coupon Purchase Prediction challenge from Kaggle.[1] We will refer to it as DS1. The data belongs to one of the most important Japanese coupon websites. They provided transactional data relative to 22,873 users and 310 different coupons of different prices, in the period from 2011-07-01 to 2012-06-30. In our case, each click implies that the user buys the clicked item (coupon) paying the associated price. From this set of transactions, we selected only those of one-hundred most active users. We then filter only the transactions involving the one-hundred most frequent coupons (items). Then, for each one of the 10,000 user-item pairs, we computed the relative CTR as the number of purchases divided by the total number of impressions. Figure 1 shows the average CTR distribution for users and items, being the items curve a bit more skewed. Being the data set of a Japanese company, the revenues are expressed in millions of Yen, although our focus is on relative performance.

The second dataset, instead, comes from the SIGIR e-commerce data challenge.[2] We will refer to it as DS2. They provided transactional data relative to 103,396,636 users and 446,362 different items of different prices. The data relates to the actions performed by users interacting with the system, such as viewing a product, adding or removing products from the cart and purchasing, in the period from 2019-01-01 to 2021-05-31. To avoid a very large and sparse matrix, we selected only users who bought at least 33 items and items bought by at least 140 users. At the end of this process, we obtained data relative to 124 users and 514 items. For each one of the user-item pairs, we computed again the relative CTR as the number of purchases divided by the total number of impressions.

[1] https://www.kaggle.com/c/coupon-purchase-prediction/overview
[2] https://github.com/coveooss/SIGIR-ecom-data-challenge

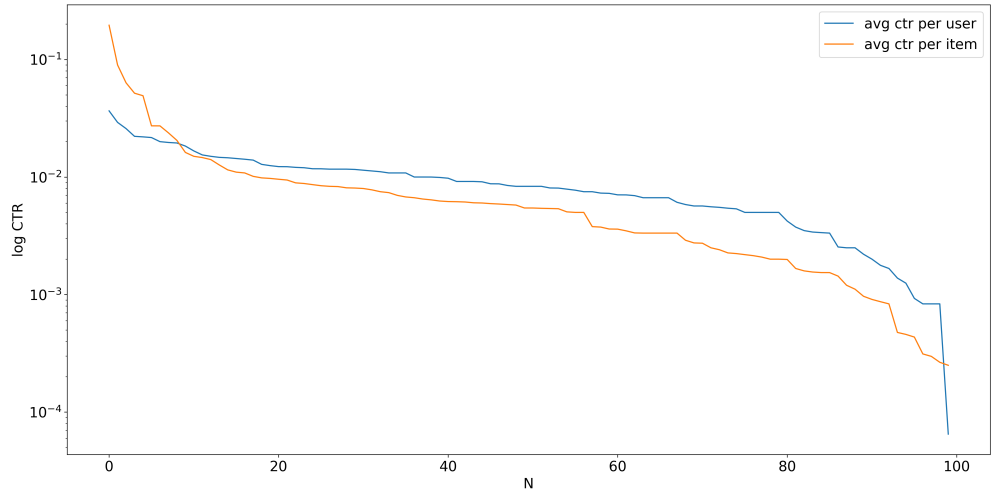

Fig. 1 Sorted Average CTRs for users and items in DS1.

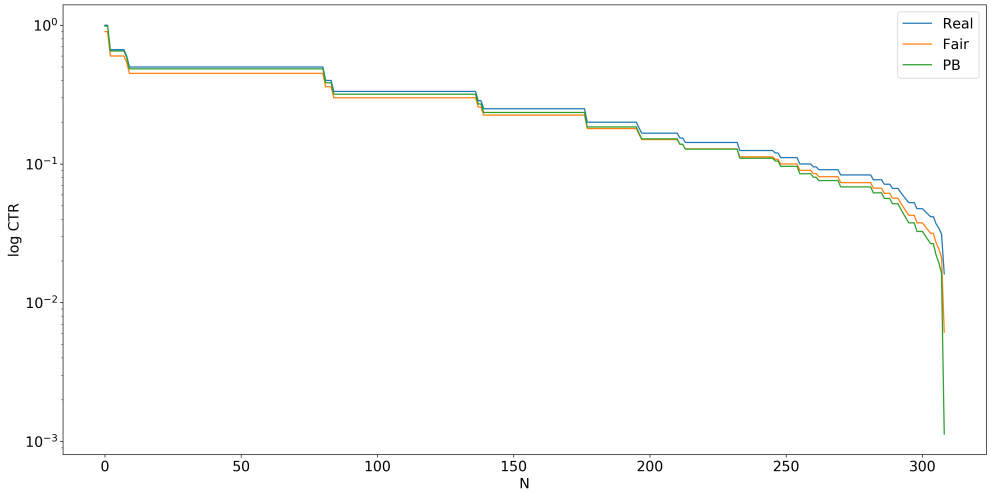

Fig. 2 Comparison of the three distributions used in DS1: Real, Fair and PB.

For both datasets, we call *Real* the actual distribution of CTRs. We also consider Fair and PB from the previous section as distributions with error, that is when the recommender system does not know exactly its world. Just to remind the reader, *Fair* was computed using a 10% relative error while in the *PB* there is less error in the head than in the tail. These can be seen as counterfactual distributions for the system. For example, the ones relative to DS1 are shown in Figure 2.

4.2 Methodology

Since we are interested in evaluating the implication of long-term exploration (or a lack of it), we evaluated two different scenarios, a static and a dynamic one. In the static one, the CTR matrix never changes (*i.e.*, the users never change their preferences and the items are always the same). In the dynamic scenario, instead, the CTR matrix changes every fixed number of epochs while the item set never changes. This is not a loss of generality as changing items is similar to changing users and having both only exacerbates the problem. The same is true if changes happen at any time. In particular, the ratio of change of user preferences with respect to an item is computed using a truncated Normal distribution with an average of 0.1 and standard deviation of 0.1, from which 0.1 is finally subtracted, so the average change for all users is null. Again, more drastic changes make things worse.

For each different distribution, we employed several algorithms for the MAB problem in order to estimate the CTRs. In particular, we use four well-known exploration algorithms: Thompson Sampling [7], ϵ-Greedy [31], Upper Confidence Bound (UCB) [32], and Exp3 [33].

Furthermore, we also compared their performances against an Oracle that always chooses the best arm (*i.e.*, the arm that maximizes the product between its CTR and its revenue), and a Random algorithm, that chooses randomly the arms, such that all the arms have the same probability of being pulled. Notice that the Oracle is our revenue baseline as is the best possible, while the Random one is the worst possible.

For the dynamic scenario, we also used a static Oracle, that always continues to take decisions based on the initial CTR matrix. The static Oracle is useful to represent situations in which, after an initial exploration that allows to understand the preferences of users, decisions are always made on the initial knowledge of the world, hence maximizing the short-term revenue, but penalizing the long-term ones, which is what partially happens in real web recommender systems. This method shows the worst-case performance when you do not adapt to a dynamic world.

