COUNTERFACTUAL EVALUATION AND LEARNING FROM LOGGED USER FEEDBACK

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Adith Swaminathan

May 2017

COUNTERFACTUAL EVALUATION AND LEARNING FROM LOGGED USER FEEDBACK

Adith Swaminathan, Ph.D.

Cornell University 2017

Interactive systems that interact with and learn from user behavior are ubiquitous today. Machine learning algorithms are core components of such systems. In this thesis, we will study how we can re-use logged user behavior data to evaluate interactive systems and train their machine learned components in a principled way. The core message of the thesis is

- Using simple techniques from causal inference, we can improve popular machine learning algorithms so that they interact reliably.
- These improvements are effective and scalable, and complement current algorithmic and modeling advances in machine learning.
- They open further avenues for research in Counterfactual Evaluation and Learning to ensure machine learned components interact reliably with users and with each other.

This thesis explores two fundamental tasks — evaluation and training of interactive systems. Solving evaluation and training tasks using logged data is an exercise in counterfactual reasoning. So we will first review concepts from causal inference for counterfactual reasoning, assignment mechanisms, statistical estimation and learning theory. The thesis then contains two parts.

In the first part, we will study scenarios where unknown assignment mechanisms underlie the logged data we collect. These scenarios often arise in

learning-to-rank and learning-to-recommend applications. We will view these applications through the lens of causal inference and modularize the problem of building a good ranking engine or recommender system into two components — first, infer a plausible assignment mechanism and second, reliably learn to rank or recommend assuming this mechanism was active when collecting data.

The second part of the thesis focuses on scenarios where we collect logged data from past interventions. We will formalize these scenarios as batch learning from logged contextual bandit feedback. We will first develop better off-policy estimators for evaluating online user-centric metrics in information retrieval applications. In subsequent chapters, we will study the bias-variance trade-off when learning from logged interventions. This study will yield new learning principles, algorithms and insights into the design of statistical estimators for counterfactual learning.

The thesis outlines a few principles, tools, datasets and software that hopefully prove to be useful to you as you build your interactive learning system.

BIOGRAPHICAL SKETCH

Adith Swaminathan was born in Tuticorin, a port city on the Coromandel Coast of India, in 1988. He grew up in Mumbai, a megacity on the Konkan Coast of India. He is a Ph.D. candidate in the Department of Computer Science at Cornell University, and is advised by Prof. Thorsten Joachims. He loves to think about Artificial Intelligence and solve applications that grapple with ambiguity. His research focuses on machine learning for interactive systems.

Adith visited the ILPS group at the University of Amsterdam and interned with Microsoft Research (in Silicon Valley, Redmond and New York) during his graduate studies. Before Cornell, he was a strategist with Tower Research Capital and developed algorithms for automated high-frequency trading. He holds a BTech in Computer Science and Engineering from IIT Bombay and an MSc in Computer Science from Cornell University. He will join Microsoft Research, Redmond, as a full-time researcher after receiving his Ph.D.

Adith is a trained Carnatic vocalist, a style of Indian Classical Music, and enjoys playing football (soccer). He is an avid reader, gamer and puzzler.

Dedicated to my grandmother, Mangalam Radhananda Kishore.

ACKNOWLEDGEMENTS

Guru lēkayeṭuvaṇṭi guṇiki teliyaga pōdu

No matter how virtuous one is, without a preceptor,

it is not possible for him to know.

Saint Tyagaraja, Composer

1767 – 1847

I am deeply grateful to Thorsten Joachims for his unwavering support throughout my journey at Cornell. I humbly submit all that I will ever accomplish as *gurudakshina* (a tradition of Thanksgiving after completing formal studies). I feel blessed to have Johannes Gehrke, Éva Tardos and Robert Kleinberg on my Ph.D. committee. Their comments and questions have always been exceptionally creative, and their clarity has guided my research throughout. I am also thankful to the Cornell CS family — Lillian Lee, Rebecca Stewart and Charles Van Loan shaped my first (and lasting) impression of Cornell CS. I am forever indebted to Soumen Chakrabarti for introducing me to CS research during my Bachelor's thesis and encouraging me during my undergraduate studies at IIT Bombay. I attribute the best ideas in our contributions to my wonderful collaborators — Ruben Sipos, Krishnaram Kenthapadi, Anitha Kannan, David Grangier, Navdeep Bhulli, Tobias Schnabel, Peter Frazier, Akshay Krishnamurthy, Alekh Agarwal, Miroslav Dudík, John Langford, Maarten de Rijke and Damien Lefortier.

I cherish the many friendships that we forged at Cornell. I miss office space antics with Stavros Nikolaou, Sidharth Telang, Karn Seth, Zhiyuan Teo, Aurosish Mishra, Shrutarshi Basu, Chenhao Tan, Jonathan Park, Yexiang Xue and Lu Wang. Shout-out to the Dungeons & Dragons gang: Isaac Sheff, Laure Thompson, Matthew Milano, Fabian Muehlboeck and Ethan Cecchetti. I learned all I know about Robotics from Hema Koppula, Ashesh Jain and Abhishek Anand. I am grateful for MLDG; thanks to Anshumali Shrivastava, Shuo Chen, Amit Sharma, Nikos Karampatziakis and Ashudeep Singh. I enjoyed being a part of the NLP seminar; thanks to Claire Cardie, David Mimno, Ainur Yessenalina, Bishan Yang and Arzoo Katiyar.

Music helps me weather the rough patches in life. I am in awe of my guru, Shri M. Sankaranarayanan. I am glad my flat-mates Ashwinkumar Badanidiyuru, Wenlei Xie and Raghavendra Rajkumar tolerated me singing in the shower. Jamming with Moontae Lee, Aravind Sivakumar, Vikram Rao, Aparna Mahadevan, Karthik Sridharan, Sarah Tan, Sumita Mitra, Tirth Pandya and Maithra Raghu was a privilege. Cheers to SPICMACAY!

I learned a lot from my IIT Bombay friends, Prathmesh Prabhu, Omkar Wagh, Sunny Goyal, Nikhil Hooda and Aditya Parameswaran, to name a few. Growing up in Mumbai was a blast, with friends like K. P. Ashwin, M. Siddharth, Amit Abbi, Ashish Jain, Harshal Namdeo, Indrajeet Shinde, Binit Mishra and Reetika Singh. Thanks also to friends in Amsterdam, Anne Schuth, Artem Grotov and Daan Odijk, to name a few.

I am deeply grateful to my family, my parents Saraswathi and Swaminathan, my siblings Arun and Abinaya, and my relatives Prema and Raja Mani, Hema and Ramesh Krish, Ria and Nina. I am much obliged to my closest friends, Karthik Raman and Siddarth Chandrasekaran. You are family.

My work was supported in part through NSF Awards IIS-0812091, IIS-0905467, IIS-1012593, IIS-0911036, IIS-1247637, IIS-1217686, IIS-1513692, IIS-1615706, the JTCII Cornell-Technion Research Fund and also through a gift from Bloomberg. A big sorry to folks I did not name above and big thanks to all my colleagues, family, friends, mentors and collaborators for this exciting journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Interactive systems — i.e., systems that interact with and learn from user behavior — are ubiquitous today (e.g., in digital services like search, e-commerce and online entertainment) and will be more pervasive with the advent of cyber-physical systems like smart homes and self-driving cars. Consequently, we collect petabytes of user interaction data as a by-product of operating these systems and use it to personalize and evaluate new interactive systems.

Building an interactive system is hard, and requires good decision-making under uncertainty. Therefore machine learning algorithms — e.g., learning to rank for information retrieval and collaborative filtering for recommendations — are core components of these systems. The current practice for training and evaluating these algorithms is plagued by an expensive trial and error cycle. For evaluation, the industry standard is to deploy new systems in weeks-long randomized controlled experiments [60]. For training directly from user interactions, the current state of the art explore-exploit algorithms demand interactive experimental control over the actions of a system to decide under uncertainty effectively [14]. Alternatively, interactive systems are also designed with offline supervised machine learned components. Evaluation and training using these offline machine learning algorithms require supervised judgments. A small fraction of the collected user interaction data is, hence, annotated with supervised judgments at great cost for training such algorithms. In this thesis, we will explore some tools that achieve the goal of combining the realism of working with online user-centric evaluation metrics with the convenience of offline machine learning experimentation, by re-using logged user interaction data.

