# Off-policy evaluation for slate recommendation

**Adith Swaminathan**
Microsoft Research, Redmond
adswamin@microsoft.com

**Akshay Krishnamurthy**
University of Massachusetts, Amherst
akshay@cs.umass.edu

**Alekh Agarwal**
Microsoft Research, New York
alekha@microsoft.com

**Miroslav Dudík**
Microsoft Research, New York
mdudik@microsoft.com

**John Langford**
Microsoft Research, New York
jcl@microsoft.com

**Damien Jose**
Microsoft, Redmond
dajose@microsoft.com

**Imed Zitouni**
Microsoft, Redmond
izitouni@microsoft.com

## Abstract

This paper studies the evaluation of policies that recommend an ordered set of items (e.g., a ranking) based on some context—a common scenario in web search, ads, and recommendation. We build on techniques from combinatorial bandits to introduce a new practical estimator that uses logged data to estimate a policy's performance. A thorough empirical evaluation on real-world data reveals that our estimator is accurate in a variety of settings, including as a subroutine in a learning-to-rank task, where it achieves competitive performance. We derive conditions under which our estimator is unbiased—these conditions are weaker than prior heuristics for slate evaluation—and experimentally demonstrate a smaller bias than parametric approaches, even when these conditions are violated. Finally, our theory and experiments also show exponential savings in the amount of required data compared with general unbiased estimators.

## 1 Introduction

In recommendation systems for e-commerce, search, or news, we would like to use the data collected during operation to test new content-serving algorithms (called *policies*) along metrics such as revenue and number of clicks [4, 25]. This task is called *off-policy evaluation*. General approaches, namely *inverse propensity scores* (IPS) [13, 18], require unrealistically large amounts of logged data to evaluate whole-page metrics that depend on multiple recommended items, which happens when showing ranked lists. The key challenge is that the number of possible lists (called *slates*) is combinatorially large. As a result, the policy being evaluated is likely to choose different slates from those recorded in the logs most of the time, unless it is very similar to the data-collection policy. This challenge is fundamental [34], so any off-policy evaluation method that works with large slates needs to make some structural assumptions about the whole-page metric or the user behavior.

Previous work on off-policy evaluation and whole-page optimization improves the probability of match between logging and evaluation by restricting attention to small slate spaces [35, 26], introducing assumptions that allow for partial matches between the proposed and observed slates [27], or assuming that the policies used for logging and evaluation are similar [4, 32]. Another line of work constructs parametric models of slate quality [8, 16, 14] (see also Sec. 4.3 of [17]). While these approaches require less data, they can have large bias, and their use in practice requires an expensive trial-and-error cycle involving weeks-long A/B tests to develop new policies [20]. In this paper we

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

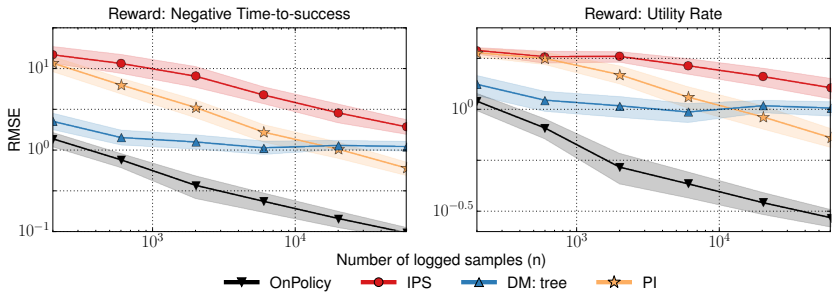


Figure 1: Off-policy evaluation of two whole-page user-satisfaction metrics on proprietary search engine data. Average RMSE of different estimators over 50 runs on a log-log scale. Our method (PI) achieves the best performance with moderate data sizes. The unbiased IPS method suffers high variance, and direct modeling (DM) of the metrics suffers high bias. ONPOLICY is the expensive choice of deploying the policy, for instance, in an A/B test.

design a method more robust to problems with bias and with only modest data requirements, with the goal of substantially shortening this cycle and accelerating the policy development process.

We frame the slate recommendation problem as a combinatorial generalization of *contextual bandits* [3, 23, 13]. In combinatorial contextual bandits, for each *context*, a policy selects a *slate* consisting of component *actions*, after which a *reward* for the entire slate is observed. In web search, the context is the search query augmented with a user profile, the slate is the search results page consisting of a list of retrieved documents (actions), and example reward metrics are page-level measures such as time-to-success, NDCG (position-weighted relevance), or other measures of user satisfaction. As input we receive contextual bandit data obtained by some *logging policy*, and our goal is to estimate the reward of a new *target policy*. This off-policy setup differs from online learning in contextual bandits, where the goal is to adaptively maximize the reward in the presence of an explore-exploit trade-off [5].

Inspired by work in *combinatorial* and *linear bandits* [7, 31, 11], we propose an estimator that makes only a weak assumption about the evaluated metric, while exponentially reducing the data requirements in comparison with IPS. Specifically, we posit a *linearity assumption*, stating that the slate-level reward (e.g., time to success in web search) decomposes additively across actions, but the action-level rewards are not observed. Crucially, the action-level rewards are allowed to depend on the context, and we do not require that they be easily modeled from the features describing the context. In fact, our method is completely agnostic to the representation of contexts.

We make the following contributions:

1. The *pseudoinverse estimator* (PI) for off-policy evaluation: a general-purpose estimator from the combinatorial bandit literature, adapted for off-policy evaluation. When ranking $\ell$ out of $m$ items under the linearity assumption, PI typically requires $\mathcal{O}(\ell m/\varepsilon^2)$ samples to achieve error at most $\varepsilon$—an exponential gain over the $m^{\Omega(\ell)}$ sample complexity of IPS. We provide distribution-dependent bounds based on the overlap between logging and target policies.
2. Experiments on real-world search ranking datasets: The strong performance of the PI estimator provides, to our knowledge, the first demonstration of high-quality off-policy evaluation of whole-page metrics, comprehensively outperforming prior baselines (see Fig. 1).
3. Off-policy optimization: We provide a simple procedure for learning to rank (L2R) using the PI estimator to impute action-level rewards for each context. This allows direct optimization of whole-page metrics via pointwise L2R approaches, *without requiring pointwise feedback*.