4.3 Evaluation Measures

Since we are interested in evaluating the relationship between the total revenue and the overall knowledge of the world, we employed two different evaluation metrics. The first one, R, is the total revenue. It is simply the sum of the products between the purchased items and their corresponding prices. In the dynamic environment, we also evaluated the percentage of revenue of each algorithm with respect to the revenue of the Oracle. We define this as Relative Revenue in percentage, that is, $RR = \mathrm{R}_{\mathrm{Algorithm}}/\mathrm{R}_{\mathrm{Oracle}}$.

To evaluate the overall knowledge of the system, instead, let E be the sum of the absolute value of the difference between the real value and the predicted value of the CTR, and let C be the sum of all CTRs. The evaluation metric is E/C in percentage, so is 0 if there is no difference between the predicted and the real CTRs and it is 100% if all the predicted CTRs are equal to 0. We call this measure the Knowledge Relative Error or KRE. E is computed using the absolute value because the direction of difference does not really matter, since we are not interested in understanding if algorithms underestimate or overestimate the CTRs. This is our proxy measure for

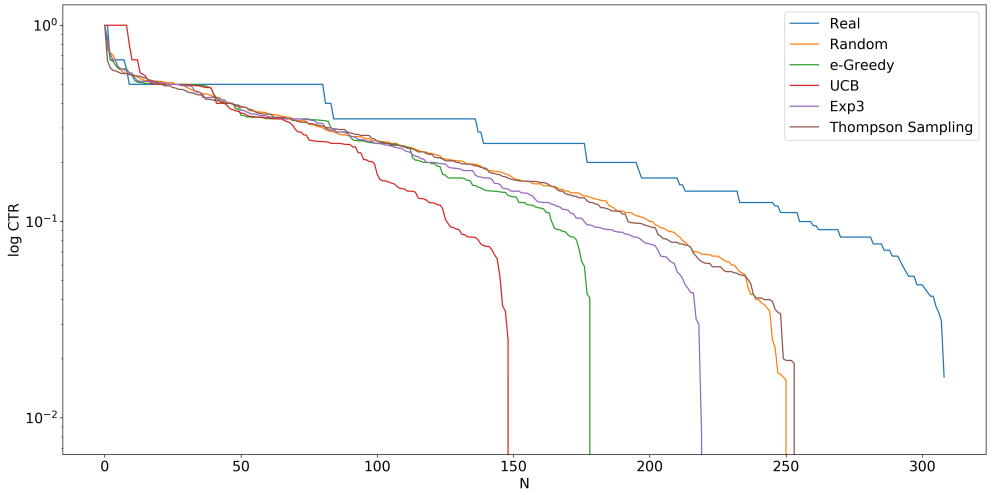

Fig. 3 Real CTR contrasted against predicted ones for a time horizon of 5,000 epochs considering DS1.

fairness as we can be more just in our decisions if we know the real CTRs. Notice that the Oracle knows the real world but uses a greedy approach as maximizes the revenue, so in that sense is the most unfair.

5 Experimental Results

The results, obtained in the static and in the dynamic world, are discussed in isolation in the following subsections.

5.1 Static World

In the static environment, we run the algorithms for each user in each data set, varying the time horizon (epochs) from 500 to 5,000. We began analyzing DS1. Figure 3 shows the predicted CTRs of all the exploration algorithms using a time horizon equal to 5,000. Here we can see that Random and Thompson sampling are the most fair (closer to the real distribution) while the Oracle is the most unfair (the curve shows the CTR that the Oracle learns while choosing to be greedy).

Tables 1 and 2 report the total revenue obtained and the knowledge gained by each algorithm with two horizon values. The best RRs and KREs among the employed algorithms are highlighted in bold, considering each distribution. Such results are reported since, in the next subsections, these algorithms drastically change their performance when the world is continuously changing.

The results are quite different for the two horizons, both, in terms of revenue and knowledge. With a large time horizon, it is possible to notice how the algorithms that explore more, such as Random and Thompson Sampling that have an error of 42% and 41%, respectively, are also the ones that get lower revenue (4M and 12M). Notice

Distribution	Real		Fair		PB	
Algorithm	**RR**	**KRE**	**RR**	**KRE**	**RR**	**KRE**
Random	0.4	74	0.3	79	0.3	73
TS	0.4	69	0.3	**71**	0.4	70
Exp3	4.4	70	3.4	**71**	7.6	**69**
UCB	7.7	**67**	**6.6**	**71**	**9.2**	71
ϵ-Greedy	**7.9**	72	6.2	74	6.2	**69**
Oracle	21.1	0	18.6	0	20.5	0

Table 1 Total revenue and Knowledge Relative Error for time horizon 500, considering a static world with DS1.

Distribution	Real		Fair		PB	
Algorithm	**RR**	**KRE**	**RR**	**KRE**	**RR**	**KRE**
Random	3.9	42	3.5	**42**	3.8	43
TS	11.5	**41**	9.3	**42**	12.6	**41**
ϵ-Greedy	135.9	51	115.3	52	130.3	52
Exp3	145.4	46	124.8	45	**150.9**	44
UCB	**159.3**	64	**129.9**	63	141.0	60
Oracle	210.7	0	188.0	0	209.0	0

Table 2 Total revenue and Knowledge Relative Error for time horizon 5,000, considering a static world with DS1.

Distribution	Real		Fair		PB	
Algorithm	**RR**	**KRE**	**RR**	**KRE**	**RR**	**KRE**
Random	0.4	**14**	0.3	21	0.4	**14**
TS	2.7	**14**	1.7	**19**	2.9	**14**
Exp3	5.7	**14**	4.0	21	3.9	**14**
ϵ-Greedy	42.2	52	35.8	57	42.2	49
UCB	**52.0**	16	**46.1**	26	**52.9**	17
Oracle	86.9	0	77.9	0	86.8	0

Table 3 Total relative revenue and knowledge relative error for time horizon 5,000, considering a static world with DS2.

also that UCB obtains the best revenue for the Real and the Fair distribution while Exp3 works better for the PB distribution.

This shows that in general there is low correlation between knowledge and revenue. For example, for a time horizon 5,000 and the Real distribution, the correlation between KRE and RE is -0.29 when we would expect -1. If we do not take into account the Oracle, the correlation jumps to 0.76, which shows that all the exploration techniques are correlated. A good trade-off between revenue and knowledge is achieved by Exp3, with an error of 46% but a revenue of 145M.

Then, the experiments are replicated using DS2. The time horizon is fixed to 5,000 epochs. The time horizon of 500 epochs was not considered, given the number of items (*i.e.*, 514) because such a number of epochs would not allow us to evaluate all pairs just once. Table 3 reports the total revenue obtained and the knowledge gained by each algorithm, always highlighting in bold the best RR and KRE among the employed algorithms with respect to the Oracle, considering each error distribution.