Our goal is challenging because the data we collect does not directly answer the questions we ask. The collected data contains records of user interactions with a prior system. The questions we typically ask are “How might a new version of the system interact with our users?” or “How can we make the system better?”. As a concrete example, imagine we are building a game recommendation engine for an online retailer like Steam[1]. Every time Alice logs on to the recommendation engine, we want to recommend a game she might enjoy. If she enjoys the recommendation, she might then buy the game. We gain revenue when our users buy games. The data we collect during such interactions contains records of games we recommended and the games our users purchased. A question we may ask is “How should we improve the recommendation policy to get more revenue?”.

We can effectively benchmark the revenue accrued under our existing recommendation policy by using the collected data. However, to understand how to improve our recommender system, we ideally want to know “Will Alice enjoy this other recommendation we can make?”. We would like answers to such questions without actually making every possible recommendation to Alice and seeing how she responds. Beyond understanding Alice’s individual preferences, we need a generalizable policy that can recommend effectively across our entire user population every time they log on to our service. Since a successful recommendation system also typically receives a large volume of traffic, we need scalable learning algorithms that produce good fast generalizable policies by directly using collected user interactions.

The dominant approach for developing such interaction policies today views the problem of learning to interact as a prediction problem — “Can we predict

[1]`https://store.steampowered.com/`

the *correct* action to take during any user interaction?". Framing the problem this way, and with access to data annotated with *correct* actions, we can then employ supervised machine learning algorithms to engineer good policies. We will frame the problem differently and try to understand "What are the outcomes of acting according to different policies?". These are fundamentally counterfactual ("What If?") questions. Re-using logged interactions to answer such questions requires a causal understanding of the mechanisms that shape the data we collect from interactive systems. We will focus on two specific counterfactual questions — how to evaluate and how to train interactive systems — and address how these underlying mechanisms may confound the answers to our questions. Individual chapters study these questions in several different applications. Throughout the thesis, we will see two themes. First, any advances we make in answering the evaluation question transfers neatly into improvements for the training problem. Second, all the algorithms that we develop build on well-understood standard machine learning algorithms. So improvements in the properties (e.g., computational efficiency) of these standard algorithms also carry over when training interactive systems.

1.1 Organization and Contributions

We will first review concepts from causal inference, Monte Carlo estimation and empirical process theory in Chapter 2. We will describe the problem of estimating user-centric metrics as an instance of Monte Carlo estimation (Section 2.1) and review techniques like importance sampling [58] to estimate them (Section 2.2). This review will help us understand statistical properties of estimators that rely on randomization and sampling distributions. These estimators are closely related to propensity scoring techniques [95] in causal estimation and missing data imputation. Section 2.3 will make this connection more explicit. The potential outcomes framework [85, 96] for causal inference will be fundamental in framing our counterfactual questions and drawing unbiased conclusions. Section 2.3 will also clarify the role of randomization in assignment mechanisms when evaluating quantities under counterfactual distributions. For instance, even if we do not know "Will Alice enjoy this other recommendation we can make?" we can still estimate the average effect on revenue when switching to a different interaction policy if we have some controlled randomized recommendations and subsequent user purchases data! Techniques from causal inference will allow us to bridge the discrepancy between the data-generating distributions and the counterfactual distributions we wish to study. We will finally review seminal work in the learning theory of an algorithm called Structural Risk Minimization [125] (Section 2.4). Statistical learning theory guarantees that this algorithm will yield good generalization performance even when only having access to samples drawn from a data distribution for training. This guarantee is remarkable — even if we do not know the distribution of users logging on to our game recommendation engine, we can guarantee the quality of trained models evaluated under this distribution. All our learning algorithms will build on this learning principle and carefully trade-off bias against variance during statistical learning. Putting causal inference and statistical learning theory together, we can hope to answer counterfactual questions using only logged user interactions.

When re-using logged data, it will be useful to consider two scenarios separately; the thesis is divided into two parts studying each of these scenarios. In the first part, we will study scenarios where unknown assignment mechanisms underlie the logged data we collect ("Observational Feedback"). These scenarios arise when building ranking algorithms and recommender systems using user volunteered feedback. We will view these applications through the lens of causal inference and modularize the problem of building a good recommender system (Chapter 3) or ranking engine (Chapter 4) into two components — first, infer a plausible assignment mechanism and second, reliably learn to rank or recommend assuming this mechanism was active when collecting data.

We will visit the classic collaborative filtering learning model in Chapter 3. This model underlies a very popular algorithm [39] that can recommend an inventory of items to a population of users. Users sometimes provide ratings for items they consume. Can we use these ratings to build a good recommender? In Chapter 3 we will see that the answer is "Yes we can!" — if we understand the confounding from users' rating behavior (e.g. users may be more likely to provide ratings for popular items) when learning a recommendation policy. I conceptualized the potential outcomes model for recommendations and derived the resulting algorithms with Thorsten Joachims and Tobias Schnabel. Assisted by Tobias Schnabel and Ashudeep Singh, I contributed the experimental results reported in Chapter 3. Ashudeep ran the generative modeling baselines we compared against, and Tobias provided an independent implementation of our logistic regression propensity models to serve as a diagnostic.

We will then turn to Learning-to-rank (LTR) algorithms [78] (e.g., for presenting search results) in Chapter 4. Can we reliably use implicit feedback from user behavior on search results as a training signal for LTR algorithms? Again we will see that we should carefully reason about confounding effects from users' clicking behavior (e.g. users exhibit presentation and positions biases when clicking on top-ranked results) if we want to reliably rank search results. I conceptualized the potential outcomes model for search rankings with Thorsten Joachims, and conducted the real-world experiment on a live search engine reported in Chapter 4.

In the second part of the thesis, we will study the framework of Batch Learning from logged contextual Bandit Feedback (BLBF) [8, 113]. This framework is useful to reason about learning problems with access to interventional data ("Logged Interventional Feedback"). In Chapter 5 we will evaluate *online* user-centric metrics for real-world interactive systems using offline log data (also called off-policy estimation). We will find that all known off-policy estimators have undesirable behavior — either they make very few assumptions and require infeasible amounts of data to give accurate estimates, or they make very strong assumptions and give uncontrollably wrong answers in practice. We will develop an estimator in Chapter 5 that strikes a good balance between these extremes. I developed the estimator and its bias analysis jointly with Miroslav Dudík, Akshay Krishnamurthy, Alekh Agarwal and John Langford, and I contributed the semi-synthetic evaluation and optimization experiments reported in Chapter 5.

In Chapter 6 we will discover that counterfactual learning using off-policy estimators often fails for the same reasons that importance sampling can sometimes fail. We will revisit Structural Risk Minimization [125] and derive a new learning principle that remains robust by reasoning about the variance of off-policy estimators. We will then find that all known learning methods in these settings are vulnerable to a new kind of overfitting, called propensity overfitting [116]. Chapter 7 remedies this issue by employing the self-normalized importance sampling estimator (reviewed in Section 2.2) and empirically demonstrates its resistance to propensity overfitting. I conceptualized the learning principles and algorithms jointly with Thorsten Joachims, and contributed all the experiments reported in Chapters 6 and 7. These chapters contribute a careful analysis of a bias-variance trade-off in BLBF problems and reveal new learning principles, algorithms and insights for designing off-policy estimators.

Chapter 8 includes pointers to collections of datasets, software and additional reading material for counterfactual analysis in learning systems. And Chapter 9 concludes the thesis with directions for future work.

1.2 Experiment Methodology

When experimenting with our proposed techniques, we will use two conceptually distinct experiment setups. The first is real-world performance — "Do offline computed metrics agree with online performance after deploying systems in practice?" and "Does offline learning find good models that reliably interact when deployed?". The second is performance in semi-synthetic experiments — these experiments help us understand the limits of our proposed techniques and discover avenues for improvement, To setup semi-synthetic experiments, in the applications we study, annotated datasets were collected at great cost for use with standard supervised machine learning algorithms. In the game recommendation example, these annotations will tell us "Will Alice enjoy this recommendation?" for every possible recommendation we can make. We can construct realistic simulations using this data by withholding information about counterfactual recommendations.

