Offline Evaluation to Make Decisions About Playlist Recommendation Algorithms

Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney,
Samantha Hansen, Damien Tardieu, Ben Carterette
Spotify, New York, NY
{agruson | praveenr | charbuillet | jamesm | slhansen | dtardieu | benjaminc}@spotify.com

ABSTRACT

Evaluating algorithmic recommendations is an important, but difficult, problem. Evaluations conducted offline using data collected from user interactions with an online system often suffer from biases arising from the user interface or the recommendation engine. Online evaluation (A/B testing) can more easily address problems of bias, but depending on setting can be time-consuming and incur risk of negatively impacting the user experience, not to mention that it is generally more difficult when access to a large user base is not taken as granted. A compromise based on *counterfactual analysis* is to present some subset of online users with recommendation results that have been randomized or otherwise manipulated, log their interactions, and then use those to de-bias offline evaluations on historical data. However, previous work does not offer clear conclusions on how well such methods correlate with and are able to predict the results of online A/B tests. Understanding this is crucial to widespread adoption of new offline evaluation techniques in recommender systems.

In this work we present a comparison of offline and online evaluation results for a particular recommendation problem: recommending playlists of tracks to a user looking for music. We describe two different ways to think about de-biasing offline collections for more accurate evaluation. Our results show that, contrary to much of the previous work on this topic, properly-conducted offline experiments do correlate well to A/B test results, and moreover that we can expect an offline evaluation to identify the best candidate systems for online testing with high probability.

ACM Reference Format:
Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney, Samantha Hansen, Damien Tardieu, Ben Carterette. 2019. Offline Evaluation to Make Decisions About Playlist Recommendation Algorithms. In *The Twelfth ACM International Conference on Web Search and Data Mining (WSDM '19), February 11–15, 2019, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3289600.3291027

1 INTRODUCTION

Recommender systems are in wide use today, used for everything from recommending video and music to recommending products for purchase to recommending things to do and places to go. Recommendation system technology is now in search engines, advertising services, and even personal fashion assistance.

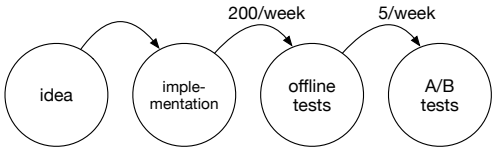


Figure 1: From idea to A/B test. Only a few ideas can ever be A/B tested, but offline evaluation is limited only by how fast we can implement our ideas. It is therefore important that offline evaluation results accurately reflect A/B test results.

Many of the use cases for recommendation concern contextual personalization: getting the right results to the right users at the right time. Evaluating this, however, can be challenging. Broadly speaking, there are two ways to evaluate recommender systems: online, using A/B tests to compare different approaches by their effect on logged user interactions, and offline, using some log of historical user interactions under the assumption that the users of that system would like the same items if they were served by different approaches.

It is hard to capture context in offline evaluations. Logged interactions are often biased by the user interface and by the recommendation engine itself: the engine is trained such that it will prefer certain types of content over others, and the types of content it prefers will naturally see more user interactions. This could be alleviated by building test collections with additional human judgments, but that is costly and moreover very difficult to do in a way that captures the personal nature of recommendations.

Online evaluation is an attractive alternative because it does not require such strong assumptions about user's interactions. It does, however, require a reasonably large user base and some time to collect enough data to be able to make a decision about the experimental systems being tested. Furthermore, there is risk in exposing users to results that are harmful in some way—even if the harm is just that the user is disinterested and leaves the platform. Offline evaluation prior to online evaluation can mitigate this by giving some indication of which experiments are likely to perform well online and which aren't.

Even with an initial online evaluation, it is not likely that we can take everything that performs well offline into an online test. Figure 1 illustrates the problem: we can often test hundreds of ideas each week just by changing feature sets and hyperparameters, but

WSDM '19, February 11–15, 2019, Melbourne, VIC, Australia

ACM ISBN 978-1-4503-5940-5/19/02…$15.00
https://doi.org/10.1145/3289600.3291027

what we can test online is always limited by time and the size of the user population. Thus we need an offline evaluation that not only separates good systems from bad, but can also help us make reliable decisions about which experimental systems are the best candidates to test online.

In this work we present a new study showing offline evaluation results can predict online A/B test results under the right approach to de-biasing. We present a simulation-based approach to deciding on the experimental systems with the highest potential for success in an A/B test. We also demonstrate how offline evaluation can be used to identify the most promising individual system components or characteristics.

The rest of this paper is organized as follows. In Section 2 we review previous and recent work on evaluation in recommender systems. Section 3 describes our specific problem setting and the evaluation challenges it presents. In Section 4 we compare offline and online evaluation results for 12 experimental runs, presenting analysis on relative rankings, statistical significance, and probability of identifying the experimental runs that perform best in online tests. We conclude in Section 5.

2 BACKGROUND

Evaluation of recommender systems has been [21] and still is [19] an important topic of ongoing research with several open questions. Since the early work by Resnick et. al [36] in 1994, evaluation has proceeded from predicting missing ratings of items in an offline setting to complicated models of user satisfaction using implicit online signals such as clicks, dwell time, etc [14, 13]. Other dimensions such as novelty, serendipity, diversity, etc. have been considered for evaluating recommendation quality as well [21, 25, 42, 16].

2.1 Online evaluation

A direct way of measuring recommendation quality is to measure the consumption of recommended items. In other words, a system that recommends music should probably focus on how often the recommendation leads to a stream. In this setting, online A/B tests are conducted to compare two systems by exposing them to real users [31]. A predefined metric is used to quantify recommendation quality (e.g., total stream per session). Recently, more sophisticated ways of modeling user satisfaction from implicit user feedback such as clicks, streams, etc. have been proposed in the context of music recommendations [14] and other domains [26].

Interleaving approaches that merge items from different engines have been used to limit the risk of poor recommendation [6]. Interleaving methods consist of two steps: (1) merging two or more ranked lists before presenting to the user; (2) interpreting interactions on the interleaved results. Several variants has been proposed in the literature, including *team-draft interleaving* [35], probabilistic interleaving [23], etc. (see [22] for a complete overview).