These results are somehow similar to the ones obtained with DS1 and considering a time horizon of 5,000 epochs, both, in terms of revenue and knowledge. Also in this case, the algorithms that explore more, such as Random and Thompson Sampling

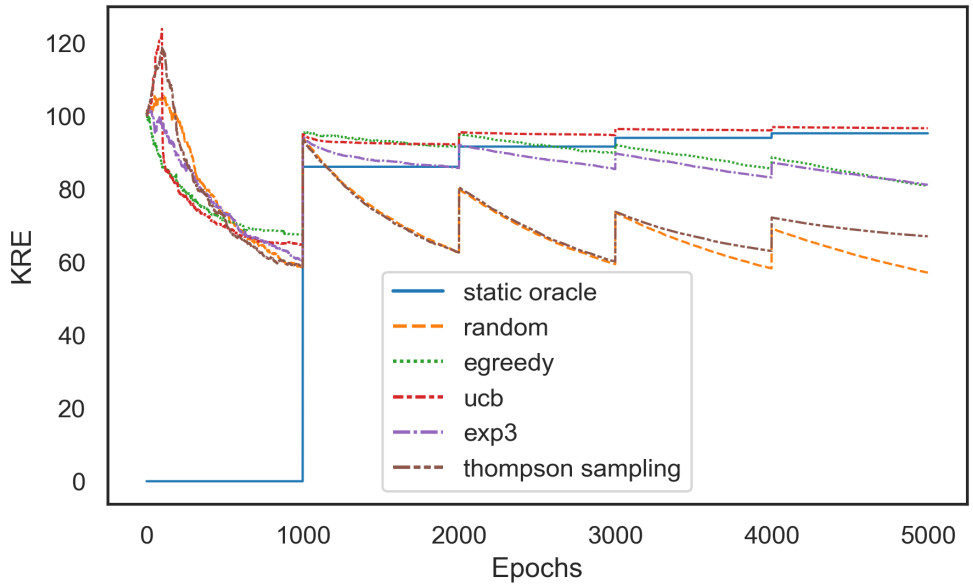

Fig. 4 Knowledge relative error for each algorithm using the Real distribution of DS1, varying the CTR matrix every 1,000 epochs.

have the lowest KREs. The highest R values, instead, are always achieved by the UCB algorithm, as before. It is interesting to notice, notice that it is also able to get very low KRE values that are close to the one of the algorithms that explore more. Finally, while in the previous experiment Exp3 achieved a high value for R, in this case the situation is the opposite. This may imply that some techniques are more sensitive to the environment than others.

5.2 Dynamic World

In the dynamic setting, the algorithms were executed for each distribution (Real, Fair, and PB), setting the time horizon to 5,000, varying the CTR matrix every 1,000 epochs. As detailed in the previous section, every 1,000 epochs random perturbations were added to the CTR matrix, making sure that for each user, the item that maximizes the product between its CTR and its price, changes.

Firstly, the results for DS1 are reported. For each experiment, there are the knowledge error of each algorithm at each epoch (Figures 4, 6, and 8) and the percentage of revenue obtained with respect to the Oracle (Figures 5, 7, and 9). Finally, Table 4 reports also the relative revenue obtained by each algorithm. Again the best RR and KRE among the employed algorithms with respect to the Oracle are highlighted in bold, considering each distribution. Unlike what happened in the static scenario, where users' preferences never changed, in a dynamic scenario the importance of exploration on long-term revenue can be observed. Apart from the Random algorithm, other algorithms are able to explore more obtaining higher revenues.

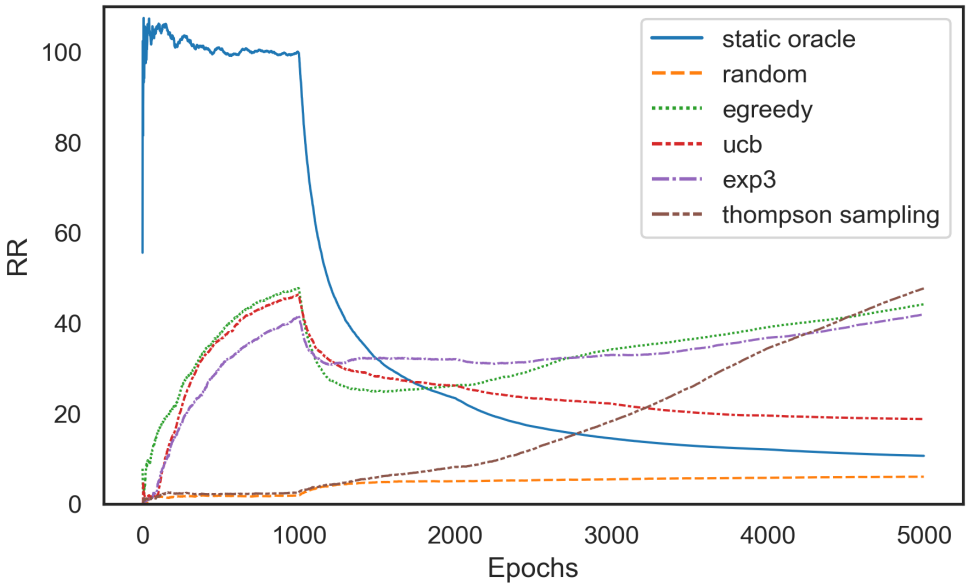

Fig. 5 Relative revenue for each algorithm using the Real distribution for DS1, varying the CTR matrix every 1,000 epochs.

Distribution	**Real**		**Fair**		**PB**	
Algorithm	**RR**	**KRE**	**RR**	**KRE**	**RR**	**KRE**
Random	6.0	**57.1**	6.0	**57.0**	5.9	**57.3**
Stat. Oracle	10.6	95.2	10.5	95.4	10.6	95.2
UCB	18.7	96.6	21.6	96.6	20.8	96.6
Exp3	41.9	81.1	39.8	81.4	43.4	81.3
ϵ-Greedy	44.1	80.8	43.7	80.2	43.0	80.7
TS	**47.6**	67.0	**48.2**	67.2	**47.8**	67.0
Oracle	100.0	0	100.0	0	100.0	0

Table 4 Results for time horizon 5,000, varying the CTR matrix every 1,000 epochs using DS1.