CHAPTER 2

BACKGROUND

2.1 Schematic of Interactive Systems

Interactive systems have a characteristic interaction loop illustrated by the cartoon in Figure 2.1. Consider the video game recommendation example of Chapter 1. Alice issues a command (or query, topic, user, context, stimulus) to the system — denoted x, requesting a game recommendation. The system responds with an action (or prediction, document, recommendation, result, response) — denoted y, suggesting a new game. Alice then interacts with this response (e.g., she may hover over the snippet, click-through to read more about the game, buy it or abandon everything and seek out pictures of cats instead). The system can measure some of her behavior — denoted by δ, encoding her feedback.

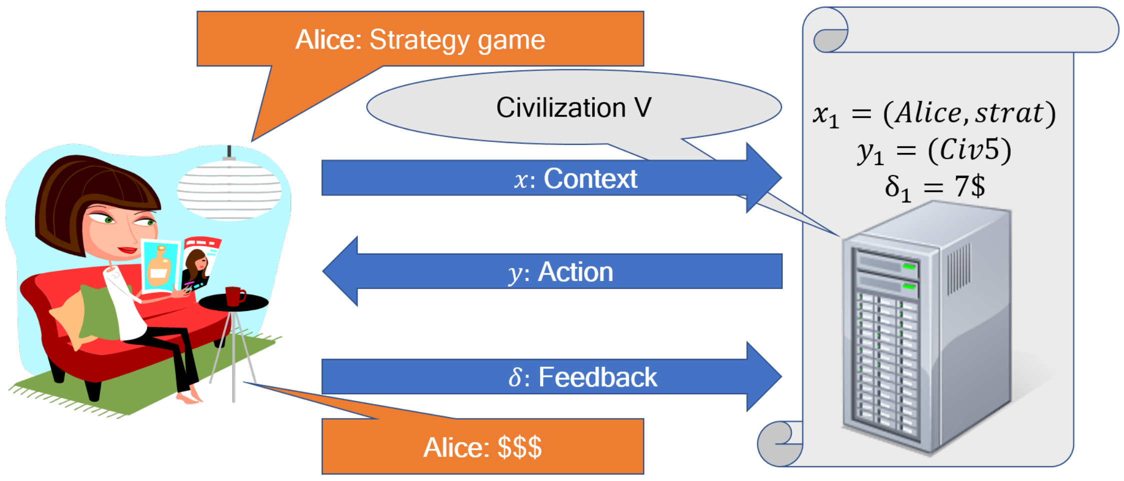


Figure 2.1: Interaction schematic of interactive systems.

This $x \mapsto y \mapsto \delta$ schematic occurs naturally in many applications that extend beyond interactive systems. In this thesis, we will study systems that interact in contextual bandit scenarios (Chapters 5, 6, 7), collaborative filtering for recommendations (Chapter 3) and systems that learn to rank (Chapter 4). Contextual bandit settings are a general framework for decision-making under uncertainty. x is a context (or side-information) that the environment provides to a decision-maker, y is an action the decision-maker takes and δ is the reward that the environment reveals for taking action y in context x. In recommendation settings x is a user, y is an item and δ denotes a rating (e.g., star rating, "*like*") that the user voluntarily provided for the item. In learning to rank x is a query submitted by a user, y is a ranking of search results and δ is a measure of ranking quality derived from the user's response (e.g., clicks on search results).

The data we collect in all these applications (e.g., (x, y, δ) triplets logged during system operation) have many things in common. x and y are instances from a large universe of possible $\mathcal{X}$ and $\mathcal{Y}$ respectively. Crucially we only get to observe δ for the (x, y) pair during an interaction — not the δ's for every possible pair in the universe $\mathcal{X} \times \mathcal{Y}$. We will typically assume that δ is stochastically distributed $\delta \sim \Pr(\cdot \mid x, y)$. This assumption precludes adversarial settings (e.g., spammers who want to confuse the learning system). We typically do not know the mechanisms that generate the x we see (e.g., "Why does Alice want a strategy game in Figure 2.1?") and assume that they come from some fixed but (typically) unknown distribution $x \sim \Pr(\mathcal{X})$. In some cases — like collaborative filtering (Chapter 3) — we will know $\Pr(\mathcal{X})$ by construction. There are two categories of mechanisms that generate y (see Section 2.3). In the first category, we are completely in control of generating y for x. This category is the "controlled setting" (known assignment mechanism). In the second category, we do not know the mechanisms that generated the y in our collected data, but we still want to re-use this data to build a system that responds well. This category is the "observational setting" (unknown assignment mechanism). We will tackle the observational setting by inferring an assignment mechanism and behaving "as if" we were in the controlled setting. When we are studying systems that deterministically respond to x, we will typically denote such systems by S, $y = S(x)$. We will quickly discover the importance of randomization when navigating a universe of unknown δ's (see Section 2.3). This discovery motivates the study of systems that randomize their responses. We denote such systems by π and say that they are operating according to a policy $y \sim \pi(x)$.

If the feedback δ is encoded as a number, we can summarize the quality of a system by a number too. Encoding feedback as a number is not the focus of this thesis, recent surveys [44, Chapter 3] and papers [35, 60] explore this topic in detail. The quality of a system is

$$V(S) = \int \mathbb{E}_{\delta \sim \Pr(\cdot|x,S(x))}[\delta] \Pr(x)dx, \qquad V(\pi) = \int \int \mathbb{E}_{\delta \sim \Pr(\cdot|x,y)}[\delta]\, \pi(y \mid x) \Pr(x)dydx. \tag{2.1}$$

This observation motivates the Monte Carlo estimation approach: simply deploy the systems S or π. During their operation, they collect samples $x \sim \Pr(X)$, $y = S(x)$ or $y \sim \pi(x)$ respectively, and their feedback δ. So it is "as if" we have a random sample to estimate the expectations in Equation (2.1). This is called On-Policy evaluation.

$$\hat{V}(S) = \hat{V}(\pi) = \frac{1}{n} \sum_{i=1}^{n} \delta_i. \tag{2.2}$$

The challenge, of course, is to evaluate and train systems (deterministic responders S or stochastic policies π) not by repeatedly deploying variants (trial-and-error) but by re-using already collected data (x, y, δ). Evaluating and training interactive systems with logged data is hard! The logging system influences the data we collect and we never directly see counterfactual outcomes for operating a new interaction policy.

A Note on Notation: In some applications we will have a clear interpretation of δ as a loss (i.e., systems should respond in such a way that δ is small). In such cases, we will replace $V(S) \mid V(\pi)$ by $R(S) \mid R(\pi)$ to denote the decision-theoretic risk of deploying $S \mid \pi$ respectively. We will reserve n to denote the size of collected data samples and the subscript $*_i$ to index into this sample (e.g., (x_i, y_i, δ_i)). Often we will deal with composite structured objects as responses. We will reserve boldface $\boldsymbol{y}$ to denote them and re-define y to refer to sub-parts of these structures. We will use subscript $*_j$ (e.g., $y = \boldsymbol{y}_j$ to refer to a document in a ranking of search results) to index these sub-parts. We will drop subscripts $\mathbb{E}_{**}$ when it is clear what we are taking expectations over. We will reserve the term "interventions" to describe interactions where the logging system actively randomized its actions.

2.2 Importance Sampling for Monte Carlo Estimation

We recap classic results in importance sampling [58, 86] here. For a detailed overview see the source material [86, Chapter 9] and the references therein.

Suppose we have the ability to sample a random variable y from a distribution $Q(Y)$ and we wish to estimate the expected value of a scalar-valued function $\delta(y)$ under a distribution $P(Y)$; $V := \mathbb{E}_{y \sim P(Y)}[\delta(y)]$. $Q(\cdot)$ is called the sampling distribution, $P(\cdot)$ is the target distribution. Assume that V exists.