**Related work** Large state spaces have typically been studied in the online, or on-policy, setting. Some works assume specific parametric (e.g., linear) models relating the metrics to the features describing a slate [2, 31, 15, 10, 29]; this can lead to bias if the model is inaccurate (e.g., we might not have access to sufficiently predictive features). Others posit the same linearity assumption as we do, but further assume a *semi-bandit* feedback model where the rewards of all actions on the slate

are revealed [19, 22, 21]. While much of the research focuses on on-policy setting, the off-policy paradigm studied in this paper is often preferred in practice since it might not be possible to implement low-latency updates needed for online learning, or we might be interested in many different metrics and require a manual review of their trade-offs before deploying new policies.

At a technical level, the PI estimator has been used in online learning [7, 31, 11], but the analysis there is tailored to the specific data collection policies used by the learner. In contrast, we provide distribution-dependent bounds without any assumptions on the logging or target policy.

## 2 Setting and notation

In combinatorial contextual bandits, a decision maker repeatedly interacts as follows:

1. the decision maker observes a *context* $x$ drawn from a distribution $D(x)$ over some space $X$;
2. based on the context, the decision maker chooses a *slate* $\mathbf{s} = (s_1, \dots, s_\ell)$ consisting of *actions* $s_j$, where a position $j$ is called a *slot*, the number of slots is $\ell$, actions at position $j$ come from some space $A_j(x)$, and the slate $\mathbf{s}$ is chosen from a set of allowed slates $S(x) \subseteq A_1(x) \times \cdots \times A_\ell(x)$;
3. given the context and slate, a reward $r \in [-1, 1]$ is drawn from a distribution $D(r \mid x, \mathbf{s})$; rewards in different rounds are independent, conditioned on contexts and slates.

The context space $X$ can be infinite, but the set of actions is finite. We assume $|A_j(x)| = m_j$ for all contexts $x \in X$ and define $m \coloneqq \max_j m_j$ as the maximum number of actions per slot. The goal of the decision maker is to *maximize the reward*. The decision maker is modeled as a *stochastic policy* $\pi$ that specifies a conditional distribution $\pi(\mathbf{s} \mid x)$ (a deterministic policy is a special case). The *value* of a policy $\pi$, denoted $V(\pi)$, is defined as the expected reward when following $\pi$:

$$V(\pi) \coloneqq \mathbb{E}_{x\sim D}\mathbb{E}_{\mathbf{s}\sim\pi(\cdot|x)}\mathbb{E}_{r\sim D(\cdot|x,\mathbf{s})}\big[r\big] \ . \tag{1}$$

To simplify derivations, we extend the conditional distribution $\pi$ into a distribution over triples $(x, \mathbf{s}, r)$ as $\pi(x, \mathbf{s}, r) \coloneqq D(r \mid x, \mathbf{s})\pi(\mathbf{s} \mid x)D(x)$. With this shorthand, we have $V(\pi) = \mathbb{E}_\pi[r]$.

To finish this section, we introduce notation for the expected reward for a given context and slate, which we call the *slate value*, and denote as:

$$V(x, \mathbf{s}) \coloneqq \mathbb{E}_{r\sim D(\cdot|x,\mathbf{s})}[r] \ . \tag{2}$$

**Example 1** (Cartesian product)**.** Consider the optimization of a news portal where the reward is the whole-page advertising revenue. Context $x$ is the user profile, slate is the news-portal page with slots corresponding to news sections,[1] and actions are the articles. The set of valid slates is the Cartesian product $S(x) = \prod_{j\le\ell} A_j(x)$. The number of valid slates is exponential in $\ell$: $|S(x)| = \prod_{j\le\ell} m_j$.

**Example 2** (Ranking)**.** Consider web search and ranking. Context $x$ is the query along with user profile. Actions correspond to search items (such as webpages). The policy chooses $\ell$ of $m$ items, where the set $A(x)$ of $m$ items for a context $x$ is chosen from a corpus by a filtering step (e.g., a database query). We have $A_j(x) = A(x)$ for all $j \le \ell$, but the allowed slates $S(x)$ have no repetitions. The number of valid slates is exponential in $\ell$: $|S(x)| = m!/(m-\ell)! = m^{\Omega(\ell)}$. A reward could be the *negative time-to-success*, i.e., negative of the time taken by the user to find a relevant item.

### 2.1 Off-policy evaluation and optimization

In the *off-policy* setting, we have access to the *logged data* $(x_1, \mathbf{s}_1, r_1), \dots, (x_n, \mathbf{s}_n, r_n)$ collected using a past policy $\mu$, called the *logging policy*. *Off-policy evaluation* is the task of estimating the value of a new policy $\pi$, called the *target policy*, using the logged data. *Off-policy optimization* is the harder task of finding a policy $\hat\pi$ that achieves maximal reward.

There are two standard approaches for off-policy evaluation. The *direct method* (DM) uses the logged data to train a (parametric) model $\hat r(x, \mathbf{s})$ for predicting the expected reward for a given context and slate. $V(\pi)$ is then estimated as

$$\hat V_{\text{DM}}(\pi) = \tfrac{1}{n}\textstyle\sum_{i=1}^n \sum_{\mathbf{s}\in S(x)} \hat r(x_i, \mathbf{s})\pi(\mathbf{s} \mid x_i) \ . \tag{3}$$

[1] For simplicity, we do not discuss the more general setting of showing multiple articles in each news section.