For all distributions, there are four groups with respect to the knowledge error: Random is the best followed by Thompson sampling. Then there is a cluster with Exp3 and ϵ-Greedy while the worst ones are UCB together with the Static Oracle. Regarding revenue, the algorithms react very differently to changes. UCB never recovers while Thompson sampling keeps improving, which may explain why this old algorithm is still popular, as it seems to be more robust to changes. On the other hand, Exp3 and ϵ-Greedy are able to recover and keep learning. The correlation between KRE and RR were also computed for each distribution and algorithm. Our hypothesis is that more knowledge implies more revenue (*i.e.*, better understanding of the world implies more revenue). Hence, such a correlation should be negative. This was confirmed by the results obtained. In fact, for all the distributions, the correlations between KRE and RR are all around -0.75, not including the Static Oracle, closer to what was

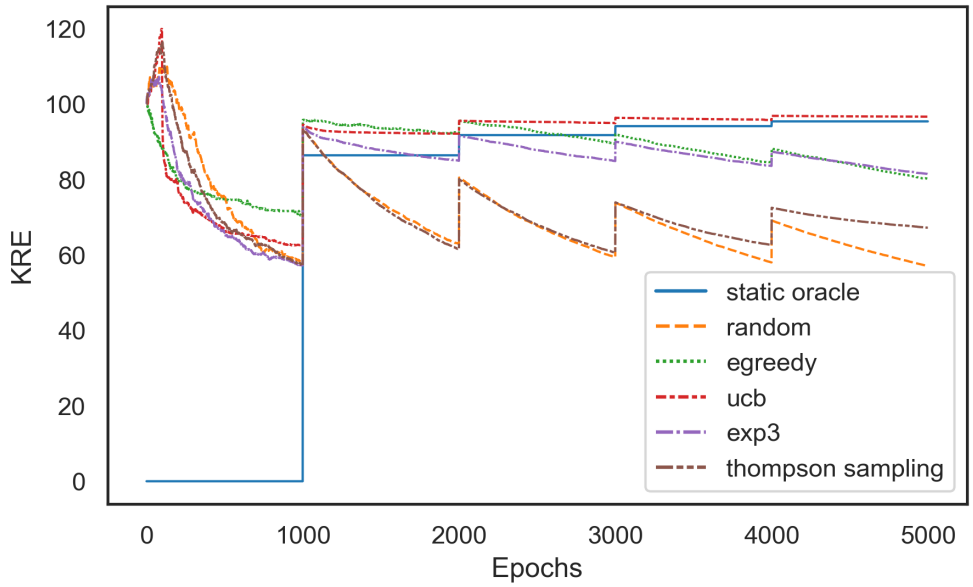

Fig. 6 Knowledge relative error for each algorithm using the Fair distribution for DS1, varying the CTR matrix every 1,000 epochs.

expected. However, if the full Oracle is not considered, the correlations change to 0.17, 0.20 and 0.21 for Real, Fair and PB, respectively. This shows that there is almost no correlation between how well you know the world and the final revenue, which strengthens one of the main hypotheses of this work. That is, optimizing revenue may lead to discrimination in the tail even with a fairer distribution.

In addition, no exploration algorithm reaches the 50% of the revenue of the Oracle, showing that all of them are far from optimal. That is, the lack of full knowledge of the world does not allow them to reach long-term optimal revenue. This certainly is not healthy for the digital market as all providers are affected and this may trigger market instability in the long run.

All the aforementioned experiments are replicated using DS2. For each experiment, the knowledge error of each algorithm at each epoch (Figures 10, 12, and 14) and the percentage of revenue obtained with respect to the Oracle (Figures 11, 13, and 15) are reported. Finally, Table 5 also reports the relative revenue obtained by each algorithm. Also, in this case, the highest RR and KRE among the employed algorithms, considering each distribution are highlighted in bold.

As shown, the KRE graphs are similar to the previous case. There are some small differences, since some algorithms performed better than others across the two data sets. For example, TS is able to get the lowest values, which are even less than the Random ones. The same cannot be said for revenue. In fact, while with DS1, all the algorithms surpassed the performance of the Static Oracle in terms of RR, with DS2 only the UCB perform better than it. This is mainly due to the number of items

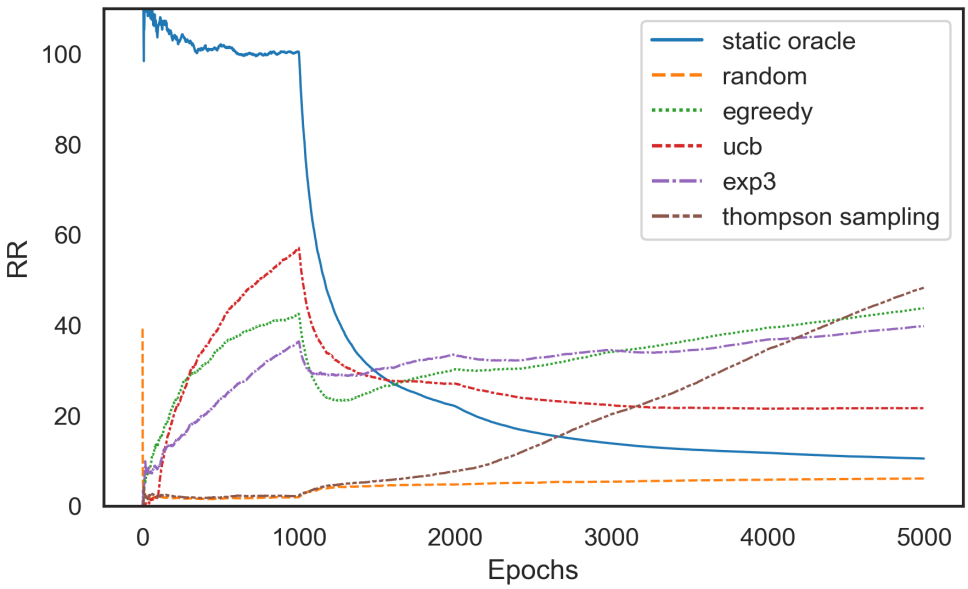

Fig. 7 Relative revenue for each algorithm using the Fair distribution for DS1, varying the CTR matrix every 1,000 epochs.

Distribution	**Real**		**Fair**		**PB**	
Algorithm	**RR**	**KRE**	**RR**	**KRE**	**RR**	**KRE**
Random	2.1	66.2	2.2	86.5	2.2	86.5
Stat. Oracle	52,1	94.3	48.2	94.9	50.8	94.4
UCB	**55.9**	82.9	**54.6**	91.8	**54.5**	91.9
Exp3	5.5	67.3	5.1	86.9	5.2	87.1
ϵ-Greedy	36.8	99.9	33.9	103.9	35.5	104
TS	5.9	**65.2**	4.3	**85.2**	4.1	**84.8**
Oracle	100.0	0	100.0	0	100.0	0

Table 5 Results for time horizon 5,000, varying the CTR matrix every 1,000 epochs using DS2.

in DS2, that is five times of DS1, hence requiring more time (*i.e.*, epochs) for the exploration. Anyway, in the end, it is clear the descending trend of the Static Oracle. With regard to the poor performance of different algorithms (TS and EXP3), they mainly derive from the fact that they never take advantage of the knowledge they have acquired before the CTR matrix changes.

To understand better how the exploration algorithms behave in different parts of the distribution, we compute the average and standard deviation of the relative error of the predicted CTR against the real CTR for the head (top 1/3 of the CTRs), the torso (middle 1/3 of the CTRs), and the tail (bottom 1/3 of the CTRs), which are shown in Table 6 for DS1.