If $Q(y) > 0$ whenever $\delta(y)P(y) \neq 0$,

$$\mathbb{E}_{y\sim Q}\left[\delta(y)\frac{P(y)}{Q(y)}\right] = \int \delta(y)P(y)dy = V.$$

This equation motivates the importance sampling estimator which uses a sample $y_i \overset{i.i.d.}{\sim} Q$,

$$\hat{V}_Q = \frac{1}{n}\sum_{i=1}^{n}\delta(y_i)\frac{P(y_i)}{Q(y_i)}.$$

The ratios $P(y)/Q(y)$ are called importance weights or likelihood ratios. This estimate is unbiased if Q has sufficient support (i.e., $\delta(y)P(y) \neq 0 \Rightarrow Q(y) > 0$).

$$\mathbb{E}_{\{y_i\}}\left[\hat{V}_Q\right] = V.$$

We can estimate the empirical variance of $\hat{V}_Q$ as

$$\hat{\sigma}^2_Q = \frac{1}{n}\sum_{i=1}^{n}\left(\delta(y_i)\frac{P(y_i)}{Q(y_i)} - \hat{V}_Q\right)^2$$

The sampling distribution Q^* that minimizes the variance of the importance sampling estimate ${\sigma^2}_Q$ [58] is $Q^*(y) \propto |\delta(y)|\, P(y)$ (except in trivial situations where $\mathbb{E}_{y\sim P}[|\delta(y)|] = 0$). Additionally, if we have a Q with a guarantee that the likelihood ratios $\frac{P(y)}{Q(y)} \leq c$, then we can assert that ${\sigma^2}_Q \leq c \cdot {\sigma^2}_P$.

The self-normalized importance sampling estimator [121] is an attractive alternative when we can only compute unnormalized versions of $\tilde{P}(y) = \alpha P(y)$ and $\tilde{Q}(y) = \beta Q(y)$.

$$\hat{V}_Q^{SN} = \frac{\sum_{i=1}^{n}\delta(y_i)P(y_i)/Q(y_i)}{\sum_{i=1}^{n}P(y_i)/Q(y_i)}.$$

Note that this estimate is computable using $\tilde{P}(y)/\tilde{Q}(y)$ as likelihood ratios. The self-normalized estimator is typically biased for finite samples $\{y_i\}$ and requires a stronger condition for asymptotic consistency. If $P(y) > 0 \Rightarrow Q(y) > 0$, then

$$\Pr(\lim_{n\to\infty}\hat{V}_Q^{SN} = V) = 1.$$

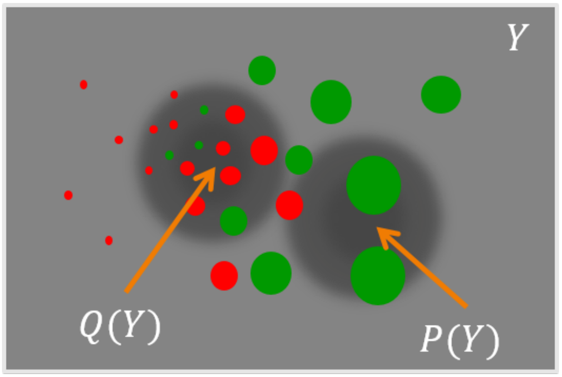


Figure 2.2: Importance sampling with P and Q having full support. The square represents the universe of possible $\mathcal{Y}$, circles represent samples $y_i \sim Q$. Their radius scales with the likelihood ratio $P(y_i)/Q(y_i)$. Colors reflect the observed value of $\delta(y_i)$. Importance sampling is especially useful when the red regions of $\mathcal{Y}$ are important for correctly estimating $V = \mathbb{E}_P[\delta]$.

The variance-optimal sampling distribution Q^*_{SN} for use with self-normalized estimators is $Q^*_{SN} \propto |\delta(y) - V|\, P(y)$ [41]. We cannot drive the variance of the self-normalized estimator to zero by choosing ever better sampling distributions. This estimator is still desirable because it is equivariant: $\hat{V}^{SN}_Q$ is exact when $\delta(y)$ is a constant. The vanilla estimator of Equation (2.2) however does not guarantee that $\hat{\mathbb{E}}\left[\delta(y) + C\right] = \hat{\mathbb{E}}\left[\delta(y)\right] + C$.

Diagnostics: Importance sampling can fail when the number of collected samples n is not sufficient to counteract the variability in likelihood ratios $P(y)/Q(y)$. One simple diagnostic uses the fact that $\mathbb{E}_{y \sim Q}\left[P(y)/Q(y)\right] = 1$ to detect whether

$$\hat{T}_Q := \frac{1}{n} \sum_{i=1}^{n} P(y_i)/Q(y_i) \simeq 1.$$

Another diagnostic is the *effective sample size*.

$$n^{eff} = \frac{n^2 \hat{T}_Q^2}{\sum_{i=1}^{n} P(y_i)^2/Q(y_i)^2}.$$

Instead of n^{eff} we can also estimate the variance $\hat{\sigma}_Q^2$ as a diagnostic. If $\hat{\sigma}_Q^2$ is very large it can mean that importance sampling has failed. However, this variance estimate uses the same weights that the estimate used. A bad variance estimate can subsequently mask situations where importance sampling failed. All these diagnostics are agnostic to $\delta(y)$. For a function-specific check [33], consider

$$n_\delta^{eff} = \frac{1}{\sum_{i=1}^n w_i(\delta)^2}, \qquad w_i(\delta) = \frac{|\delta(y_i)|\, P(y_i)/Q(y_i)}{\sum_{j=1}^n \left|\delta(y_j)\right| P(y_j)/Q(y_j)}.$$

If this effective sample size n_δ^{eff} is too small, then importance sampling has failed for estimating $V = \mathbb{E}_P[\delta]$. However, these diagnostics cannot reliably detect whether Q (and the sample $\{y_i\}$) has sufficient support over all important regions that matter for estimating V.

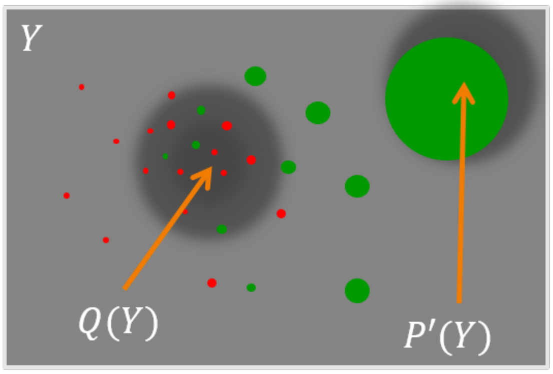


Figure 2.3: Failure mode when using importance sampling, especially likely to occur in "What if" simulations. If the target distribution P' is very different from the sampling distribution Q, one of the importance weights can be vastly larger than all the others. Although we collected n samples, we are estimating V with actually just one sample.

These diagnostics are critical when importance sampling is used to conduct "What if" simulations [121, 2] as illustrated in Figure 2.3. In these simulations we use the same samples $\{y_i\}$ drawn from Q to estimate $\mathbb{E}[\delta(y)]$ under many possible distributions $\{P\}$. This procedure can work well when the distributions in $\{P\}$ are all small "perturbations" around Q. However if one of the $P'(y) \in \{P\}$ is too different from Q, then importance sampling can fail (one of the diagnostics above will hopefully detect this). In Chapter 6 we will employ the $\hat{\sigma}_Q^2$ diagnostic to detect such cases because of its compatibility with statistical learning theory (see Section 2.4). We will additionally use the $\hat{T}_Q$ diagnostic in Chapter 7 to remedy new kinds of overfitting in counterfactual learning.

2.3 Potential Outcomes Framework for Causal Inference

We recap key elements of the Neyman-Rubin causal model (also called the Potential Outcomes Framework) [85, 96] here. For a detailed overview see recent books [50, 94, 77] and their references.

Recall the classic story of survivorship bias [126]. American airplanes were returning after engagements in Europe in World War 2, with non-uniform damage — there were more bullet holes in the fuselage and hardly any in the engine compartment. Military officers considered improving the plane designs by reinforcing the places where the planes were getting more damage (e.g., fuselage). Wald countered that armor should go where the bullet holes are not (the engine compartment). The reason planes were coming back with fewer hits to the engine is that planes that got hit in the engine were lethally damaged and not able to return. This phenomenon created a missing data problem confounded by survivorship bias and Wald's insight was to ask "Where are the missing holes?"