The direct method is often biased due to mismatch between model assumptions and ground truth.

The second approach, which is provably unbiased (under modest assumptions), is the *inverse propensity score* (IPS) estimator [18]. The IPS estimator re-weights the logged data according to ratios of slate probabilities under the target and logging policy. It has two common variants:

$$\hat{V}_{\text{IPS}}(\pi) = \frac{1}{n}\sum_{i=1}^{n} r_i \cdot \frac{\pi(\mathbf{s}_i|x_i)}{\mu(\mathbf{s}_i|x_i)} \quad , \quad \hat{V}_{\text{wIPS}}(\pi) = \sum_{i=1}^{n} r_i \cdot \frac{\pi(\mathbf{s}_i|x_i)}{\mu(\mathbf{s}_i|x_i)} \Big/ \Big(\sum_{i=1}^{n} \frac{\pi(\mathbf{s}_i|x_i)}{\mu(\mathbf{s}_i|x_i)}\Big) \quad . \tag{4}$$

wIPS generally has better variance with an asymptotically zero bias. The variance of both estimators grows linearly with $\frac{\pi(\mathbf{s}|x)}{\mu(\mathbf{s}|x)}$, which can be $\Omega(|S(x)|)$. This is prohibitive when $|S(x)| = m^{\Omega(\ell)}$.

## 3 Our approach

The IPS estimator is minimax optimal [34], so its exponential variance is unavoidable in the worst case. We circumvent this hardness by positing an assumption on the structure of rewards. Specifically, we assume that the slate-level reward is a sum of unobserved action-level rewards that depend on the context, the action, and the position on the slate, but not on the other actions on the slate.

Formally, we consider *slate indicator vectors* in $\mathbb{R}^{\ell m}$ whose components are indexed by pairs $(j, a)$ of slots and possible actions in them. A slate is described by an indicator vector $\mathbf{1}_{\mathbf{s}} \in \mathbb{R}^{\ell m}$ whose entry at position $(j, a)$ is equal to 1 if the slate $\mathbf{s}$ has action $a$ in slot $j$, i.e., if $s_j = a$. The above assumption is formalized as follows:

**Assumption 1** (Linearity Assumption)**.** For each context $x \in X$ there exists an (unknown) *intrinsic reward* vector $\boldsymbol{\phi}_x \in \mathbb{R}^{\ell m}$ such that the slate value satisfies $V(x, \mathbf{s}) = \mathbf{1}_{\mathbf{s}}^T \boldsymbol{\phi}_x = \sum_{j=1}^{\ell} \phi_x(j, s_j)$.

The slate indicator vector can be viewed as a feature vector, representing the slate, and $\boldsymbol{\phi}_x$ can be viewed as a *context-specific* weight vector. The assumption refers to the fact that the value of a slate is a linear function of its feature representation. However, note that this linear dependence is allowed to be completely different across contexts, because we make no assumptions on how $\boldsymbol{\phi}_x$ depends on $x$, and in fact our method does not even attempt to accurately estimate $\boldsymbol{\phi}_x$. Being agnostic to the form of $\boldsymbol{\phi}_x$ is the key departure from the direct method and parametric bandits.

While Assumption 1 rules out interactions among different actions on a slate,[2] its ability to vary intrinsic rewards arbitrarily across contexts captures many common metrics in information retrieval, such as the *normalized discounted cumulative gain* (NDCG) [6], a common metric in web ranking:

**Example 3** (NDCG)**.** For a slate $\mathbf{s}$, we first define $\text{DCG}(x, \mathbf{s}) \coloneqq \sum_{j=1}^{\ell} \frac{2^{rel(x,s_j)}-1}{\log_2(j+1)}$ where $rel(x, a)$ is the relevance of document $a$ on query $x$. Then $\text{NDCG}(x, \mathbf{s}) \coloneqq \text{DCG}(x, \mathbf{s})/\text{DCG}^\star(x)$ where $\text{DCG}^\star(x) = \max_{\mathbf{s}\in S(x)} \text{DCG}(x, \mathbf{s})$, so NDCG takes values in $[0, 1]$. Thus, NDCG satisfies Assumption 1 with $\phi_x(j, a) = \left(2^{rel(x,a)} - 1\right) \Big/ \log_2(j + 1)\text{DCG}^\star(x)$.

In addition to Assumption 1, we also make the standard assumption that the logging policy puts non-zero probability on all slates that can be potentially chosen by the target policy. This assumption is also required for IPS, otherwise unbiased off-policy evaluation is impossible [24].

**Assumption 2** (Absolute Continuity)**.** The off-policy evaluation problem satisfies the *absolute continuity* assumption if $\mu(\mathbf{s} \mid x) > 0$ whenever $\pi(\mathbf{s} \mid x) > 0$ with probability one over $x \sim D$.

### 3.1 The pseudoinverse estimator

Using Assumption 1, we can now apply the techniques from the combinatorial bandit literature to our problem. In particular, our estimator closely follows the recipe of Cesa-Bianchi and Lugosi [7], albeit with some differences to account for the off-policy and contextual nature of our setup. Under Assumption 1, we can view the recovery of $\boldsymbol{\phi}_x$ for a given context $x$ as a linear regression problem. The covariates $\mathbf{1}_{\mathbf{s}}$ are drawn according to $\mu(\cdot \mid x)$, and the reward follows a linear model, conditional on $\mathbf{s}$ and $x$, with $\boldsymbol{\phi}_x$ as the "weight vector". Thus, we can write the MSE of an estimate $\mathbf{w}$ as $\mathbb{E}_{\mathbf{s}\sim\mu(\cdot|x)}\mathbb{E}_{r\sim D(\cdot|\mathbf{s},x)}[(\mathbf{1}_{\mathbf{s}}^T\mathbf{w} - r)^2]$, or more compactly as $\mathbb{E}_{\mu}[(\mathbf{1}_{\mathbf{s}}^T\mathbf{w} - r)^2 \mid x]$, using our definition of $\mu$ as a distribution over triples $(x, \mathbf{s}, r)$. We estimate $\boldsymbol{\phi}_x$ by the MSE minimizer with the smallest