Reliably comparing systems in an online setting can take a considerable amount of time. In practice, A/B tests might need to be run for several weeks to get enough data to detect a statistically significant effect. This is particularly true when there is periodicity in user behavior that may affect metrics. Further, transforming every idea into a production-ready system to experiment online is prohibitively expensive. Thus there is some limit on the number of A/B tests that can be run depending on the total size of the user population, the size of samples needed for each cell, and the amount of time the tests need to run for. Additionally, we likely do not want to A/B test *every* potential system we can think of. Many are simply going to perform poorly, and it would be better to know that in advance so that we do not risk exposing users to them. Thus, A/B testing is not suitable for many common evaluation cases.

2.2 Offline evaluation

It is common in the academic study of recommender systems and search to use offline test collections for experimentation. In search, the TREC initiative has produced large test collections that are reliable for testing new models for search. These collections involve a large cost in acquiring human assessments of the relevance of documents to queries which has traditionally been infeasible for academics. But they do not require any search systems to be running online. Instead, they rely on expert assessors to develop search needs and judge the relevance of documents retrieved by experimental systems offline [43]. Such collections are rare for recommender systems because it is difficult for assessors to judge relevance when it is personal and contextual. However, some recent work has applied ideas from personalized recommendation to search with experiments on large human-annotated collections such as the MSLR-WEB30k data [41].

A possible compromise between fully-online and fully-offline evaluation with test collections is offline evaluation using historical data collected online. This involves curating a dataset using historical logs generated from users interacting with a recommender system, including "ground truth" inferred from interactions such as ratings or clicks. This dataset is used to evaluate experimental systems in an offline setting, i.e., without having to expose the experimental system to real users. This has become the favored approach for offline evaluation of recommender systems.

Early on, the focus was on predicting ratings that the user would provide to an item. In this setting, *error-based metrics* such as mean absolute error (MAE) and root mean squared error (RMSE) are commonly used to measure the prediction accuracy. As recommender systems became more sophisticated with UI advancements, there was a need to take into account the user's natural browsing behavior [19]. *Rank-based metrics* that gave more importance to the top-k results were used to evaluate systems [10, 3, 39]. Metrics such as *precision*, *recall*, and *mean-average-precision* commonly used in information retrieval were adopted [1]. Other IR metrics such as normalized discounted cumulative gain (nDCG) [27] and mean reciprocal rank (MRR) have also been used to evaluate systems [1].

Incorporating historical log data for evaluation in practice is challenging due to various biases present. Some of the most common datasets used in recommender systems research such as MovieLens, Last.FM, etc [20, 4] suffer from biases due to their UI or underlying recommendation algorithms, which the researchers have no control over. Examples of biases include position bias [28] (that users are more likely to pay attention to top-ranked items even if they are not as relevant), popularity bias (that many systems have a tendency to promote more popular content), attractiveness bias [45], trust bias [30], and what we call "selection bias", that is the propensity of the system to lean towards particular types of content due to

how it was trained or how content is modeled. A commonly used approach to handle bias in historical log data is to model user's behavior when interacting recommendation results and then use them to minimize the effect of the bias in click data [12, 7, 8].

However, studies have pointed out that *error-based*, *rank-based* and other metrics computed on offline dataset do not correlate with online evaluation results [2, 38, 15, 17] and researchers have questioned validity of offline evaluation.

2.3 Counterfactual analysis

Recently, approaches based on importance sampling [24] have been proposed for evaluating recommender systems in an unbiased manner [32, 41]. The proposed techniques are based on propensity weighting originally used in causal inference [37], and more recently in other domains such as IR evaluation [29, 44] and computational advertising [5]. These approaches rely on propensity scores that are computed and logged. In order to ensure data is collected on combinations that wouldn't happen with a production system, some random exploration is often included.

In this setting, the estimator uses the logs collected using a production policy μ to compute the expected reward of a target policy π. However, the importance sampling estimator and its variants such as capped importance sampling [5], normalized importance sampling [34] and doubly robust estimation [11], suffer from high variance. Swaminathan et al. [40] proposed the normalized capped importance sampling (NCIS) estimator using control variates to address this issue. Gilotte et al. [18] successfully demonstrated the use of NCIS to evaluate recommender systems.

In this work, we present a new study showing offline evaluation results can predict online A/B test results under the right approach to de-biasing. We present a simulation-based approach to deciding on the experimental systems with the highest potential for success in an A/B test. We also demonstrate how offline evaluation can be used to identify the most promising individual system components or characteristics.

3 PROBLEM SETTING

The specific recommendation problem we are considering is illustrated in Figure 2: a user comes to a recommender system (in this case, a music recommendation system) via a mobile interface and is presented with recommendations for playlists, called *cards* in this setting, organized into thematically-related *shelves*. The user can scroll up and down to reveal shelves, and left and right within a shelf to see additional cards.

The presentation of both shelves and cards is personalized to user and context. One user's top shelf may not be the same as another user's, and within a given shelf, the top cards recommended to one user may not be the same as those recommended to another user, and the same user may not see the same layout if they visit the page a second time. Importantly, the set of cards available for recommendation on this page is relatively small: about 200 algorithmic or editorially-curated playlists in addition to playlists from the user's recent listening history, each of which is manually assigned to one or more shelves.

In principle, the goal of the personalized recommendation engine is to assemble a layout of shelves and cards that maximizes user satisfaction. This problem is made more tractable in practice by assuming that cards can be selected independently of one another as well as independently of shelves, and the full layout can be assembled post hoc from a simple ranking of cards. Thus the goal of the personalized recommendation engine is to learn a reward function $f : C \times \mathcal{X} \rightarrow \mathbb{R}$ that accurately predicts the likelihood of a user engaging with a recommended card $c \in C$ for a given context $x \in \mathcal{X}$. Given that reward function, we wish to identify a policy $\pi(C = c|x \in \mathcal{X})$ for picking a card to be presented to the user/context x such that total reward $V = \sum_i f(c_i, x_i) \cdot \pi(c_i|x_i)$ is maximized. In this work, we adopt a contextual bandits approach similar to the one proposed by McInerney et al. [33]. A contextual bandit requires that we specify a reward model, a representation of context x, an action space, and a policy for selecting an action given a context.