The Potential Outcomes Framework provides a formal model to reason about cause-effect (add armor, save planes) relationships. We will review [32] the definitions of causal effects, understand the role of randomization and see how the framework extends the logic of randomization to observationally collected data. Consider a binary treatment $y = \{0, 1\}$ (e.g., administer a placebo $y = 0$, versus administer a drug $y = 1$). An individual in a population has two potential outcomes $\{\delta_0, \delta_1\}$, one for each treatment. We can typically only observe one of these outcomes when the individual receives a specific treatment — the unobserved one is the *counterfactual outcome*. The Individual Causal Effect (**ICE**) is $\delta_1 - \delta_0$. **ICE** may differ across a population, so we typically study the population average of **ICE** — Average Treatment Effect (**ATE**).

$$\textbf{ATE} = \mathbb{E}\left[\delta_1 - \delta_0\right] = \mathbb{E}\left[\delta_1\right] - \mathbb{E}\left[\delta_0\right].$$

Given the heterogeneity of individual effects, we may also want to study average treatment effects restricted to some sub-populations of individuals. Let x denote some subset of observable covariates of an individual (e.g., the individual's demographic). The Conditional Average Treatment Effect (**CATE**) is

$$\textbf{CATE} = \mathbb{E}\left[\delta_1 - \delta_0 \mid X = x\right].$$

There are many other treatment effects we could study (e.g., Average Treatment Effect on the Treated). We will focus on estimating **ATE** — the techniques can be co-opted in a straightforward manner for estimating other averages. A core assumption that underlies many population-level aggregates is SUTVA (Stable Unit Treatment Value Assumption). Intuitively the outcomes $\{\delta_0, \delta_1\}$ for one individual should not depend on how other people are being treated. SUTVA lets us relate the expectations above to sample averages computed using the data we collect analogous to Equation (2.2).

The Fundamental Problem of Causal Inference [46] is that we can never directly observe a treatment effect since it requires both potential outcomes for an individual. We can however compute

$$V = \mathbb{E}\left[\delta_1 \mid y = 1\right] - \mathbb{E}\left[\delta_0 \mid y = 0\right].$$

We can ensure that V is an unbiased and consistent estimate of **ATE** using randomization! To see this, a sufficient condition for unbiasedness is

$$\begin{aligned}\mathbb{E}[\delta_1 \mid y = 1] = \mathbb{E}[\delta_1 \mid y = 0] &= \mathbb{E}[\delta_1], \\ \mathbb{E}[\delta_0 \mid y = 0] = \mathbb{E}[\delta_0 \mid y = 1] &= \mathbb{E}[\delta_0].\end{aligned}$$

This condition can be achieved simply by randomly assigning individuals in our population to be treated ($y = 1$) or untreated ($y = 0$). Situations where we can perform such a random assignment is called the *Controlled* Assignment setting. Uniform random assignment can often be prohibitive (e.g., trials for a drug that is a priori likely to be lethal to a sub-population). In such cases, we can use non-uniform randomization with a *probabilistic* assignment mechanism $\Pr(y \mid x) > 0$. A propensity score $\pi(x) := \Pr(y = 1 \mid x)$ is the probability of an individual getting assigned to a treatment conditional on their observable covariates. The Horvitz-Thompson estimator [47] uses these propensities to estimate treatment effects as

$$\hat{V} = \frac{1}{n} \sum_{i=1}^{n} \delta_i \left(\frac{\mathbf{1}\{y_i = 1\}}{\Pr(y = 1 \mid x_i)} - \frac{\mathbf{1}\{y_i = 0\}}{\Pr(y = 0 \mid x_i)} \right). \tag{2.3}$$

Closely related to the Controlled Assignment setting is the *Off-Policy* setting. In this setting, we cannot arbitrarily control assignment mechanisms to sculpt the data we collect. However, we know the assignment mechanism $\Pr(y \mid x) > 0$ that was active during data collection. We can still estimate causal effects using Equation (2.3) in this setting. The second part of the thesis (Chapters 5, 6 and 7) will employ Equation (2.3) to re-use data collected through interventions in this Off-Policy setting. In particular, we will see ways to improve Equation (2.3) in Chapter 5 for very high-dimensional treatments $\boldsymbol{y}$.

Often we have data that was collected from an unknown treatment regime. This setting, where we cannot directly manipulate treatment assignment or even know the underlying assignment mechanism, is called the *Observational* setting.

In such cases, we can use propensity matching or propensity weighting (a generalization of matching) methods. We will focus on one kind of weighting — Inverse Propensity Weighting — because of its similarity to Equation (2.3).

Suppose we know the propensities $\pi(x) := \Pr(y = 1 \mid x)$ of an individual getting assigned to a treatment. Observe that $y \perp\!\!\!\perp x \mid \pi(x)$. To reliably estimate causal effects from observational data, we must make an additional assumption: *unconfoundedness*

$$\{\delta_0, \delta_1\} \perp\!\!\!\perp y \mid x. \qquad \Rightarrow \qquad \{\delta_0, \delta_1\} \perp\!\!\!\perp y \mid \pi(x).$$

When we have unconfoundedness, we can estimate a propensity $\hat{\pi}(y \mid x)$ using observed (x_i, y_i, δ_i) data, and use these propensity estimates in Equation (2.3). Such a strategy is, however, vulnerable to misspecification of the propensity model. We will introduce propensity models for rankings (Chapter 4) and recommendations (Chapter 3), and study the effect of misspecification in detail.

Causal Estimation in all three settings — Controlled, Observational and Off-Policy — is closely related to missing data problems. After all, the essence of causal inference is a missing data problem and the techniques discussed above allow us to circumvent the missing potential (counterfactual) outcomes.

2.4 Learning Theory for Structural Risk Minimization

We recap the key results of Structural Risk Minimization [124] here. For a detailed overview see the source material [125, 124] and their references.

In statistical learning, the data we get is $(x_i, y_i^*) \overset{i.i.d.}{\sim} \Pr(\mathcal{X} \times \mathcal{Y})$ where $\Pr(\mathcal{X} \times \mathcal{Y})$ is a fixed unknown joint distribution over x (inputs) and y^* (labels). We additionally have a bounded loss function $\Delta(y^*, y) : |\Delta(\cdot, \cdot)| \leq M$, that can return the loss for any possible prediction y we make when the true label is y^*. The goal of learning is to pick a "good" function $h \in \mathcal{H}$ that can map inputs to labels $h : \mathcal{X} \mapsto \mathcal{Y}$. The risk of a function h is

$$R(h) = \int \Delta(y^*, h(x)) \Pr(x, y^*) d(x, y^*).$$

$R(h)$ is not computable since it requires knowledge of $\Pr(\mathcal{X} \times \mathcal{Y})$. The goal of learning is to pick a $\hat{h}$ using training data samples $(x_i, y_i^*)_{i=1}^n$ such that $R(\hat{h})$ is close to the minimal risk $\min_{h \in \mathcal{H}} R(h)$. Consider the empirical risk

$$\hat{R}(h) = \frac{1}{n} \sum_{i=1}^{n} \Delta(y_i^*, h(x_i)).$$

Consider a hypothesis class $\mathcal{H}$ with a finite VC-dimension $\mathcal{N}$[1]

Theorem. *With probability at least* $1 - \eta$*, for all* $h \in \mathcal{H}$ *simultaneously,*

$$R(h) \leq \hat{R}(h) + \frac{M\epsilon}{2} \left(1 + \sqrt{1 + \frac{4\hat{R}(h)}{M\epsilon}} \right) \tag{2.4}$$

$$\textit{where } \epsilon := \frac{4}{n} \Big[\mathcal{N} \left(1 + \ln \frac{2n}{\mathcal{N}} \right) - \ln \eta \Big].$$

This bound tells us that Empirical Risk Minimization (ERM) — i.e., select $\hat{h}^{ERM} := \operatorname{argmin}_{h \in \mathcal{H}} \hat{R}(h)$ — is a consistent learning strategy because the second term of Equation (2.4) vanishes when $n/\mathcal{N}$ is large. However, the bound can be very loose when the number of samples n is small compared to $\mathcal{N}$.

The Structural Risk Minimization (SRM) principle suggests a trade-off between lowering $\hat{R}(h)$ and the complexity of h, to guard against overfitting to

[1]Intuitively VC-dimension indicates the "capacity" of $\mathcal{H}$ to contain an h that can fit each arbitrary pattern we could see in the training data. See the source material [124, 1] and their references for a formal treatment.

the training data. It implements this trade-off when given a hierarchy of function sets $\mathcal{H}_0 \subset \mathcal{H}_1 \subset \mathcal{H}_2 \dots$ in the hypothesis space each with finite VC-dimension $\mathcal{N}_0 < \mathcal{N}_1 < \dots$ to pick $\hat{h}$. SRM computes the empirical risk minimizers $C = \{\hat{h}_0 \in \mathcal{H}_0, \hat{h}_1 \in \mathcal{H}_1, \dots\}$ and evaluates the bound in Equation (2.4) for every $h \in C$ to return the minimizer of the bound. Remarkably, for any $\Pr(\mathcal{X} \times \mathcal{Y})$, the SRM method converges to the best possible solution with probability 1 [124].