[2] We discuss limitations of Assumption 1 and directions to overcome them in Sec. 5.

norm, which can be written in closed form as

$$\bar{\boldsymbol{\phi}}_x = \left(\mathbb{E}_\mu[\mathbf{1}_\mathbf{s}\mathbf{1}_\mathbf{s}^T \mid x]\right)^\dagger \mathbb{E}_\mu[r\mathbf{1}_\mathbf{s} \mid x] \ , \tag{5}$$

where $\mathbf{M}^\dagger$ is the Moore-Penrose pseudoinverse of a matrix $\mathbf{M}$. Note that this idealized "estimator" $\bar{\boldsymbol{\phi}}_x$ uses conditional expectations over $\mathbf{s} \sim \mu(\cdot \mid x)$ and $r \sim D(\cdot \mid \mathbf{s}, x)$. To simplify the notation, we write $\boldsymbol{\Gamma}_{\mu,x} \coloneqq \mathbb{E}_\mu[\mathbf{1}_\mathbf{s}\mathbf{1}_\mathbf{s}^T \mid x] \in \mathbb{R}^{\ell m \times \ell m}$ to denote the (uncentered) covariance matrix for our regression problem, appearing on the right-hand side of Eq. (5). We also introduce notation for the second term in Eq. (5) and its empirical estimate: $\boldsymbol{\theta}_{\mu,x} \coloneqq \mathbb{E}_\mu[r\mathbf{1}_\mathbf{s} \mid x]$, and $\hat{\boldsymbol{\theta}}_i \coloneqq r_i \mathbf{1}_{\mathbf{s}_i}$.

Thus, our regression estimator (5) is simply $\bar{\boldsymbol{\phi}}_x = \boldsymbol{\Gamma}_{\mu,x}^\dagger \boldsymbol{\theta}_{\mu,x}$. Under Assumptions 1 and 2, it is easy to show that $V(x, \mathbf{s}) = \mathbf{1}_\mathbf{s}^T \bar{\boldsymbol{\phi}}_x = \mathbf{1}_\mathbf{s}^T \boldsymbol{\Gamma}_{\mu,x}^\dagger \boldsymbol{\theta}_{\mu,x}$. Replacing $\boldsymbol{\theta}_{\mu,x}$ with $\hat{\boldsymbol{\theta}}_i$ motivates the following estimator for $V(\pi)$, which we call the *pseudoinverse estimator* or PI:

$$\hat{V}_{\text{PI}}(\pi) \ = \ \frac{1}{n}\sum_{i=1}^n \sum_{\mathbf{s} \in S} \pi(\mathbf{s} \mid x_i) \mathbf{1}_\mathbf{s}^T \boldsymbol{\Gamma}_{\mu,x_i}^\dagger \hat{\boldsymbol{\theta}}_i \ = \ \frac{1}{n}\sum_{i=1}^n r_i \cdot \mathbf{q}_{\pi,x_i}^T \boldsymbol{\Gamma}_{\mu,x_i}^\dagger \mathbf{1}_{\mathbf{s}_i} \ . \tag{6}$$

In Eq. (6) we have expanded the definition of $\hat{\boldsymbol{\theta}}_i$ and introduced the notation $\mathbf{q}_{\pi,x}$ for the expected slate indicator under $\pi$ conditional on $x$, $\mathbf{q}_{\pi,x} \coloneqq \mathbb{E}_\pi[\mathbf{1}_\mathbf{s} \mid x]$. The summation over $\mathbf{s}$ required to obtain $\mathbf{q}_{\pi,x_i}$ in Eq. (6) can be replaced by a small sample. We can also derive a weighted variant of PI:

$$\hat{V}_{\text{wPI}}(\pi) = \frac{\sum_{i=1}^n r_i \cdot \mathbf{q}_{\pi,x_i}^T \boldsymbol{\Gamma}_{\mu,x_i}^\dagger \mathbf{1}_{\mathbf{s}_i}}{\sum_{i=1}^n \mathbf{q}_{\pi,x_i}^T \boldsymbol{\Gamma}_{\mu,x_i}^\dagger \mathbf{1}_{\mathbf{s}_i}} \ . \tag{7}$$

We prove the following unbiasedness property in Appendix A.

**Proposition 1.** *If Assumptions 1 and 2 hold, then the estimator $\hat{V}_{\text{PI}}$ is unbiased, i.e., $\mathbb{E}_{\mu^n}[\hat{V}_{\text{PI}}] = V(\pi)$, where the expectation is over the $n$ logged examples sampled i.i.d. from $\mu$.*

As special cases, PI reduces to IPS when $\ell = 1$, and simplifies to $\sum_{i=1}^n r_i/n$ when $\pi = \mu$ (see Appendix C). To build further intuition, we consider the settings of Examples 1 and 2, and simplify the PI estimator to highlight the improvement over IPS.

**Example 4** (PI for a Cartesian product when $\mu$ is a product distribution)**.** The PI estimator for the Cartesian product slate space, when $\mu$ factorizes across slots as $\mu(\mathbf{s} \mid x) = \prod_j \mu(s_j \mid x)$, simplifies to

$$\hat{V}_{\text{PI}}(\pi) = \tfrac{1}{n}\textstyle\sum_{i=1}^n r_i \cdot \left(\sum_{j=1}^\ell \frac{\pi(s_{ij}|x_i)}{\mu(s_{ij}|x_i)} - \ell + 1\right) \ ,$$

by Prop. 2 in Appendix D. Note that unlike IPS, which divides by probabilities of whole slates, the PI estimator only divides by probabilities of actions appearing in individual slots. Thus, the magnitude of each term of the outer summation is only $\mathcal{O}(\ell m)$, whereas the IPS terms are $m^{\Omega(\ell)}$.