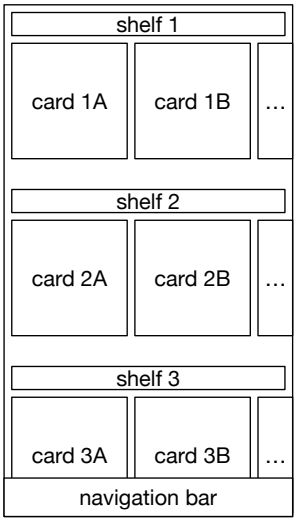


Figure 2: Illustration of a recommender system mobile interface that organizes recommendations into "shelves" of related "cards". Users can scroll left and right within shelves to reveal more cards, and up and down to reveal more shelves.

The reward model we currently use is impression-to-stream, that is, whether a card was seen by the user, and if so, whether the user streamed at least one track from the playlist represented by the card for at least 30 seconds. This is a binary rather than real-valued reward, which we use for simplicity; we note that it can be replaced by more complex measures of user satisfaction.

The context is essentially the user request. It is typically represented using features such as user demographic characteristics, the time of the request, the affinity the user has for particular tracks, the user's recent play history, and other contextual features.

The action space is the set of cards C.We simplify the problem by assuming that the set of cards for the context is static[1]. Furthermore, we have a set of manual constraints specifying the cards that are to be grouped into each shelf. Shelves are not explicitly part of the action space; instead, the policy is based on a machine-learned model that scores cards and uses those scores along with the shelf

[1]In reality this is not the case, as new items can become available for recommendation and old ones can be rotated out. This is a direction we wish to explore in the future.

constraints to rank shelves. In this way the selection of shelves follows deterministically from the scoring of cards[2].

There are several factors we can manipulate to optimize impression-to-stream rate: the feature sets that represent the user context, the particular machine learning models that score cards, the way a model is instantiated and optimized, the model hyperparameters, and so on. Identifying the best combination of variables for a recommendation policy demands a reliable evaluation methodology.

3.1 Online evaluation

Different approaches to selecting the factors listed above can be compared by a set of A/B tests: given several possible sets of factors, a test cell is configured for each one and each user is randomly assigned to a test cell or to a control group. Users in a particular test cell see recommendations based on a particular policy, with about 1% of users assigned to each cell. We record relevant information about cards (such as features, scores, shelves, etc.) and user interactions (impressions, streams, etc). After some time has passed we can compare each test cell on various metrics. As mentioned above, our reward is impression-to-stream, so impression-to-stream rate is the metric we are interested in.

3.2 Offline evaluation with historical logs

The offline evaluation problem is to estimate mean reward $V = \frac{1}{n}\sum_i^n r_i \cdot \pi(c_i|x_i)$ using a historical log of length n.

We obtain historical logs from systems running in production. Production systems are performing some level of exploration, which is captured in the policy. As we described in Section 2.2, these logs are biased; in order to use them for offline evaluation, they need to be de-biased in some way. We explore three estimators for V:

(1) the inverse propensity score estimate, also known as the importance sampling estimate (IS);
(2) a capped importance sampling estimate (CIS);
(3) a normalized and capped importance sampling estimate (NCIS).

In each case, the system being logged has a particular policy which we call the *logging policy* and denote $\mu(c_i|x_i)$. The goal of offline evaluation is to estimate the value of a new target policy π to be evaluated against the logging policy μ.

3.2.1 Importance sampling. The importance sampling estimator for V is based on re-weighting rewards by the propensity weight, which is the ratio of the target policy probability to the logging policy probability. In particular:

$$V_{\text{IS}} = \frac{1}{n}\sum_{i=1}^{n} r_i \cdot \frac{\pi(c_i|x_i)}{\mu(c_i|x_i)}$$

This estimator is unbiased in expectation, but its variance is high, as it grows polynomially with the size of the action space.

3.2.2 Capping propensities. When propensities are taken from a production log that is performing exploration, some values may be very low because random exploration produces layouts that are unlikely to be selected otherwise. These low propensities can have an outsized effect on the estimator. For example, if a card selected by the logging policy with propensity 0.001 has much higher probability of being selected by the target system—say 0.5 or more—the reward for that card could be multiplied by a factor of over 500. This is particularly a problem when interactions are sparse, as just a handful of low-propensity cards that received a user stream end up dominating the estimator.

One way to solve this problem is to cap the propensity weight $\frac{\pi(c_i|x_i)}{\mu(c_i|x_i)}$. The capped importance sampling (CIS) estimator introduces a parameter λ which is the maximum value we allow the propensity weight can take:

$$V_{\text{CIS}} = \frac{1}{n}\sum_{i=1}^{n} r_i \cdot \min\left(\lambda, \frac{\pi(a_i|x_i)}{\mu(a_i|x_i)}\right)$$

3.2.3 Normalizing the estimator. Another common technique to reduce variance is to normalize the CIS estimator by the sum of the propensity weights. This is normalized capped importance sampling (NCIS):

$$V_{\text{NCIS}} = \frac{\frac{1}{n}\sum_{i=1}^{n} r_i \cdot \min\left(\lambda, \frac{\pi(c_i|x_i)}{\mu(c_i|x_i)}\right)}{\frac{1}{n}\sum_{i=1}^{n} \min\left(\lambda, \frac{\pi(c_i|x_i)}{\mu(c_i|x_i)}\right)}$$

Note that while V_{IPS} is unbiased in expectation, V_{CIS} and V_{NCIS} introduce bias in order to help control variance and to allow the use of more of the data from production logs.

3.2.4 Fully-shuffled logs. Finally, a completely orthogonal way to address bias is by making sure some subset of users see recommendations unbiased by the UI or the recommendation engine. The simplest way to do this is to simply shuffle the cards and show users a completely random layout. This ensures that over the group of users exposed to this treatment there is no bias from the UI or from the engine. This randomization comes at a cost: the user experience is almost certainly going to be worse, and that may translate to longer-term losses. We can therefore only send a small fraction of traffic to fully-shuffled results.

3.3 Deciding on systems to test online

As we wrote above, not all experiments that can be tested offline can or should be tested online. A good offline evaluation should help us select those that are the best candidates for online testing. Since our goal is to improve recommendations overall, the best candidates for online testing are the systems that perform the best in offline evaluations, so we want our offline evaluations to be reliable.