SRM requires the ability to compute the VC-dimension of a hypothesis class $\mathcal{H}_j$ efficiently and to perform ERM within each $\mathcal{H}_j$. For the first step, we have good characterizations for the capacity of linear function classes; for the second step, we have powerful algorithms that can implement ERM within such classes. We will exploit these results in Chapters 6 and 7 when developing learning principles for Batch Learning from Bandit Feedback (BLBF).

Part I

Learning from Observational Feedback

CHAPTER 3

BUILDING RECOMMENDER SYSTEMS USING CAUSAL INFERENCE

3.1 Chapter Notes

This chapter describes joint work with Tobias Schnabel, Ashudeep Singh, Navin Chandak and Thorsten Joachims. It is a lightly edited version of a conference publication [101].

We interpret recommending items to users as treatment assignments in causal studies (Section 2.3). Learning recommenders using user-volunteered ratings for items is then essentially causal estimation in the observational setting. Recalling the $x \mapsto y \mapsto \delta$ schematic of Section 2.1, δ is known for some user-item pairs (x, y) — the challenge is to understand the process that determines which (x, y, δ) are observed. The ratings we do not see are Missing Not at Random (MNAR). For instance, users may be more likely to volunteer a rating for an item they like. So we introduce propensity estimation models for user-volunteered datasets and reason about causal effects despite MNAR ratings. Our approach naturally fits into classic collaborative filtering algorithms for learning recommenders [6] and can be more broadly applied whenever the learning objectives behave like average treatment effects. Theoretically, we characterize the modeling bias when propensity models are misspecified. We discover an interesting new bias-variance trade-off when employing propensity models in learning algorithms. Our algorithm is the first principled discriminative model that achieves state of the art rating prediction performance in both real world and semi-synthetic datasets.

3.2 Introduction

Virtually all data for training recommender systems is subject to selection biases. For example, in a movie recommendation system users typically watch and rate those movies that they like, and rarely rate movies that they do not like [89]. Similarly, when an ad-placement system recommends ads, it shows ads that it believes to be of interest to the user, but will less frequently display other ads. Having observations be conditioned on the effect we would like to optimize (e.g. the star rating, the probability of a click, etc.) leads to data that is Missing Not At Random (MNAR) [77]. This phenomenon creates a widely-recognized challenge for evaluating recommender systems [79, 84].

We develop an approach to evaluate and train recommender systems that remedies selection biases in a principled, practical and highly effective way. Viewing recommendation from a causal inference perspective, we argue that exposing a user to an item in a recommendation system is an intervention analogous to exposing a patient to treatments in a medical study. In both cases, the goal is to accurately estimate the effect of new interventions (e.g. a new treatment policy or a new set of recommendations) despite incomplete and biased data due to self-selection or experimenter bias. By connecting recommendation to causal inference from experimental and observational data, we derive a principled framework for unbiased evaluation and learning of recommender systems under selection biases.

The main contribution of this chapter is four-fold. First, we show how estimating the quality of a recommendation system can be approached with propensity weighting techniques commonly used in causal inference [50], complete-case analysis [77] and other problems [25, 9, 111]. In particular, we derive unbiased estimators for a wide range of performance measures (e.g. MSE, MAE, DCG). Second, with these estimators in hand, we propose an Empirical Risk Minimization (ERM) framework for learning recommendation systems under selection bias, for which we derive generalization error bounds. Third, we use the ERM framework to derive a matrix factorization method that can account for selection bias while remaining conceptually simple and highly scalable. Fourth, we explore methods to estimate propensities in observational settings where selection bias is due to self-selection by the users, and we characterize the robustness of the framework against misspecified propensities.

We validate our conceptual and theoretical contributions in an extensive empirical evaluation. For the task of evaluating recommender systems, we show that our performance estimators can be orders-of-magnitude more accurate than standard estimators commonly used in the past [6]. For the task of learning recommender systems, we show that our new matrix factorization method substantially outperforms methods that ignore selection bias, as well as existing state-of-the-art methods that perform joint-likelihood inference under MNAR data [40]. Such performance is especially promising given the conceptual simplicity and scalability of our approach compared to joint-likelihood inference. We provide an implementation of our method and a new benchmark dataset online[1].

[1]https://www.cs.cornell.edu/~schnabts/mnar/

3.3 Related Work

Past work that explicitly dealt with the MNAR nature of recommendation data approached the problem as missing-data imputation based on the joint likelihood of the missing data model and the rating model [80, 79, 40]. This approach has led to sophisticated and highly complex methods to deal with MNAR ratings. We take a fundamentally different approach that treats both models separately, making our approach modular and scalable. Furthermore, our approach is robust to misspecification of the rating model, and we characterize how the overall learning process degrades gracefully under a misspecified missing data model. We empirically compare against the state-of-the-art joint likelihood model [40] in this chapter.

Related but different from the problem we consider is the issue of recommending using positive feedback alone [48, 74]. Related to this setting are also alternative approaches to learning with MNAR data which aim to avoid the problem by considering performance measures less affected by selection bias under mild assumptions [107, 108, 75]. One can view item popularity heuristic in a recall estimator [108] as a proxy for propensity in our framework. Similar to our work, weighted matrix factorization methods have been developed before [107, 108, 48], but with weighting schemes that are either heuristic or need to be tuned via cross-validation. In contrast, our weighted matrix factorization method enjoys rigorous learning guarantees in an ERM framework.

Propensity-based approaches have been widely used in causal inference from observational studies [50], as well as in complete-case analysis for missing data [77, 104] and in survey sampling [120]. However, their use in matrix

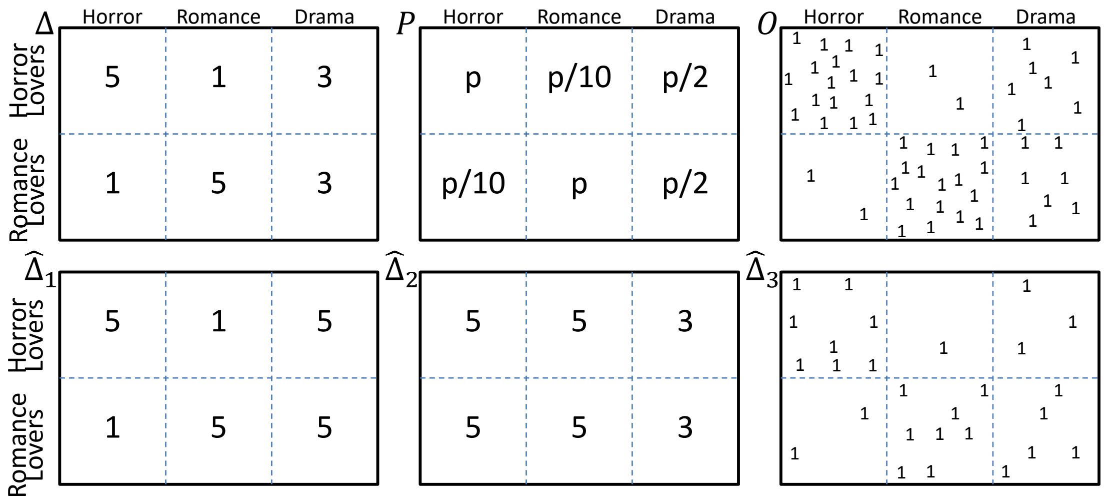


Figure 3.1: Movie-Lovers toy example. Top row: true rating matrix Δ, propensity matrix P, observation indicator matrix O. Bottom row: two rating prediction matrices $\hat{\Delta}_1$ and $\hat{\Delta}_2$, and intervention indicator matrix $\hat{\Delta}_3$.

completion is new to our knowledge. Weighting approaches are also widely used in domain adaptation and covariate shift, where data from one source is used to train for a different problem [49, 9, 111]. We will draw upon this work, especially the learning theory of weighting approaches [25, 24].