**Example 5** (PI for rankings with $\ell = m$ and uniform logging)**.** In this case,

$$\hat{V}_{\text{PI}}(\pi) = \tfrac{1}{n}\textstyle\sum_{i=1}^n r_i \cdot \left(\sum_{j=1}^\ell \frac{\pi(s_{ij}|x_i)}{1/(m-1)} - m + 2\right) \ ,$$

by Prop. 4 in Appendix E.1. The summands are again $\mathcal{O}(\ell m) = \mathcal{O}(m^2)$.

### 3.2 Deviation analysis

So far, we have shown that PI is unbiased under our assumptions and overcomes the deficiencies of IPS in specific examples. We now derive a finite-sample error bound, based on the overlap between $\pi$ and $\mu$. We use Bernstein's inequality, for which we define the variance and range terms:

$$\sigma^2 \coloneqq \mathbb{E}_{x \sim D}\left[\mathbf{q}_{\pi,x}^T \boldsymbol{\Gamma}_{\mu,x}^\dagger \mathbf{q}_{\pi,x}\right] \ , \quad \rho \coloneqq \sup_x \sup_{\mathbf{s}:\mu(\mathbf{s}|x)>0} \left|\mathbf{q}_{\pi,x}^T \boldsymbol{\Gamma}_{\mu,x}^\dagger \mathbf{1}_\mathbf{s}\right| \ . \tag{8}$$

The quantity $\sigma^2$ bounds the variance whereas $\rho$ bounds the range. They capture the "average" and "worst-case" mismatch between $\mu$ and $\pi$. They equal one when $\pi = \mu$ (see Appendix C), and yield the following deviation bound:

**Theorem 1.** *Under Assumptions 1 and 2, let $\sigma^2$ and $\rho$ be defined as in Eq.* (8)*. Then, for any $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$\left|\hat{V}_{\mathrm{PI}}(\pi) - V(\pi)\right| \leq \sqrt{\frac{2\sigma^2 \ln(2/\delta)}{n}} + \frac{2(\rho+1)\ln(2/\delta)}{3n} \ .$$

We observe that this finite sample bound is structurally different from the regret bounds studied in the prior works on combinatorial bandits. The bound incorporates the extent of overlap between $\pi$ and $\mu$ so that we have higher confidence in our estimates when the logging and evaluation policies are similar—an important consideration in off-policy evaluation.

While the bound might look complicated, it simplifies if we consider the class of $\varepsilon$-uniform logging policies. Formally, for any policy $\mu$, define $\mu_\varepsilon(\mathbf{s} \mid x) = (1-\varepsilon)\mu(\mathbf{s} \mid x) + \varepsilon\nu(\mathbf{s} \mid x)$, with $\nu$ being the uniform distribution over the set $S(x)$. For suitably small $\varepsilon$, such logging policies are widely used in practice. We have the following corollary for these policies, proved in Appendix E:

**Corollary 1.** *In the settings of Example 1 or Example 2, if the logging is done with $\mu_\varepsilon$ for some $\varepsilon > 0$, we have $|\hat{V}_{\mathrm{PI}}(\pi) - V(\pi)| \leq \mathcal{O}\big(\sqrt{\varepsilon^{-1}\ell m/n}\big)$.*

Again, this turns the $\Omega(m^\ell)$ data dependence of IPS into $O(m\ell)$. The key step in the proof is the bound on a certain norm of $\mathbf{\Gamma}_\nu^\dagger$, similar to the bounds of Cesa-Bianchi and Lugosi [7], but our results are a bit sharper.

# 4 Experiments

We empirically evaluate the performance of the pseudoinverse estimator for ranking problems. We first show that PI outperforms prior works in a comprehensive semi-synthetic study using a public dataset. We then use our estimator for *off-policy optimization*, i.e., to learn ranking policies, competitively with supervised learning that uses more information. Finally, we demonstrate substantial improvements on proprietary data from search engine logs for two user-satisfaction metrics used in practice: *time-to-success* and *utility rate*, which do not satisfy the linearity assumption. More detailed results are deferred to Appendices F and G. All of our code is available online.[3]

## 4.1 Semi-synthetic evaluation

Our semi-synthetic evaluation uses labeled data from the Microsoft Learning to Rank Challenge dataset [30] (MSLR-WEB30K) to create a contextual bandit instance. Queries form the contexts $x$ and actions $a$ are the available documents. The dataset contains over 31K queries, each with up to 1251 judged documents, where the query-document pairs are judged on a 5-point scale, $\mathit{rel}(x,a) \in \{0,\ldots,4\}$. Each pair $(x,a)$ has a feature vector $\mathbf{f}(x,a)$, which can be partitioned into title and body features ($\mathbf{f}_{\text{title}}$ and $\mathbf{f}_{\text{body}}$). We consider two slate rewards: NDCG from Example 3, and the *expected reciprocal rank*, ERR [9], which *does not* satisfy linearity, and is defined as

$$\mathrm{ERR}(x,\mathbf{s}) \coloneqq \textstyle\sum_{r=1}^{\ell} \frac{1}{r} \prod_{i=1}^{r-1}(1-R(s_i))R(s_r) \ , \quad \text{where } R(a) = \frac{2^{\mathit{rel}(x,a)}-1}{2^{\mathit{maxrel}}} \text{ with } \mathit{maxrel} = 4.$$