The UCB is the algorithm with more error for the head and the torso. On the other hand, the worst error in the tail is for TS. Here we can also see that the error is

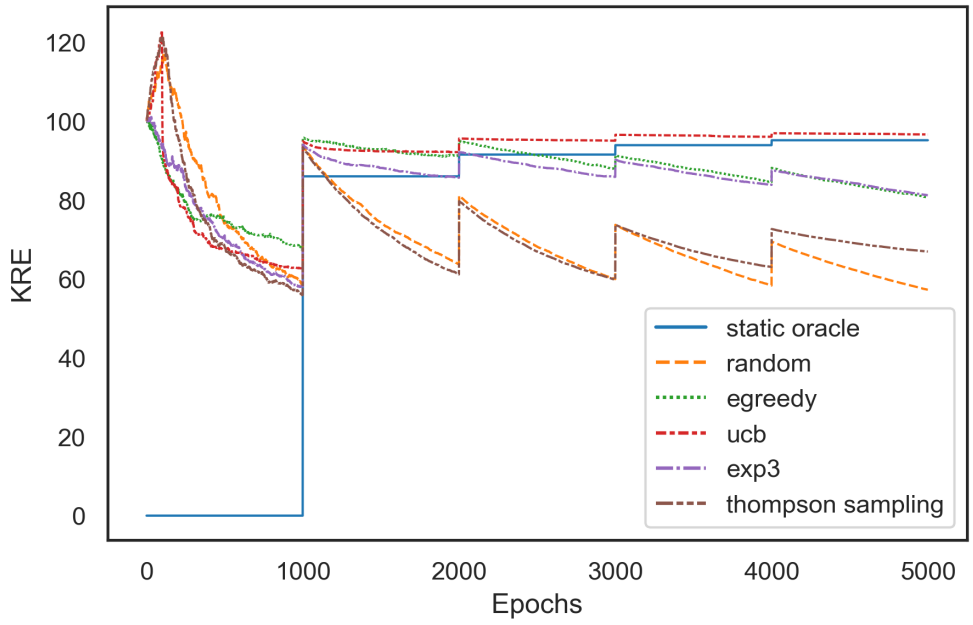

Fig. 8 Knowledge relative error for each algorithm using the PB distribution for DS1, varying the CTR matrix every 1,000 epochs.

larger for the Popularity Bias distribution, which is what we expect. However, for the other regions the differences are much smaller, stressing the fact that discrimination happens in the tail. Notice that for the Fair distribution, in the case of the torso, Exp3 is almost as bad as TS. Finally, the algorithm with less error in all regions is Random as there is no selection bias.

5.3 Limitations

Notwithstanding the results obtained in the experiments in both the static and dynamic environment, it is important to highlight the limitations of our work. The first one is that we used just two data sets. In fact, we could not find another public data set that had all the characteristics needed (many users, items and transactions as well as the price per item). The second limitation is the size of the data sets. In fact, we evaluated transactions for a limited number of users and items. However, larger data sets would just make the world more complex and hence the impact in revenue and knowledge would be larger, further emphasizing the results obtained. This is specially true if we add more less frequent items and users with low engagement. The third limitation concerns the changes used in the dynamic case. We do synchronized changes for all users and the sets of users and items remain the same throughout the epochs, in fact no user or item is added or removed. Yes, but notice that arbitrary changes will make the problem just more difficult. The fourth and final limitation concerns the exploration algorithms. We employed well-known algorithms because they

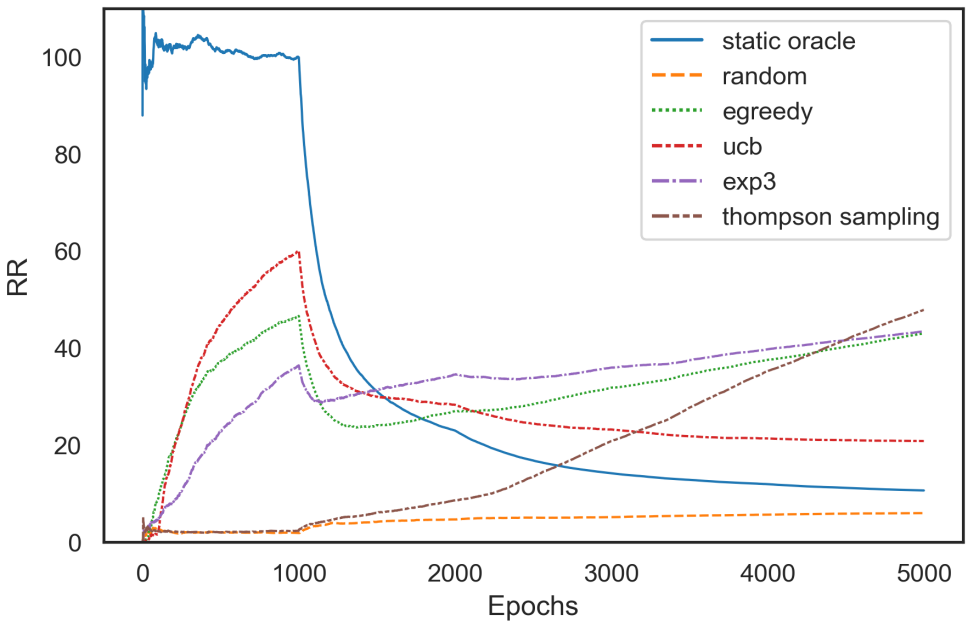

Fig. 9 Relative revenue for each algorithm using the PB distribution for DS1, varying the CTR matrix every 1,000 epochs.

are the ones used today by large companies and to show that the problem formulated in this work is real, relevant and that current solutions are sub-optimal. However, there might exist better algorithms that are not public.

6 Conclusions and Future Work

Our results show that there is no exploration algorithm that is the best for all cases, either considering error distributions that we want to achieve for the specific environment where the system works (that is, the data set used). This is not strange for problems that depend in so many factors. Hence, it is clear that better *ad-hoc* exploration algorithms need to be devised such that there is a better balance between how much you know about the world and the revenue obtained in a given period. In a counterfactual approach, we will need to consider the impact in the provider side when there is discrimination, such as sellers leaving the market and decreasing competition. In this case, we would need to define additional proxy fairness measures such as coverage of providers (*e.g.*, all of them should be exposed) and the level of discrimination (*e.g.*, all of them are shown proportional to their real CTRs, as CTRs can be considered a relevance proxy). Similarly, we need to consider the impact on the user experience if fairness affects relevance and satisfaction [26]. In a real scenario, the best approach would be to use contextual MABs based on deep learning [34] and/or meta-bandits [35], but for that we need to understand first the trade-offs involved.