We adopt a separate policy for selecting experiments to test. This policy is based on simulating an offline evaluation over many different logs to determine which experiments are most likely to rank highly. Since variance is high, we do not necessarily expect the same experiments to consistently rank at the top; given a distribution of positions in a ranking for each experiment, we can sample from this distribution experiments to deploy in A/B tests.

As noted above, all of the offline evaluations are done using a single log. This means the estimated rewards of experiments are highly correlated: two experiments are more likely than not to preserve their ordering relative to one another when evaluated against a new sample offline. We can assume rewards are sampled from some distribution that captures this correlation, then simulate

[2] Although the policy may allow for some random exploration as well.

new evaluations by sampling from that distribution repeatedly and seeing which systems come out on top.

We assume a multivariate normal distribution with mean vector $\widehat{\mu}$ estimated from the mean estimated rewards from a large offline evaluation and covariance matrix $\widehat{\Sigma}/n$ estimated from the covariance between each pair of experiments and divided by the log size. Mean estimated reward r is sampled from the normal distribution:

$$r \sim \mathcal{N}(\widehat{\mu}, \widehat{\Sigma}/n)$$

The simulated ordering experiments is the ordering of sampled rewards. Comparing simulated rankings over many of these trials, we can estimate a probability distribution over ranks for each experiment in the evaluation.

4 EXPERIMENT

In this section we present results and analysis of offline experiments on different recommendation algorithms with the goal of selecting the best subset for online testing.

4.1 Playlist recommendation algorithms

We compare 12 different recommendation methods. Algorithms differ on three dimensions:

- feature set used (two different feature sets);
- source of training data (raw biased logs vs. logs debiased by the NCIS estimator vs. fully-shuffled logs);
- hyperparameter values and modeling decisions.

We number experiments 1–12. Experiment 1 is the baseline.

These algorithms were tested online for a period during the summer of 2018. We have logs from these online tests, as well as logs from production systems that are not identical to these A/B tested systems. We also have a small log of user traffic that saw fully-shuffled results as described above.

4.2 Metrics and gold standard

Evaluation of the online tests is by impression-to-stream rate, which is 1 if a card was seen by a user and at least one track from the corresponding playlist was streamed for at least 30 seconds, and 0 if the card was seen but no track was streamed for 30 seconds. We denote this V. The online test results are the gold standard.

An offline evaluation is attempting to estimate what impression-to-stream rate would have been had the target policy been deployed. We use the V_{IS}, V_{CIS}, V_{NCIS} estimators defined above.

When reporting results, we normalize impression-to-stream rate and its estimates by the baseline (experiment 1) value, so that the baseline experiment will always receive a score of 1.000 in each online and offline evaluation. The other experiments will be evaluated by how many times better they are than the baseline.

4.3 Statistical testing

Here we briefly discuss statistical testing. Typically an A/B test is started with the goal of detecting a statistically significant effect when it is finished. Since A/B tests are done online by randomly sampling the user population to receive the treatment or control groups, the correct statistical testing procedure is a *two-sample* or *unpaired* test. Offline experiments, on the other hand, are done by computing the metric repeatedly on the same historical sample, and thus statistical significance testing can be done offline using *one-sample* or *paired* tests. Paired tests have the advantage of requiring fewer samples to find a significant effect.

The t-test is a common test that has both unpaired and paired variants, but sometimes criticized for requiring the data to conform to a normal distribution. An alternative is the bootstrap test, which involves sampling with replacement from the data to form a distribution of the mean. Bootstrapping is slow when sample sizes are large, so we would like to avoid bootstrapping if we can. To validate whether we could rely on the much more computationally efficient t-test, we compared bootstrap estimates of mean and variance to standard frequentist estimates. There is no difference between them, meaning we incur no loss in validity by using the t-test.

				stat. signif. vs experiment											
exp	V	std.err$\times 10^5$	$n/10^6$	1	2	3	4	5	6	7	8	9	10	11	12
1	1.000	0.132	47.51	·	★	★	★	★	★	★	★	★	★	★	★
2	1.167	0.147	45.07	★	·	★	★	★	★	★	★	★	★	★	★
3	1.340	0.153	47.36	★	★	·	★	★	★	★	★	★	★	★	★
4	1.380	0.158	45.91	★	★	★	·	★	★	★	★	★	★	★	★
5	1.386	0.158	46.41	★	★	★	★	·	·	★	★	★	★	★	★
6	1.391	0.157	46.86	★	★	★	★	·	·	★	★	★	★	★	★
7	1.421	0.157	47.96	★	★	★	★	★	★	·	★	★	★	★	★
8	1.453	0.166	43.84	★	★	★	★	★	★	★	·	★	★	★	★
9	1.472	0.163	45.83	★	★	★	★	★	★	★	★	·	★	★	★
10	1.487	0.162	47.12	★	★	★	★	★	★	★	★	★	·	★	★
11	1.494	0.162	47.07	★	★	★	★	★	★	★	★	★	★	·	★
12	1.559	0.164	47.81	★	★	★	★	★	★	★	★	★	★	★	·

Table 1: Relative ordering of 12 experimental systems tested online, measured by how many times better the impression-to-stream rate is to the baseline experiment 1. The grid of ★s indicates statistical significance: a pair of experiments (i, j) are statistically significantly different if and only if there is a ★ in cell (i, j) in the grid.

4.4 Online vs offline evaluation

Table 1 shows the relative ordering of experimental cells after the online test by normalized impression-to-stream ratio, along with statistical significance (by a two-sample t-test) between each pair of experiments. We consider this the "gold standard". Each online test is attempting to match this table as closely as possible.