To derive several distinct logging and target policies, we first train two lasso regression models, called $\mathit{lasso}_{\text{title}}$ and $\mathit{lasso}_{\text{body}}$, and two regression tree models, called $\mathit{tree}_{\text{title}}$ and $\mathit{tree}_{\text{body}}$, to predict relevances from $\mathbf{f}_{\text{title}}$ and $\mathbf{f}_{\text{body}}$, respectively. To create the logs, queries $x$ are sampled uniformly, and the set $A(x)$ consists of the top $m$ documents according to $\mathit{tree}_{\text{title}}$. The logging policy is parametrized by a model, either $\mathit{tree}_{\text{title}}$ or $\mathit{lasso}_{\text{title}}$, and a scalar $\alpha \geq 0$. It samples from a multinomial distribution over documents $p_\alpha(a|x) \propto 2^{-\alpha\lfloor \log_2 \mathit{rank}(x,a)\rfloor}$ where $\mathit{rank}(x,a)$ is the rank of document $a$ for query $x$ according to the corresponding model. Slates are constructed slot-by-slot, sampling *without replacement* according to $p_\alpha$. Varying $\alpha$ interpolates between uniformly random and deterministic logging. Thus, all logging policies are based on the models derived from $\mathbf{f}_{\text{title}}$. We consider two deterministic target policies based on the two models derived from $\mathbf{f}_{\text{body}}$, i.e., $\mathit{tree}_{\text{body}}$ and $\mathit{lasso}_{\text{body}}$, which select the top $\ell$ documents according to the corresponding model. The four base models are fairly distinct: on average fewer than 2.75 documents overlap among top 10 (see Appendix H).

[3] https://github.com/adith387/slates_semisynth_expts

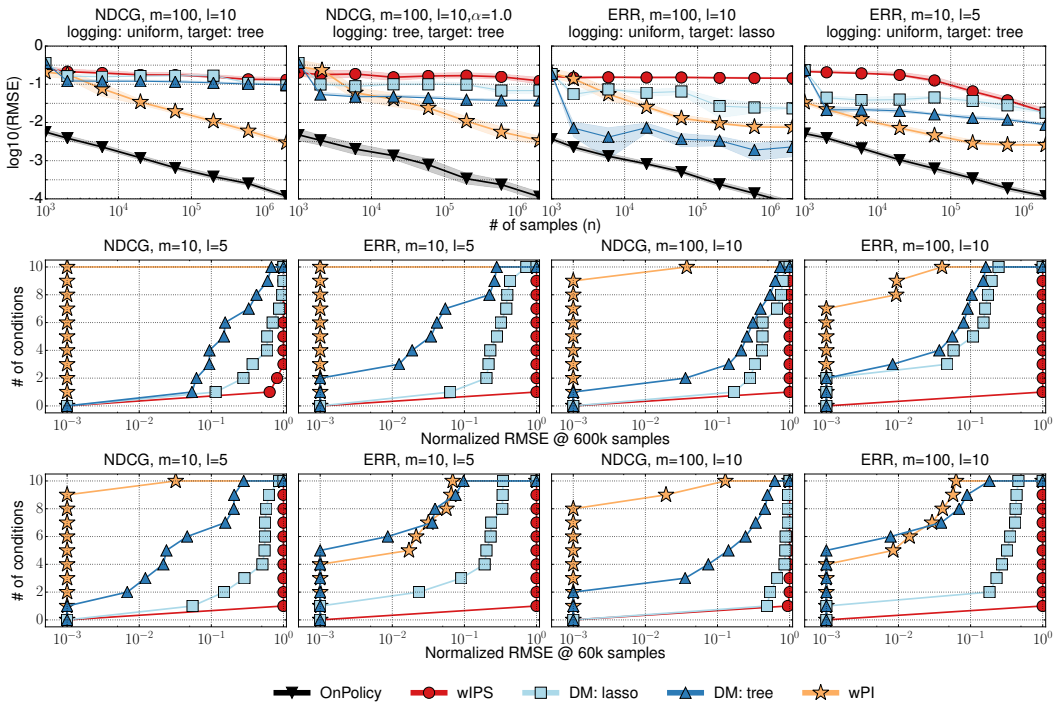


Figure 2: *Top:* RMSE of various estimators under four experimental conditions (see Appendix F for all 40 conditions). *Middle:* CDF of normalized RMSE at 600k samples; each plot aggregates over 10 logging-target combinations; closer to top-left is better. *Bottom:* Same as middle but at 60k samples.

We compare the weighted estimator wPI with the direct method (DM) and weighted IPS (wIPS). (Weighted variants outperformed the unweighted ones.) We implement two variants of DM: regression trees and lasso, each trained on the first $n/2$ examples and using the remaining $n/2$ examples for evaluation according to Eq. (3). We also include an aspirational baseline, ONPOLICY, which corresponds to deploying the target policy as in an A/B test and returning the average of observed rewards. This is the expensive alternative we wish to avoid.

We evaluate the estimators by recording the root mean square error (RMSE) as a function of the number of samples, averaged over at least 25 independent runs. We do this for 40 different experimental conditions, considering two reward metrics, two slate-space sizes, and 10 combinations of target and logging policies (including the choice of $\alpha$). The top row of Fig. 2 shows results for four representative conditions (see Appendix F for all results), while the middle and bottom rows aggregate across conditions. To produce the aggregates, we shift and rescale the RMSE of all methods, at 600k (middle row) or 60k (bottom row) samples, so the best performance is at 0.001 and the worst is at 1.0 (excluding ONPOLICY). (We use 0.001 instead of 0.0 to allow plotting on a log scale.) The aggregate plots display the cumulative distribution function of these normalized RMSE values across 10 target-logging combinations, keeping the metric and the slate-space size fixed.