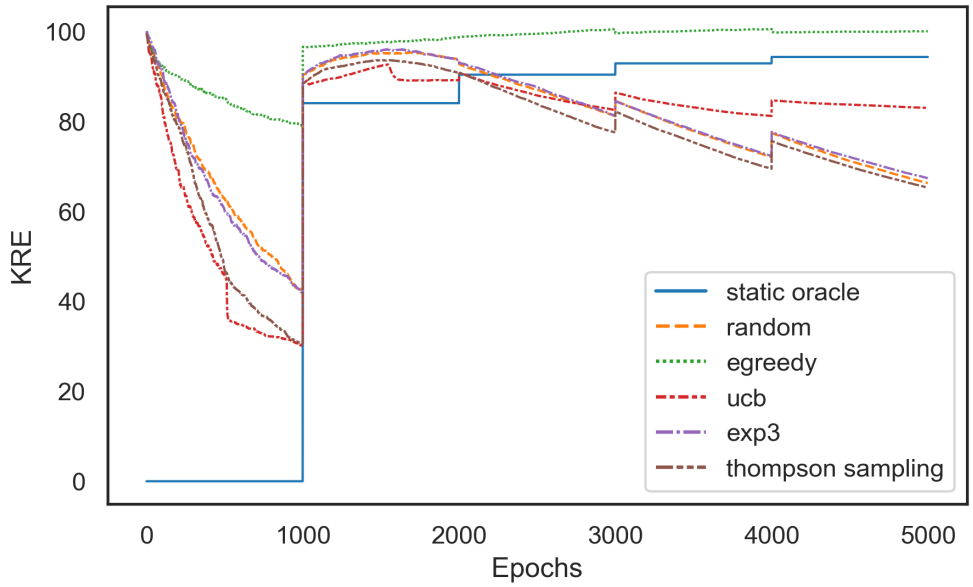

Fig. 10 Knowledge relative error for each algorithm using the Real distribution for DS2, varying the CTR matrix every 1,000 epochs.

Future work includes using additional (larger) data sets with even more dynamic scenarios including adding new users and new items, as well as other distributions that may improve the balance between the knowledge of the real world and revenue. We also need to analyze what is the impact of considering only frequent users/items in our results, as well as connect our measures to the standard regret used in arm bandits optimization. Finally, new algorithms that mitigate exposure bias as formulated in this research should be devised.

References

[1] Baeza-Yates, R.: Bias on the Web. Communications of the ACM **61**(6), 54–61 (2018) https://doi.org/10.1145/3209581

[2] Pariser, E.: The Filter Bubble: What the Internet Is Hiding from You. Penguin Press, New York, USA (2011)

[3] Agarwal, D., Chen, B.C., Elango, P.: Explore/Exploit Schemes for Web Content Optimization. In: Proceedings of the 2009 Ninth IEEE International Conference on Data Mining, pp. 1–10. IEEE Computer Society, Washington, DC, USA (2009)

[4] Barraza-Urbina, A.: The exploration-exploitation trade-off in interactive recommender systems. In: Proceedings of the Eleventh ACM Conference on Recommender Systems. RecSys '17, pp. 431–435. Association for Computing Machinery,

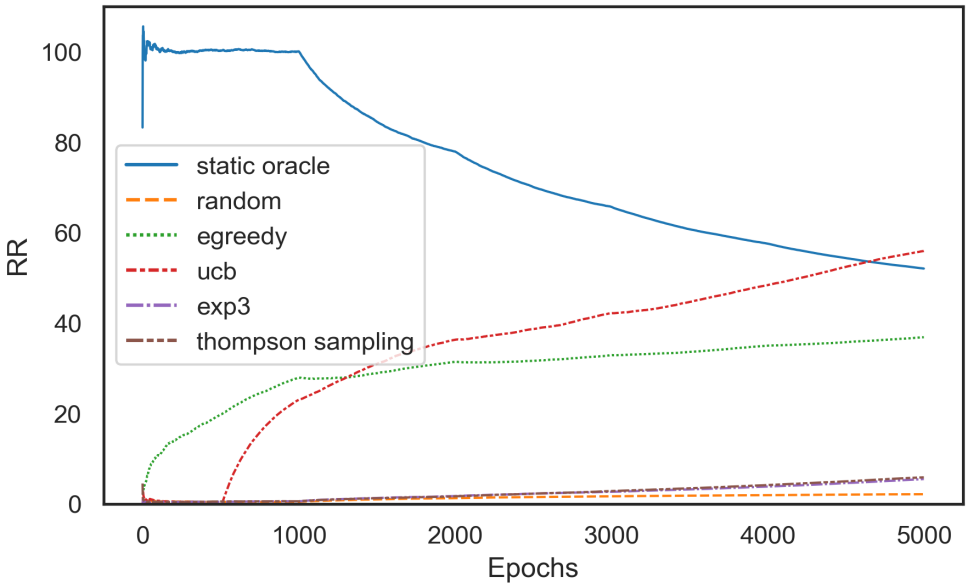

Fig. 11 Relative revenue for each algorithm using the Real Distribution for DS2, varying the CTR matrix every 1,000 epochs.

New York, NY, USA (2017). https://doi.org/10.1145/3109859.3109866

[5] Tsagkias, M., King, T.H., Kallumadi, S., Murdock, V., Rijke, M.: Challenges and research opportunities in ecommerce search and recommendations. SIGIR Forum **54**(1) (2020)

[6] Adomavicius, G., Bockstedt, J., Curley, S.P., Zhang, J., Ransbotham, S.: The hidden side effects of recommendation systems. MIT Sloan Management Review **60**(2) (2019)

[7] Thompson, W.R.: On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika **25**(3/4), 285–294 (1933)

[8] Robbins, H.: Some aspects of the sequential design of experiments. Bulletin of the American Mathematical Society **58**(5), 527–535 (1952)

[9] Jiang, R., Chiappa, S., Lattimore, T., György, A., Kohli, P.: Degenerate feedback loops in recommender systems. In: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, pp. 383–390. ACM, Honolulu, USA (2019)

[10] Zhou, L., Brunskill, E.: Latent contextual bandits and their application to personalized recommendations for new users. In: Twenty-Fifth International Joint