3.4 Unbiased Performance Estimation for Recommendation

Consider a toy example adapted from Steck's Movie-Lovers example [107] to illustrate the disastrous effect that selection bias can have on conventional evaluation using a test set of held-out ratings. Denote with $x \in \{1, \dots \mathcal{X}\}$ the users and with $y \in \{1, \dots \mathcal{Y}\}$ the movies. Figure 3.1 shows the matrix of true ratings $\Delta \in \mathbb{R}^{\mathcal{X} \times \mathcal{Y}}$ for our toy example, where a subset of users are "horror lovers" who rate all horror movies 5 stars and all romance movies 1 star. Similarly, there is a subset of "romance lovers" who rate just the opposite way. However, both groups give drama movies 3 stars. The binary matrix $O \in \{0,1\}^{\mathcal{X} \times \mathcal{Y}}$ in Figure 3.1 shows for which movies the users provided their rating to the system, $[O_{x,y} = 1] \Leftrightarrow [\Delta_{x,y} \text{ observed}]$. Our toy example shows a strong correlation between liking and rating a movie, and the matrix P describes the marginal probabilities $P_{x,y} = \Pr(O_{x,y} = 1)$ with which each rating is revealed. For this data, consider the following two evaluation tasks.

3.4.1 Task 1: Estimating Rating Prediction Accuracy

For the first task, we want to evaluate how well a predicted rating matrix $\hat{\Delta}$ reflects the true ratings in Δ. We can write standard evaluation measures like Mean Absolute Error (MAE) or Mean Squared Error (MSE) as:

$$R(\hat{\Delta}) = \frac{1}{\mathcal{X} \cdot \mathcal{Y}} \sum_{x=1}^{\mathcal{X}} \sum_{y=1}^{\mathcal{Y}} \delta_{x,y}(\Delta, \hat{\Delta}), \tag{3.1}$$

for an appropriately chosen $\delta_{x,y}(\Delta, \hat{\Delta})$.

$$\text{MAE:} \quad \delta_{x,y}(\Delta, \hat{\Delta}) = |\Delta_{x,y} - \hat{\Delta}_{x,y}|, \tag{3.2}$$

$$\text{MSE:} \quad \delta_{x,y}(\Delta, \hat{\Delta}) = (\Delta_{x,y} - \hat{\Delta}_{x,y})^2, \tag{3.3}$$

$$\text{Accuracy:} \quad \delta_{x,y}(\Delta, \hat{\Delta}) = \mathbf{1}\{\hat{\Delta}_{x,y} = \Delta_{x,y}\}. \tag{3.4}$$

Since Δ is only partially known, the conventional practice is to estimate $R(\hat{\Delta})$ using the average over only the observed entries,

$$\hat{R}_{naive}(\hat{\Delta}) = \frac{1}{|\{(x,y) : O_{x,y} = 1\}|} \sum_{(x,y):O_{x,y}=1} \delta_{x,y}(\Delta, \hat{\Delta}). \tag{3.5}$$

We call this the naïve estimator, and its naïvety leads to a gross misjudgment for the $\hat{\Delta}_1$ and $\hat{\Delta}_2$ given in Figure 3.1. Even though $\hat{\Delta}_1$ is clearly better than $\hat{\Delta}_2$ by any reasonable measure of performance, $\hat{R}_{naive}$ will reliably claim that $\hat{\Delta}_2$ has better

MAE than $\hat{\Delta}_1$. This error is due to selection bias since 1-star ratings are under-represented in the observed data and $\delta_{x,y}$ is correlated with $\Delta_{x,y}$. More generally, under selection bias, $\hat{R}_{naive}$ is a biased estimate of the true performance R [109]:

$$\mathbb{E}_O \left[\hat{R}_{naive}(\hat{\Delta}) \right] \neq R(\hat{\Delta}). \tag{3.6}$$

Before we design an improved estimator to replace $\hat{R}_{naive}$, let's turn to a related evaluation task.

3.4.2 Task 2: Estimating Recommendation Quality

Instead of evaluating the accuracy of predicted ratings, we may want to evaluate the quality of a particular recommendation more directly. To this effect, let's redefine $\hat{\Delta}$ to now encode recommendations as a binary matrix analogous to O, where $[\hat{\Delta}_{x,y} = 1] \Leftrightarrow [y \text{ is recommended to } x]$, limited to a budget of k recommendations per user. An example is $\hat{\Delta}_3$ in Figure 3.1. A reasonable way to measure the quality of a recommendation is the Cumulative Gain (CG) that the user derives from the recommended movies, which we define as the average star-rating of the recommended movies in our toy example[2]. We can again write CG in the form of Equation (3.1) with

$$\text{CG:} \qquad \delta_{x,y}(\Delta, \hat{\Delta}) = (\mathcal{Y}/k)\hat{\Delta}_{x,y} \cdot \Delta_{x,y}. \tag{3.7}$$

However, unless users have watched all movies in $\hat{\Delta}$, we cannot compute CG directly via Equation (3.1). Hence, we must answer the counterfactual question: "How well would our users have enjoyed themselves (in terms of CG), if they had followed our recommendations $\hat{\Delta}$ instead of watching (and rating) the

[2]More realistically, Δ would contain quality scores derived from indicators like "clicked" and "watched the movie to the end".

movies indicated in O?". Note that rankings of recommendations are similar to the set-based recommendation described above, and measures like Discounted Cumulative Gain (DCG), *DCG@k*, *Precision@k*, and others [3, 132] also fit in this setting. Let the values of $\hat{\Delta}$ in each row define the predicted ranking, then

$$\text{DCG:} \qquad \delta_{x,y}(\Delta, \hat{\Delta}) = (\mathcal{Y}/\log(rank(\hat{\Delta}_{x,y})))\Delta_{x,y}, \tag{3.8}$$

$$\text{Precision@k:} \qquad \delta_{x,y}(\Delta, \hat{\Delta}) = (\mathcal{Y}/k)\Delta_{x,y} \cdot \mathbf{1}\{rank(\hat{\Delta}_{x,y}) \leq k\}. \tag{3.9}$$

One approach, similar in spirit to condensed DCG [99], is to use the naïve estimator from Equation (3.5) again. However, this and similar estimators are biased for $R(\hat{\Delta})$ [89, 109].

To get unbiased estimates of recommendation quality despite missing observations, consider the following connection to estimating the average treatment effect of a given policy in causal inference, that was already explored in the contextual bandit setting [72, 30]. If we think of a recommendation as an intervention analogous to treating a patient with a specific drug, in both settings we want to estimate the effect of a new treatment policy (e.g. give drug A to women and drug B to men, or new recommendations $\hat{\Delta}$). The challenge in both cases is that we have only partial knowledge of how much certain patients (users) benefited from certain treatments (recommended items) (i.e., $\Delta_{x,y}$ with $O_{x,y} = 1$), while the vast majority of potential outcomes in Δ is unobserved.

3.4.3 Propensity-Scored Performance Estimators

The key to handling selection bias in both of the tasks mentioned above lies in understanding the process that generates the observation pattern in O. This process is typically called the *Assignment Mechanism* in causal inference [50] or the *Missing Data Mechanism* in missing data analysis [77]. We differentiate the following two settings:

Experimental Setting. In this setting, the assignment mechanism is under the control of the recommendation system. An example is an ad-placement system that controls which ads to show to which user.

Observational Setting. In this setting, the users are part of the assignment mechanism that generates O. An example is an online streaming service for movies, where users self-select the movies they watch and rate.

In this chapter, we assume that the assignment mechanism is probabilistic, meaning that the marginal probability $P_{x,y} = \Pr(O_{x,y} = 1)$ of observing an entry $\Delta_{x,y}$ is non-zero for all user/item pairs. This assumption ensures that, in principle, every element of Δ could be observed, even though any particular collection of observed ratings O reveals only a small subset. We refer to $P_{x,y}$ as the *propensity* of observing $\Delta_{x,y}$. In the *experimental* setting, we know the matrix P of all propensities, since we have implemented the assignment mechanism. In the *observational* setting, we will need to estimate P from the observed matrix O. We defer the discussion of propensity estimation to Section 3.6, and focus on the *experimental* setting first.