4.4.1 Offline evaluation with CIS and NCIS. Table 2 shows the relative ordering of experimental cells by one baseline offline evaluation setting using the CIS estimator with a high cap (λ =1,000,000) and no normalization. Since the cap is so high, this is similar to an un-capped estimate. We note the following:

(1) The ordering of experiments differs substantially from the online experiment. The Kendall's τ rank correlation between the two is 0.424. This is not statistically significant, which means we cannot rule out the possibility that the offline evaluation is just ordering experiments randomly.
(2) Sample size is identical for every experiment. This is because each experiment is evaluated using the same offline data.
(3) Standard errors in this table are much higher than in Table 1: Table 1 reports standard errors $\times 10^5$, while this table reports them $\times 10^1$—the latter are 10^4 times larger than the former!

				stat. signif. vs experiment											
exp	V_{CIS}	std.err×10	$n/10^6$	1	3	2	9	6	8	5	7	12	11	10	4
1	1.000	0.262	1.34	·	·	·	·	·	★	★	★	★	★	★	★
3	1.447	0.205	1.34	·	·	·	·	·	★	·	★	★	★	★	★
2	1.450	0.167	1.34	·	·	·	·	·	·	·	★	★	★	★	★
9	1.522	0.195	1.34	·	·	·	·	·	·	·	·	★	★	★	★
6	1.539	0.229	1.34	·	·	·	·	·	·	·	·	·	★	★	★
8	1.906	0.268	1.34	★	★	·	·	·	·	·	·	·	·	·	★
5	1.945	0.263	1.34	★	·	·	·	·	·	·	·	·	·	★	·
7	1.981	0.261	1.34	★	★	★	·	·	·	·	·	·	·	·	·
12	2.008	0.266	1.34	★	★	★	★	·	·	·	·	·	·	·	·
11	2.126	0.272	1.34	★	★	★	★	★	·	·	·	·	·	·	·
10	2.603	0.339	1.34	★	★	★	★	★	·	★	·	·	·	·	·
4	2.608	0.288	1.34	★	★	★	★	★	★	·	·	·	·	·	·

Table 2: Relative ordering of 12 experimental systems evaluated offline by the CIS estimate ($\lambda = 10^6$) of how many times better the impression-to-stream rate is to the baseline, with statistical significance outcomes.

				stat. signif. vs experiment											
exp	V_{CIS}	std.err×10	$n/10^6$	1	6	3	5	9	12	2	7	8	11	10	4
1	1.000	0.014	1.34	·	★	★	★	★	★	★	★	★	★	★	★
6	1.968	0.019	1.34	★	·	★	★	★	★	★	★	★	★	★	★
3	2.089	0.020	1.34	★	★	·	★	★	★	★	★	★	★	★	★
5	2.369	0.022	1.34	★	★	★	·	·	·	★	★	★	★	★	★
9	2.386	0.021	1.34	★	★	★	·	·	·	★	★	★	★	★	★
12	2.434	0.021	1.34	★	★	★	·	·	·	·	·	★	★	★	★
2	2.510	0.022	1.34	★	★	★	★	★	·	·	·	·	★	★	★
7	2.528	0.022	1.34	★	★	★	★	★	·	·	·	·	★	★	★
8	2.600	0.022	1.34	★	★	★	★	★	★	·	·	·	★	★	★
11	2.762	0.023	1.34	★	★	★	★	★	★	★	★	★	·	·	★
10	2.824	0.024	1.34	★	★	★	★	★	★	★	★	★	·	·	★
4	3.167	0.025	1.34	★	★	★	★	★	★	★	★	★	★	★	·

Table 3: Relative ordering of 12 experimental systems evaluated offline by the CIS estimate ($\lambda = 100$) of how many times better the impression-to-stream rate is to the baseline, with statistical significance outcomes.

(4) Because the standard errors are so much higher, statistical significance signals are lost en masse. Thirty-seven pairs that were significantly different in the online evaluation are not significant in the offline evaluation.
(5) The best experiment by the online evaluation—experiment 12—is only 4th-best by the offline evaluation, and is statistically indistinguishable from the 5th-worst experiment.
(6) The best experiment by the offline evaluation—experiment 4—is 4th-*worst* by the online evaluation.

Table 2 represents one possible case for the offline evaluation with de-biased data. Table 3, with $\lambda = 100$ and no normalization, represents something close to a worst-case scenario. The ranking is inaccurate (Kendall's tau correlation of 0.333), the best experiment by the online gold standard is ranked in the middle, and the fourth-worst experiment is ranked highest. Yet the confidence in results is high: most pairs are thought to be statistically significantly different from one another. Table 4, with normalization and $\lambda = 10^5$, represents a much better outcome. The ranking is more accurate (Kendall's tau of 0.636, which is significant), and moreover the best two experiments are ranked in the top two positions. The

				stat. signif. vs experiment											
exp	V_{NCIS}	std.err×10	$n/10^6$	1	2	4	7	5	10	3	9	6	8	12	11
1	1.000	0.102	1.34	·	·	·	★	★	★	★	★	★	★	★	★
2	1.454	0.065	1.34	·	·	·	·	·	★	★	★	★	★	★	★
4	1.495	0.064	1.34	·	·	·	·	·	★	★	★	★	★	★	★
7	1.745	0.089	1.34	★	·	·	·	·	·	·	·	★	★	★	★
5	1.934	0.101	1.34	★	·	·	·	·	·	·	·	·	·	·	★
10	2.073	0.105	1.34	★	★	★	·	·	·	·	·	·	·	·	·
3	2.154	0.118	1.34	★	★	★	·	·	·	·	·	·	·	·	·
9	2.168	0.108	1.34	★	★	★	·	·	·	·	·	·	·	·	·
6	2.346	0.135	1.34	★	★	★	★	·	·	·	·	·	·	·	·
8	2.540	0.138	1.34	★	★	★	★	·	·	·	·	·	·	·	·
12	2.589	0.133	1.34	★	★	★	★	·	·	·	·	·	·	·	·
11	2.684	0.133	1.34	★	★	★	★	★	·	·	·	·	·	·	·