The pseudoinverse estimator wPI easily dominates wIPS across all experimental conditions, as can be seen in Fig. 2 (top) and in Appendix F. While wIPS and IPS are (asymptotically) unbiased even without linearity assumption, they both suffer from a large variance caused by the slate size. The variance and hence the mean square error of wIPS and IPS grows exponentially with the slate size, so they perform poorly beyond the smallest slate sizes. DM performs well in some cases, especially with few samples, but often plateaus or degrades eventually as it overfits on the logging distribution, which is different from the target. While wPI does not always outperform DM methods (e.g., Fig. 2, top row, second from right), it is the only method that works robustly across all conditions, as can be seen in the aggregate plots. In general, choosing between DM and wPI is largely a matter of bias-variance tradeoff. DM can be particularly good with very small data sizes, because of its low variance, and in those settings it is often the best choice. However, PI performs comprehensively better given enough data (see Fig. 2, middle row).

In the top row of Fig. 2, we see that, as expected, wPI is biased for the ERR metric since ERR does not satisfy linearity. The right two panels also demonstrate the effect of varying $m$ and $\ell$. While wPI deteriorates somewhat for the larger slate space, it still gives a meaningful estimate. In contrast, wIPS fails to produce any meaningful estimate in the larger slate space and its RMSE barely improves with more data. Finally, the left two plots in the top row show that wPI is fairly insensitive to the amount of stochasticity in logging, whereas DM improves with more overlap between logging and target.

### 4.2 Semi-synthetic policy optimization

We now show how to use the pseudoinverse estimator for off-policy optimization. We leverage pointwise learning to rank (L2R) algorithms, which learn a scoring function for query-document pairs by fitting to relevance labels. We call this the *supervised* approach, as it requires relevance labels.

Instead of requiring relevance labels, we use the pseudoinverse estimator to convert page-level reward into per-slot reward components—the estimates of $\phi_x(j, a)$—and these become targets for regression. Thus, the pseudoinverse estimator enables pointwise L2R to optimize whole-page metrics even without relevance labels. Given a contextual bandit dataset $\{(x_i, \mathbf{s}_i, r_i)\}_{i \le n}$ collected by the logging policy $\mu$, we begin by creating the estimates of $\boldsymbol{\phi}_{x_i}$: $\hat{\boldsymbol{\phi}}_i = \boldsymbol{\Gamma}^{\dagger}_{\mu, x_i} \hat{\boldsymbol{\theta}}_i$, turning the $i$-th contextual bandit example into $\ell m$ regression examples. The trained regression model is used to create a slate, starting with the highest scoring slot-action pair, and continuing greedily (excluding the pairs with the already chosen slots or actions). This procedure is detailed in Appendix G. Note that without the linearity assumptions, our imputed regression targets might not lead to the best possible learned policy, but we still expect to adapt somewhat to the slate-level metric.

We use the MSLR-WEB10K dataset [30] to compare our approach with benchmarked results [33] for NDCG@3 (i.e., $\ell = 3$).[4] This dataset contains 10k queries, over 1.2M relevance judgments, and up to 908 judged documents per query. The state-of-the-art *listwise* L2R method on this dataset is a highly tuned variant of LambdaMART [1] (with an ensemble of 1000 trees, each with up to 70 leaves).

We use the provided 5-fold split and always train on bandit data collected by uniform logging from four folds, while evaluating with supervised data on the fifth. We compare our approach, titled PI-OPT, against the supervised approach (SUP), trained to predict the *gains*, equal to $2^{rel(x,a)} - 1$, computed using annotated relevance judgements in the training folds (predicting raw relevances was inferior). Both PI-OPT and SUP train gradient boosted regression trees (with 1000 trees, each with up to 70 leaves). Additionally, we also experimented with the ERR metric.

The average test-set performance (computed using ground-truth relevance judgments for each test set) across the 5-folds is reported in Table 1. Our method, PI-OPT is competitive with the supervised baseline SUP for NDCG, and is substantially superior for ERR. A different transformation instead of gains might yield a stronger supervised baseline for ERR, but this only illustrates the key benefit of PI-OPT: *the right pointwise targets are automatically inferred for any whole-page metric.* Both PI-OPT and SUP are slightly worse than LambdaMART for NDCG@3, but they are arguably not as highly tuned, and PI-OPT only uses the slate-level metric.

Table 1: Comparison of L2R approaches optimizing NDCG@3 and ERR@3. LambdaMART is a tuned list-wise approach. SUP and PI-OPT use the same pointwise L2R learner; SUP uses $8 \times 10^5$ relevance judgments, PI-OPT uses $10^7$ samples (under uniform logging) with page-level rewards.

| Metric | LambdaMART | uniformly random | SUP | PI-OPT |
|---|---|---|---|---|
| NDCG@3 | 0.457 | 0.152 | 0.438 | 0.421 |
| ERR@3 | — | 0.096 | 0.311 | 0.321 |

### 4.3 Real-world experiments

We finally evaluate all methods using logs collected from a popular search engine. The dataset consists of search queries, for which the logging policy randomly (non-uniformly) chooses a slate of

[4] Our dataset here differs from the dataset MSLR-WEB30K used in Sec. 4.1. There our goal was to study realistic problem dimensions, e.g., constructing length-10 rankings out of 100 candidates. Here, we use MSLR-WEB10K, because it is the largest dataset with public benchmark numbers by state-of-the-art approaches (specifically LambdaMART).

size $\ell = 5$ from a small pre-filtered set of documents of size $m \leq 8$. After preprocessing, there are 77 unique queries and 22K total examples, meaning that for each query, we have logged impressions for many of the available slates. As before, we create the logs by sampling queries uniformly at random, and using a logging policy that samples uniformly from the slates shown for this query.

We consider two page-level metrics: time-to-success (TTS) and UTILITYRATE. TTS measures the number of seconds between presenting the results and the first satisfied click from the user, defined as any click for which the user stays on the linked page for sufficiently long. TTS value is capped and scaled to $[0, 1]$. UTILITYRATE is a more complex page-level metric of user satisfaction. It captures the interaction of a user with the page as a timeline of events (such as clicks) and their durations. The events are classified as revealing a positive or negative utility to the user and their contribution is proportional to their duration. UTILITYRATE takes values in $[-1, 1]$.