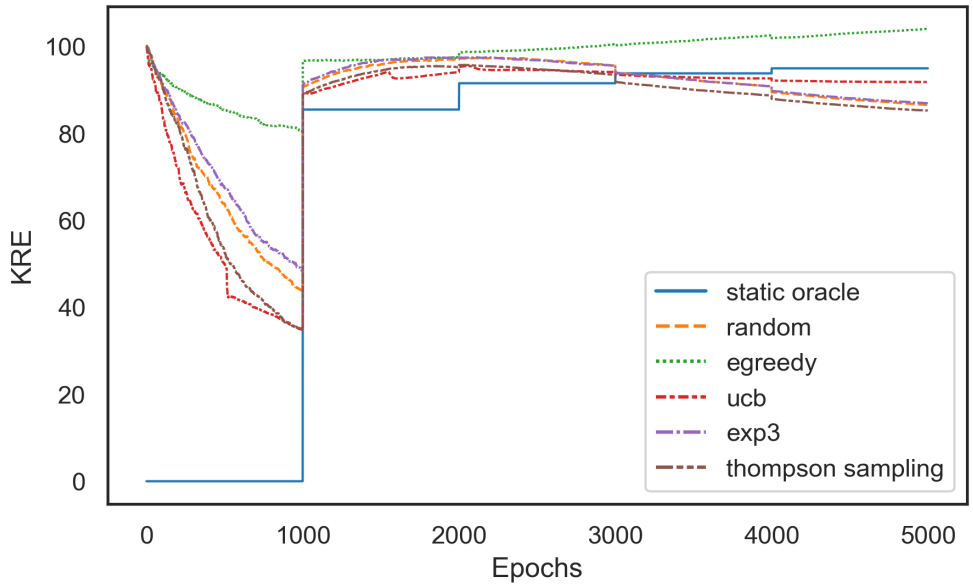

Fig. 12 Knowledge relative error for each algorithm using the Fair distribution for DS2, varying the CTR matrix every 1,000 epochs.

Conference on Artificial Intelligence, pp. 3646–3653 (2016)

[11] Mansoury, M., Abdollahpouri, H., Pechenizkiy, M., Mobasher, B., Burke, R.: Feedback loop and bias amplification in recommender systems. arXiv preprint arXiv:2007.13019 (2020)

[12] Cañamares, R., Castells, P.: Should i follow the crowd? a probabilistic analysis of the effectiveness of popularity in recommender systems. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 415–424 (2018)

[13] Abdollahpouri, H.: Popularity bias in ranking and recommendation. In: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, pp. 529–530 (2019)

[14] Abdollahpouri, H., Burke, R., Mobasher, B.: Controlling popularity bias in learning-to-rank recommendation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 42–46 (2017)

[15] Kiswanto, D., Nurjanah, D., Rismala, R.: Fairness aware regularization on a learning-to-rank recommender system for controlling popularity bias in e-commerce domain. In: 2018 International Conference on Information Technology Systems and Innovation (ICITSI), pp. 16–21 (2018). IEEE

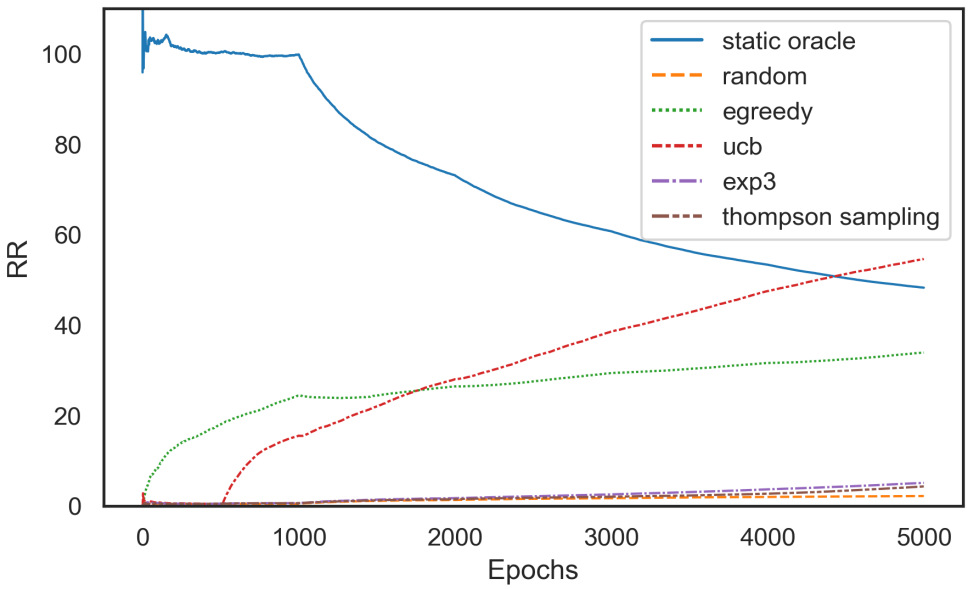

Fig. 13 Relative revenue for each algorithm using the Fair distribution for DS2, varying the CTR matrix every 1,000 epochs.

[16] Edizel, B., Bonchi, F., Hajian, S., Panisson, A., Tassa, T.: Fairecsys: Mitigating algorithmic bias in recommender systems. International Journal of Data Science and Analytics **9**(2), 197–213 (2020)

[17] Bellogín, A., Castells, P., Cantador, I.: Statistical biases in information retrieval metrics for recommender systems. Information Retrieval Journal **20**(6), 606–634 (2017)

[18] Khenissi, S., Nasraoui, O.: Modeling and counteracting exposure bias in recommender systems. arXiv preprint arXiv:2001.04832 (2020)

[19] Abdollahpouri, H., Mansoury, M.: Multi-sided exposure bias in recommendation. arXiv preprint arXiv:2006.15772 (2020)

[20] Kumar, B., Bala, P.K.: Fattening the long tail items in e-commerce. Journal of theoretical and applied electronic commerce research **12**(3), 27–49 (2017)

[21] Sreepada, R.S., Patra, B.K.: Mitigating long tail effect in recommendations using few shot learning technique. Expert Systems with Applications **140**, 112887 (2020)

[22] Basu, K., DiCiccio, C., Logan, H., Karoui, N.E.: A framework for fairness in two-sided marketplaces. arXiv preprint arXiv:2006.12756 (2020)

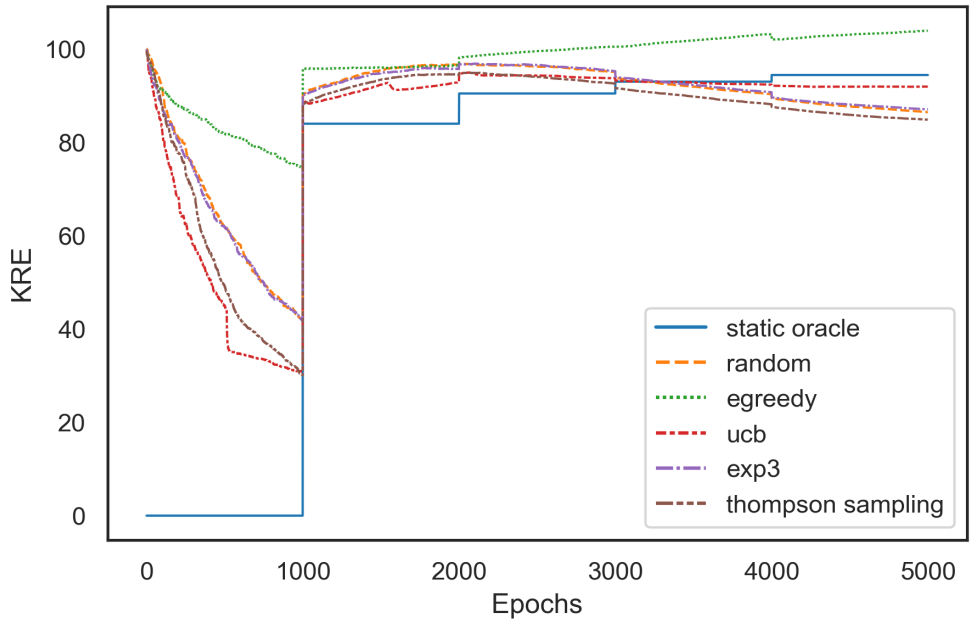

Fig. 14 Knowledge relative error for each algorithm using the PB distribution for DS2, varying the CTR matrix every 1,000 epochs.