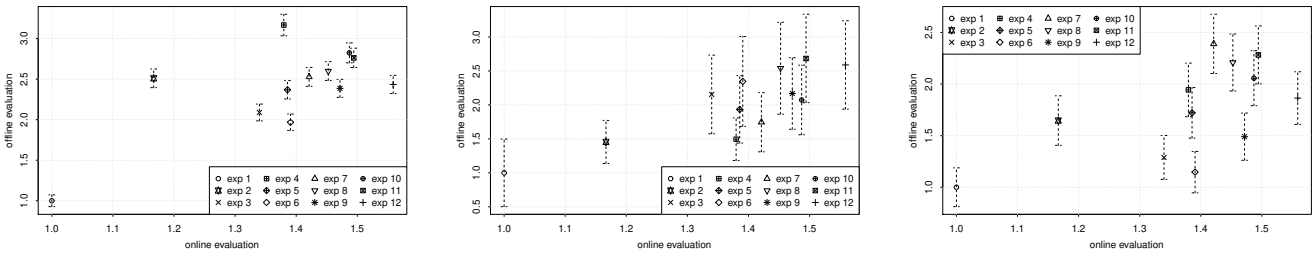
Table 4: Relative ordering of 12 experimental systems evaluated offline by the NCIS estimate ($\lambda = 10^5$) of how many times better the impression-to-stream rate is to the baseline, with statistical significance outcomes.

confidence in results is much lower as well; if we can believe that the 11th experiment could be the best, experiments 10, 3, 9, 6, 8, and 12 are all within the confidence interval of experiment 11 and thus could have a claim to being the best as well. This group contains the actual 5 best experiments, though it also contains one clear miss (experiment 3). Counterintuitively, the lower confidence (represented by wider confidence intervals) gives us *greater* confidence that this offline evaluation is providing results that are usable.

Figures 3a and 3b show Table 3 and 4 (respectively) as scatterplots. The discrepancies in ordering from the former evaluation are easy to see, as are the narrow confidence intervals that make nearly every pair look significantly different. In contrast, the latter evaluation clearly indicates the uncertainty present from evaluating offline, suggesting immediately which experiments are in contention.

It is important to understand the interpretation of the confidence bars in these figures. Note in the right figure that the confidence intervals for experiment 2 and experiment 6 overlap by a small amount, but in Table 4, experiment 2 is statistically significantly worse than experiment 6. This is because non-overlapping confidence intervals is a sufficient but *not necessary* condition for statistical significance. In the offline case, the sample data is the same for both experiments, which means we can use a one-sample (paired) test. This means that the confidence interval used for statistical testing is the confidence interval on the *difference* in means, not the intervals on the means themselves. It is certainly possible that two confidence intervals on means could be overlapping, yet the confidence interval on the difference in means does not contain 0.

4.4.2 Offline evaluation with shuffled data. We also wanted to try to verify the use of fully-shuffled data for offline evaluation. Table 5 shows the relative ordering of experiments by an evaluation over fully-shuffled logs only, with the V_{IS} estimator with no capping and no normalization. The sample size in this case is quite small: only 6,285 sessions that had at least one user impression. As a result the variance is much higher than the online case. Note that the standard errors shown in this table are three orders of magnitude larger than those in Table 1. This translates to fewer experiment pairs being found statistically significantly different: 20 pairs that



(a) V_{CIS} with $\lambda = 100$. Kendall's $\tau = 0.325$. **(b) V_{NCIS} with $\lambda = 100,000$. Kendall's $\tau = 0.636$.** **(c) V_{IS} computed on fully-shuffled logs only. Kendall's $\tau = 0.394$.**

Figure 3: Three plots comparing online and offline evaluations with different estimators.

exp	V_{IS}	std.err×10^2	n	stat. signif. vs experiment
1	1.000	0.165	6,285	· · ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
6	1.147	0.177	6,285	· · · ★ ★ ★ ★ ★ ★ ★ ★ ★
3	1.289	0.187	6,285	★ · · · ★ ★ ★ ★ ★ ★ ★ ★
9	1.490	0.201	6,285	★ ★ · · · · ★ ★ ★ ★ ★ ★
2	1.646	0.211	6,285	★ ★ ★ · · · · · ★ ★ ★ ★
5	1.721	0.215	6,285	★ ★ ★ · · · · · ★ ★ ★ ★
12	1.864	0.224	6,285	★ ★ ★ ★ · · · · · ★ ★ ★
4	1.944	0.228	6,285	★ ★ ★ ★ · · · · · · · ★
10	2.057	0.235	6,285	★ ★ ★ ★ ★ ★ · · · · · ·
8	2.209	0.243	6,285	★ ★ ★ ★ ★ ★ ★ · · · · ·
11	2.283	0.247	6,285	★ ★ ★ ★ ★ ★ ★ · · · · ·
7	2.390	0.253	6,285	★ ★ ★ ★ ★ ★ ★ ★ · · · ·

Table 5: Relative ordering of 12 experimental systems tested offline with shuffled data, measured by how many times better the impression-to-stream rate is to the baseline experiment 1, with statistical significance outcomes.

had been statistically significantly different in the online evaluation are no longer significant (by a one-sample t-test).

The ranking of experiments is also substantially different. The best system and the 7th-best system by the online evaluation swap positions so that the 7th-best is ranked best by the offline evaluation. The Kendall's τ rank correlation between the two is only 0.394[3]. This is not statistically significant, which means that we cannot rule out that the offline evaluation is just producing a random shuffling of experiments.

Figure 3c compares the online and offline evaluation with shuffled logs. It is evident from the plot that the correlation is not very strong, and moreover the error bars do not give us much confidence that more data would improve the result.

4.5 Deciding on systems to test online

In this section we use the simulation-based policy described in Section 3.3 to identify the systems with highest probability of providing good online results. We sample from the multivariate normal distribution fit to offline evaluation results, then transform the resulting simulated means to rank positions.

[3]A τ correlation of 1 means perfect correspondence in the ranking; -1 means the rankings are reversed. A τ of 0 is a random re-ordering.

Figure 4 shows the probability that each system is identified as the "best" over 100,000 simulations of an offline evaluation. The left plot is with no normalization; the right plot uses normalization. Both plots have as their x-axis the $\log_{10}$ of the value used for capping propensities. The assignment of point types to experiments is identical to Figure 3.

Figure 4 clearly shows the importance of normalization. Without it, only experiments 4, 10, and 11 are ever in contention. The actual best experiment—experiment 12—never exceeds 0.01 probability of being identified as best. The right plot, with normalization, shows more variety. With low cap values, experiment 7 is favored, but experiment 11 (the second best overall) quickly rises and remains most likely to be identified as best for cap ≥ 1. As cap increases over 10,000, experiment 12 (the actual best) takes some of the probability mass along with experiments 8 and 6.