We evaluate a target policy based on a logistic regression classifier trained to predict clicks and using the predicted probabilities to score slates. We restrict the target policy to pick among the slates in our logs, so we know the ground truth slate-level reward. Since we know the query distribution, we can calculate the target policy's value exactly, and measure RMSE relative to this true value.

We compare our estimator (PI) with three baselines similar to those from Sec. 4.1: DM, IPS and ONPOLICY. DM uses regression trees over roughly 20,000 slate-level features.

Fig. 1 from the introduction shows that PI provides a consistent multiplicative improvement in RMSE over IPS, which suffers due to high variance. Starting at moderate sample sizes, PI also outperforms DM, which suffers due to substantial bias.

## 5 Discussion

In this paper we have introduced a new estimator (PI) for off-policy evaluation in combinatorial contextual bandits under a linearity assumption on the slate-level rewards. Our theoretical and empirical analysis demonstrates the merits of the approach. The empirical results show a favorable bias-variance tradeoff. Even in datasets and metrics where our assumptions are violated, the PI estimator typically outperforms all baselines. Its performance, especially at smaller sample sizes, could be further improved by designing doubly-robust variants [12] and possibly also incorporating weight clipping [34].

One promising approach to relax Assumption 1 is to posit a decomposition over pairs (or tuples) of slots to capture higher-order interactions such as diversity. More generally, one could replace slate spaces by arbitrary compact convex sets, as done in linear bandits. In these settings, the pseudoinverse estimator could still be applied, but tight sample-complexity analysis is open for future research.

## References


[1] Nima Asadi and Jimmy Lin. Training efficient tree-based models for document ranking. In *European Conference on Advances in Information Retrieval*, 2013.

[2] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 2002.

[3] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 2002.

[4] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis Charles, Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 2013.

[5] Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 2012.

[6] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *International Conference on Machine Learning*, 2005.

[7] Nicolo Cesa-Bianchi and Gábor Lugosi. Combinatorial bandits. *Journal of Computer and System Sciences*, 2012.

[8] Olivier Chapelle and Ya Zhang. A dynamic Bayesian network click model for web search ranking. In *International Conference on World Wide Web*, 2009.

[9] Olivier Chapelle, Donald Metlzer, Ya Zhang, and Pierre Grinspan. Expected reciprocal rank for graded relevance. In *Conference on Information and Knowledge Management*, 2009.

[10] Wei Chu, Lihong Li, Lev Reyzin, and Robert E Schapire. Contextual bandits with linear payoff functions. In *Artificial Intelligence and Statistics*, 2011.

[11] Varsha Dani, Thomas P. Hayes, and Sham M. Kakade. The price of bandit information for online optimization. In *Advances in Neural Information Processing Systems*, 2008.

[12] Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. In *International Conference on Machine Learning*, 2011.

[13] Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. Doubly robust policy evaluation and optimization. *Statistical Science*, 2014.

[14] Georges E. Dupret and Benjamin Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR Conference on Research and Development in Information Retrieval*, 2008.

[15] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems*, 2010.

[16] Fan Guo, Chao Liu, Anitha Kannan, Tom Minka, Michael Taylor, Yi-Min Wang, and Christos Faloutsos. Click chain model in web search. In *International Conference on World Wide Web*, 2009.

[17] Katja Hofmann, Lihong Li, Filip Radlinski, et al. Online evaluation for information retrieval. *Foundations and Trends in Information Retrieval*, 2016.

[18] Daniel G Horvitz and Donovan J Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 1952.

[19] Satyen Kale, Lev Reyzin, and Robert E Schapire. Non-stochastic bandit slate problems. In *Advances in Neural Information Processing Systems*, 2010.

[20] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M Henne. Controlled experiments on the web: survey and practical guide. *Knowledge Discovery and Data Mining*, 2009.

[21] Akshay Krishnamurthy, Alekh Agarwal, and Miroslav Dudík. Efficient contextual semi-bandit learning. *Advances in Neural Information Processing Systems*, 2016.

[22] Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvári. Tight regret bounds for stochastic combinatorial semi-bandits. In *Artificial Intelligence and Statistics*, 2015.

[23] John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in Neural Information Processing Systems*, 2008.

[24] John Langford, Alexander Strehl, and Jennifer Wortman. Exploration scavenging. In *International Conference on Machine Learning*, 2008.

[25] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *International Conference on World Wide Web*, 2010.

[26] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *International Conference on Web Search and Data Mining*, 2011.

[27] Lihong Li, Imed Zitouni, and Jin Young Kim. Toward predicting the outcome of an a/b experiment for search relevance. In *International Conference on Web Search and Data Mining*, 2015.

[28] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 2008.

[29] Lijing Qin, Shouyuan Chen, and Xiaoyan Zhu. Contextual combinatorial bandit and its application on diversified online recommendation. In *International Conference on Data Mining*, 2014.

[30] Tao Qin and Tie-Yan Liu. Introducing LETOR 4.0 datasets. *arXiv:1306.2597*, 2013.

[31] Paat Rusmevichientong and John N Tsitsiklis. Linearly parameterized bandits. *Mathematics of Operations Research*, 2010.

[32] Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*, 2015.

[33] Niek Tax, Sander Bockting, and Djoerd Hiemstra. A cross-benchmark comparison of 87 learning to rank methods. *Information Processing and Management*, 2015.

[34] Yu-Xiang Wang, Alekh Agarwal, and Miroslav Dudik. Optimal and adaptive off-policy evaluation in contextual bandits. In *International Conference on Machine Learning*, 2017.

[35] Yue Wang, Dawei Yin, Luo Jie, Pengyuan Wang, Makoto Yamada, Yi Chang, and Qiaozhu Mei. Beyond ranking: Optimizing whole-page presentation. In *International Conference on Web Search and Data Mining*, pages 103–112, 2016.