[23] Wang, L., Joachims, T.: Fairness and diversity for rankings in two-sided markets. arXiv preprint arXiv:2010.01470 (2020)

[24] Deldjoo, Y., Anelli, V.W., Zamani, H., Bellogín, A., Di Noia, T.: Recommender systems fairness evaluation via generalized cross entropy. arXiv preprint arXiv:1908.06708 (2019)

[25] Nedelec, T., Roux, N.L., Perchet, V.: A comparative study of counterfactual estimators. arXiv preprint arXiv:1704.00773 (2017)

[26] Mehrotra, R., McInerney, J., Bouchard, H., Lalmas, M., Diaz, F.: Towards a fair marketplace: Counterfactual evaluation of the trade-off between relevance, fairness & satisfaction in recommendation systems. In: Proceedings of the 27th Acm International Conference on Information and Knowledge Management, pp. 2243–2251 (2018)

[27] Verma, S., Gao, R., Shah, C.: Facets of fairness in search and recommendation. In: Boratto, L., Faralli, S., Marras, M., Stilo, G. (eds.) Bias and Social Aspects in Search and Recommendation, pp. 1–11. Springer, Cham (2020)

[28] Chen, J., Dong, H., Wang, X., Feng, F., Wang, M., He, X.: Bias and Debias in Recommender System: A Survey and Future Directions (2020)

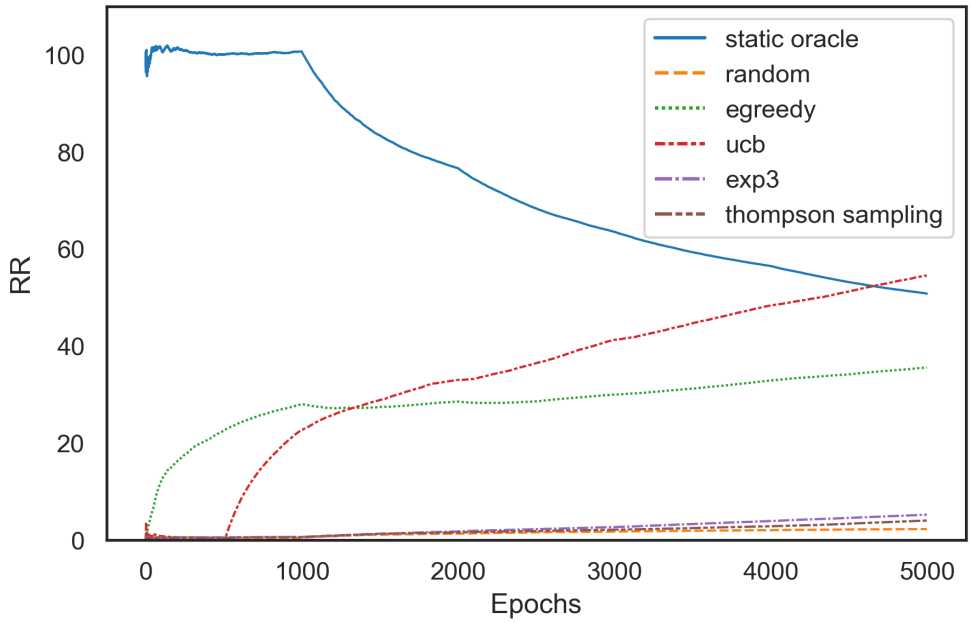

Fig. 15 Relative revenue for each algorithm using the PB distribution for DS2, varying the CTR matrix every 1,000 epochs.

[29] Chen, X.: Exact computation of minimum sample size for estimation of binomial parameters. Journal of Statistical Planning and Inference **141**(8), 2622–2632 (2011)

[30] Agresti, A., Coull, B.A.: Approximate is better than "exact" for interval estimation of binomial proportions. The American Statistician **52**(2), 119–126 (1998)

[31] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine learning **47**(2-3), 235–256 (2002)

[32] Garivier, A., Moulines, E.: On upper-confidence bound policies for switching bandit problems. In: International Conference on Algorithmic Learning Theory, pp. 174–188. Springer, Budapest, Hungary (2011)

[33] Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: The nonstochastic multiarmed bandit problem. SIAM Journal on Computing **32**(1), 48–77 (2002)

[34] Eide, S., Zhou, N.: Deep neural network marketplace recommenders in online experiments. In: Proceedings of the 12th ACM Conference on Recommender Systems, pp. 387–391 (2018)

Distribution		Real		Fair		PB	
Method	Region	Mean	SD	Mean	SD	Mean	SD
	Head	**0.96**	0.17	**0.96**	0.17	**0.96**	0.17
2-8 UCB	Torso	**0.99**	0.14	**0.99**	0.10	**0.99**	0.10
2-8	Tail	1.07	1.75	1.04	0.90	1.08	1.86
	Head	0.78	0.30	0.79	0.28	0.78	0.28
2-8 Exp3	Torso	0.85	0.30	0.83	0.30	0.85	0.30
2-8	Tail	1.45	8.16	*2.55*	41.88	2.01	34.76
	Head	0.54	0.24	0.54	0.24	0.55	0.24
2-8 Random	Torso	0.58	0.28	0.57	0.28	0.57	0.28
2-8	Tail	2.23	27.16	2.29	27.39	2.22	23.56
	Head	0.75	0.33	0.73	0.33	0.75	0.33
2-8 ϵ-Greedy	Torso	0.89	0.42	0.87	0.35	0.87	0.37
2-8	Tail	2.58	53.98	1.9	11.64	1.95	12.29
	Head	0.64	0.28	0.64	0.28	0.64	0.28
2-8 TS	Torso	0.68	0.32	0.7	0.30	0.68	0.32
2-8	Tail	**2.91**	57.46	**2.56**	36.36	**3.57**	86.78

Table 6 Average and standard deviation of the relative error in the CTR prediction for three different regions of all distributions and all exploration algorithms for DS1.

[35] Santana, M.R., Melo, L.C., Camargo, F.H., Brandão, B., Soares, A., Oliveira, R.M., Caetano, S.: Contextual meta-bandit for recommender systems selection. In: Fourteenth ACM Conference on Recommender Systems, pp. 444–449 (2020)