Note that Figure 4 is only showing the probability of each experiment being identified as best. We could also look at distributions for second- and third-best. Figure 5 shows plots for the second- and third-best experiments. For second-best, at high cap values we have experiments 6, 8, 11, and 12 in contention. For third-best, experiments 8, 12, 6, 11, 9, 3, and 10 are in contention. Though we do note that there is not a very high probability of the actual third-best experiment ranking third or higher in an offline evaluation.

4.6 System components and characteristics

In Section 4.1 we noted that our 12 experiments used two different feature sets and three different sources of training data. In this section we investigate whether we can reliably identify the best feature set and the best source of training data separately from the experimental systems.

The specific groupings are as follows:

- feature set 1: experiments 1, 2, 4, 11, and 12.
- feature set 2: experiments 3, 5, 6, 7, 8, 9, and 10.
- trained on biased data: experiments 1 and 11.
- trained on shuffled data: experiments 3, 7, 8, and 10.
- trained on de-biased (by NCIS) data: experiments 2, 4, 5, 6, 9, and 12.

To try to predict which feature set or which source of training data is best, we use the same method as above, but collapse predictions by grouping samples by either feature set or training data source.

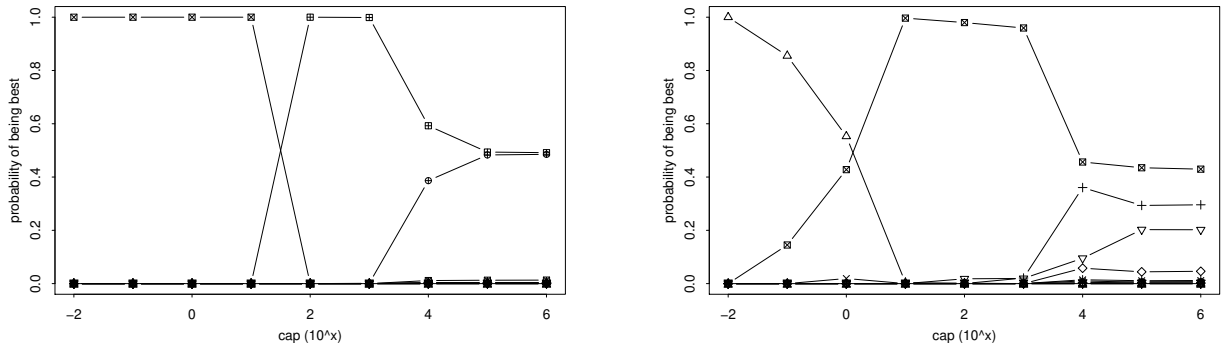


Figure 4: Probability of each experiment being identified as "best" as capping parameter increases (shown on the x-axis as the $\log_{10}$ of the parameter value). The left uses no normalization, the right uses normalization.

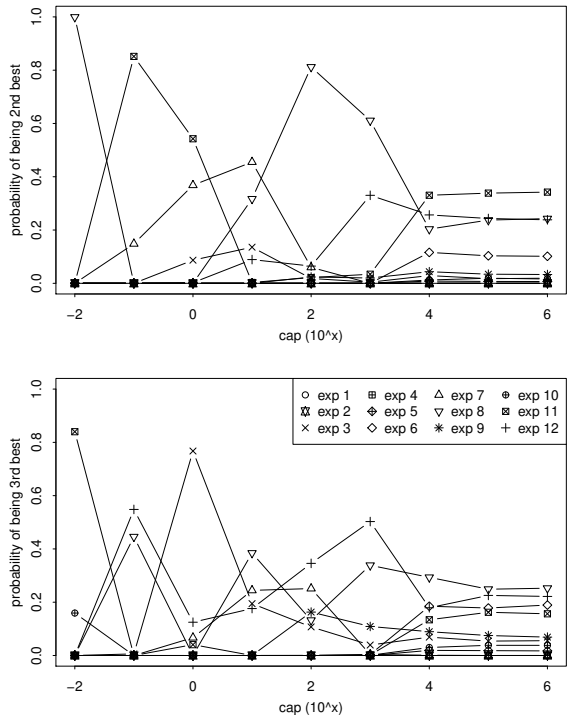


Figure 5: Probability of each experiment being identified as the second-best (top) and third-best (bottom) as capping parameter increases, using normalization.

The result for feature set is that there is a 59% chance that the first feature set is better than the second.It is not surprising that this result is uncertain, given that both the two best and two worst experiments used the first feature set.

For training data source the result is clear. There is a 97% chance that using production log data de-biased by NCIS is the best source of training data, and only a 0.01% chance that the fully-shuffled data is. Of course, this is likely because the amount of fully-shuffled data is very small, as it must be due to the risk of degrading the user experience.

5 CONCLUSION

We have presented a comparison and analysis of an online evaluation via A/B tests and offline evaluation with three different estimators computed from historical log data. Taking the online evaluation as the gold standard, we find best results in an offline evaluation using a normalized and capped estimator based on importance sampling, with a relatively high capping parameter providing the best tradeoff between variance and preserving the relative ordering of experiments. We are thus able to use offline evaluation to predict the results of online evaluation more accurately than previous work.

Our analysis illustrates that problems arising from both bias and variance in offline estimators are mitigated by the practical considerations of identifying the right experiments to A/B test and failing to miss good experiments. Variance may be high for the offline estimator of an individual experiment, but because the practical concern is whether one experiment is better than another and by how much, the variance we are more concerned with is the variance in the difference in the estimator. And because the two experiments are evaluated using the same sample, that variance is typically lower than the variance of either of the experiments separately. Similarly, while capping and normalization may add bias to the estimator, as long as the bias does not affect the relative ordering of experiments it can be acceptable.

It is worth discussing particular aspects of our problem setting that may help make the prediction of online test results easier. For one, the action space we are considering is relatively small—200 cards—compared to other settings. For another, we are assuming independent rewards, which is likely not the case in reality: the expected reward of placing one card after another may depend highly on which card is placed first. This may also explain some of the discrepancy between the online and offline results.

There are a number of directions for future work. Gilotte et al. proposed additional refinements to the NCIS estimator they call *piecewise* and *pointwise* NCIS. We believe these can be refined further and adapted to our problem. There is also potential for exploring rank loss functions and metrics. Finally, we plan to perform this analysis on offline-to-online prediction for other recommendation problems.

REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval: The Concepts and Technology Behind Search.* Addison-Wesley Publishing Company, USA, 2nd edition, 2008.

[2] J. Beel and S. Langer. A comparison of offline evaluations, online evaluations, and user studies in the context of research-paper recommender systems. In *Proceedings of the 19th International Conference on Theory and Practice of Digital Libraries (TPDL)*, volume 9316 of *Lecture Notes in Computer Science*, pages 153–168, 2015.

[3] A. Bellogin, P. Castells, and I. Cantador. Precision-oriented evaluation of recommender systems: An algorithmic comparison. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 333–336, 2011.

[4] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[5] L. Bottou, J. Peters, J. Q. Candela, D. X. Charles, M. Chickering, E. Portugaly, D. Ray, P. Y. Simard, and E. Snelson. Counterfactual reasoning and learning systems - the example of computational advertising. *Journal of Machine Learning Research ()*, 2013.

[6] O. Chapelle, T. Joachims, F. Radlinski, and Y. Yue. Large-scale validation and analysis of interleaved search evaluation. *Trans. Inf. Systems*, 30(1):6:1–41, 2012.

[7] O. Chapelle and Y. Zhang. *A dynamic bayesian network click model for web search ranking*. 2009.

[8] A. Chuklin, I. Markov, and M. de Rijke. Click Models for Web Search. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 7(3):1–115, 2015.

[9] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 659–666, 2008.

[10] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, pages 39–46, 2010.

[11] M. Dudík, D. Erhan, J. Langford, and L. Li. Doubly Robust Policy Evaluation and Optimization. *Statistical Science*, 29(4):485–511, 2014.

[12] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 331–338, 2008.

[13] M. D. Ekstrand, F. M. Harper, M. C. Willemsen, and J. A. Konstan. User perception of differences in recommender algorithms. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 161–168, 2014.

[14] J. Garcia-Gathright, B. St. Thomas, C. Hosey, Z. Nazari, and F. Diaz. Understanding and evaluating user satisfaction with music discovery. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, pages 55–64, 2018.

[15] F. Garcin, B. Faltings, O. Donatsch, A. Alazzawi, C. Bruttin, and A. Huber. Offline and online evaluation of news recommender systems at swissinfo.ch. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 169–176, 2014.

[16] M. Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: Evaluating recommender systems by coverage and serendipity. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, pages 257–260, 2010.

[17] G. G. Gebremeskel and A. P. de Vries. Recommender systems evaluations : Offline, online, time and a/a test. In *CLEF*, 2016.

[18] A. Gilotte, C. Calauzènes, T. Nedelec, A. Abraham, and S. Dollé. Offline a/b testing for recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 198–206, 2018.

[19] A. Gunawardana and G. Shani. *Evaluating Recommender Systems*, pages 265–308. Springer US, Boston, MA, 2015.

[20] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, Dec. 2015.

[21] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, Jan. 2004.

[22] K. Hofmann, L. Li, and F. Radlinski. Online Evaluation for Information Retrieval. *Foundations and Trends® in Information Retrieval*, 10(1):1–117, 2016.

[23] K. Hofmann, S. Whiteson, and M. de Rijke. A probabilistic method for inferring preferences from clicks. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, pages 249–258, 2011.

[24] D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260):663–685, 1952.

[25] N. Hurley and M. Zhang. Novelty and diversity in top-n recommendation – analysis and evaluation. *ACM Trans. Internet Technol.*, 10(4):14:1–14:30, 2011.

[26] D. Jannach and M. Ludewig. Determining characteristics of successful recommendations from log data: a case study. In *Proceedings of the Symposium on Applied Computing - SAC '17*, pages 1643–1648, 2017.

[27] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, Oct. 2002.

[28] T. Joachims, L. a. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in Web search. *ACM Transactions on Information Systems*, 25(2):7–es, 2007.

[29] T. Joachims, A. Swaminathan, and T. Schnabel. Unbiased Learning-to-Rank with Biased Feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining - WSDM '17*, 2017.

[30] M. T. Keane and M. O'Brien. Click Models for Web Search. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 28, 2006.

[31] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: Survey and practical guide. *Data Min. Knowl. Discov.*, 18(1):140–181, 2009.

[32] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 297–306, 2011.

[33] J. McInerney, B. Lacker, S. Hansen, K. Higley, H. Bouchard, A. Gruson, and R. Mehrotra. Explore, exploit, explain: Personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM conference on Recommender Systems (RecSys)*. ACM, 2018.

[34] M. J. D. POWELL and J. SWANN. Weighted uniform sampling ? a monte carlo technique for reducing variance. *IMA Journal of Applied Mathematics*, 2(3):228–236, 1966.

[35] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 43–52, 2008.

[36] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, pages 175–186, 1994.

[37] P. R. ROSENBAUM and D. B. RUBIN. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

[38] M. Rossetti, F. Stella, and M. Zanker. Contrasting offline and online results when evaluating recommendation algorithms. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 31–34, 2016.

[39] A. Said and A. Bellogín. Comparative recommender system evaluation: Benchmarking recommendation frameworks. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 129–136, 2014.

[40] A. Swaminathan and T. Joachims. The self-normalized estimator for counterfactual learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 3231–3239, 2015.

[41] A. Swaminathan, A. Krishnamurthy, A. Agarwal, M. Dudik, J. Langford, D. Jose, and I. Zitouni. Off-policy evaluation for slate recommendation. pages 3635–3645, 2017.

[42] S. Vargas and P. Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 109–116, 2011.

[43] E. M. Voorhees and D. K. Harman. *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*. The MIT Press, 2005.

[44] X. Wang, M. Bendersky, D. Metzler, and M. Najork. Learning to Rank with Selection Bias in Personal Search. pages 115–124, 2016.

[45] Y. Yue, R. Patel, and H. Roehrig. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *Proc. of the 19th International Conference on World Wide Web*, WWW, pages 1011–1018, 2010.