IPS Estimator: The Inverse-Propensity-Scoring (IPS) estimator [120, 77, 50], which applies equally to the task of rating prediction evaluation as to the task of recommendation quality estimation, is defined as,

$$\hat{R}_{IPS}(\hat{\Delta} \mid P) = \frac{1}{\mathcal{X} \cdot \mathcal{Y}} \sum_{(x,y):O_{x,y}=1} \frac{\delta_{x,y}(\Delta, \hat{\Delta})}{P_{x,y}}. \quad (3.10)$$

Unlike the naive estimator $\hat{R}_{naive}$, the IPS estimator is unbiased for any probabilistic assignment mechanism. Note that the IPS estimator only requires the marginal probabilities $P_{x,y}$ and unbiasedness is not affected by dependencies within O:

$$\begin{aligned}\mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta} \mid P)\right] &= \frac{1}{\mathcal{X}\cdot\mathcal{Y}}\sum_x\sum_y \mathbb{E}_{O_{x,y}}\left[\frac{\delta_{x,y}(\Delta,\hat{\Delta})}{P_{x,y}}O_{x,y}\right] \\ &= \frac{1}{\mathcal{X}\cdot\mathcal{Y}}\sum_x\sum_y \delta_{x,y}(\Delta,\hat{\Delta}) = R(\hat{\Delta}). \qquad (3.11)\end{aligned}$$

However, to characterize the variability of the IPS estimator, we assume that observations are independent given P, which corresponds to a multivariate Bernoulli model where each $O_{x,y}$ is a biased coin flip with probability $P_{x,y}$. The following proposition (proof in Appendix A.1) provides some intuition about how the accuracy of the IPS estimator changes as the propensities become more "non-uniform".

Proposition 1 (Tail Bound for IPS Estimator). *For any given Δ, let P be the independent Bernoulli probabilities of observing each entry of Δ. For any $\hat{\Delta}$, with probability $1-\eta$, the IPS estimator $\hat{R}_{IPS}(\hat{\Delta} \mid P)$ does not deviate from the true $R(\hat{\Delta})$ by more than:*

$$\left|\hat{R}_{IPS}(\hat{\Delta} \mid P) - R(\hat{\Delta})\right| \le \frac{1}{\mathcal{X}\cdot\mathcal{Y}}\sqrt{\frac{\log\frac{2}{\eta}}{2}\sum_{x,y}\rho_{x,y}^2},$$

where $\rho_{x,y} = \frac{\delta_{x,y}(\Delta,\hat{\Delta})}{P_{x,y}}$ if $P_{x,y} < 1$, and $\rho_{x,y} = 0$ otherwise.

To illustrate this bound, consider the case of uniform propensities $P_{x,y} = p$. Under uniform propensities, $n = \sum P_{x,y} = p\mathcal{X}\mathcal{Y}$ elements of Δ are revealed in expectation. In this case, the bound is $O(1/(p\sqrt{\mathcal{X}\mathcal{Y}}))$. If the $P_{x,y}$ are non-uniform, the bound can be much larger even if the expected number of revealed elements, $\sum P_{x,y}$ is n. We are paying for the unbiasedness of IPS with variability, and we will evaluate whether this price is well spent throughout the chapter.

SNIPS Estimator: One technique that can reduce variability is the use of control variates [86]. For instance, we know that $\mathbb{E}_O\left[\sum_{(x,y):O_{x,y}=1} \frac{1}{P_{x,y}}\right] = \mathcal{X} \cdot \mathcal{Y}$. This observation yields the Self-Normalized IPS (SNIPS) estimator [121, 116]

$$\hat{R}_{SNIPS}(\hat{\Delta} \mid P) = \frac{\sum_{(x,y):O_{x,y}=1} \frac{\delta_{x,y}(\Delta,\hat{\Delta})}{P_{x,y}}}{\sum_{(x,y):O_{x,y}=1} \frac{1}{P_{x,y}}}. \tag{3.12}$$

The SNIPS estimator often has lower variance than the IPS estimator but has a small bias [42].

3.4.4 Empirical Illustration of Estimators

To illustrate the effectiveness of the proposed estimators we conducted an experiment on the semi-synthetic ML100K dataset described in Section 3.7.2. For this dataset, Δ is completely known so that we can compute true performance via Equation (3.1). We chose the probability $P_{x,y}$ of observing a rating $\Delta_{x,y}$ to mimic the observed marginal rating distribution in the original ML100K dataset (see Section 3.7.2) such that, on average, 5% of the Δ matrix was revealed.

	MAE			
	True	IPS	SNIPS	Naïve
REC_ONES	0.102	0.102 ± 0.007	0.102 ± 0.007	0.011 ± 0.001
REC_FOURS	0.026	0.026 ± 0.000	0.026 ± 0.000	0.173 ± 0.001
ROTATE	2.579	2.581 ± 0.031	2.579 ± 0.012	1.168 ± 0.003
SKEWED	1.306	1.304 ± 0.012	1.304 ± 0.009	0.912 ± 0.002
COARSENED	1.320	1.314 ± 0.015	1.318 ± 0.005	0.387 ± 0.002

Table 3.1: Mean and standard deviation of the Naïve, IPS, and SNIPS estimators compared to true MAE for five predicted rating matrices on a semi-synthetic experiment on the ML100K dataset (see Section 3.7.2 for details).

Table 3.1 shows the results for estimating rating prediction accuracy via MAE and Table 3.2 shows the results for evaluating recommendation quality

	DCG@50			
	True	IPS	SNIPS	Naïve
REC_ONES	30.76	30.64 ± 0.75	30.66 ± 0.74	153.07 ± 2.13
REC_FOURS	52.00	51.98 ± 0.41	52.08 ± 0.58	313.48 ± 2.36
ROTATE	12.90	13.00 ± 0.85	12.99 ± 0.83	1.38 ± 0.09
SKEWED	24.59	24.55 ± 0.92	24.58 ± 0.93	54.87 ± 1.03
COARSENED	46.45	46.45 ± 0.53	46.44 ± 0.70	293.27 ± 1.99

Table 3.2: Mean and standard deviation of the Naïve, IPS, and SNIPS estimators compared to true DCG@50 for five recommender systems on a semi-synthetic experiment on the ML100K dataset (see Section 3.7.2 for details).

via DCG@50 for the following five prediction matrices $\hat{\Delta}_i$. Let $|\Delta = r|$ be the number of r-star ratings in Δ.

REC_ONES: $\hat{\Delta}$ is identical to the true rating matrix Δ except that we flip $|\{(x, y) : \Delta_{x,y} = 5\}|$ randomly selected true ratings of 1 to 5. This procedure ensures half of the predicted fives are true fives, and half are true ones.

REC_FOURS: Same as REC_ONES, but flipping 4-star ratings instead.

ROTATE: For each predicted rating $\hat{\Delta}_{x,y} = \Delta_{x,y} - 1$ when $\Delta_{x,y} \geq 2$, and $\hat{\Delta}_{x,y} = 5$ when $\Delta_{x,y} = 1$.

SKEWED: Predictions $\hat{\Delta}_{x,y}$ are sampled from a normal distribution centered at $\Delta_{x,y}$, $\hat{\Delta}_{x,y} \sim \mathcal{N}(\hat{\Delta}^{raw}_{x,y} | \mu = \Delta_{x,y}, \sigma = \frac{6-\Delta_{x,y}}{2})$ and clipped to the interval $[0, 6]$.

COARSENED: If the true rating $\Delta_{x,y} \leq 3$, then $\hat{\Delta}_{x,y} = 3$. Otherwise $\hat{\Delta}_{x,y} = 4$.

Rankings for DCG@50 were created by sorting items according to $\hat{\Delta}_i$ for each user. In Table 3.1 and Table 3.2, we report the average and standard deviation of estimates over 50 samples of O from P. We see that the mean IPS estimate perfectly matches the true performance for both MAE and DCG@50 as expected. The bias of SNIPS is negligible as well. The naïve estimator is severely biased, and its estimated MAE incorrectly ranks the prediction matrices $\hat{\Delta}_i$ (e.g. it ranks the performance of *REC_ONES* higher than *REC_FOURS*). The standard deviation of IPS and SNIPS is substantially smaller than the bias that Naïve incurs. Furthermore, SNIPS manages to reduce the standard deviation of IPS for MAE but not for DCG. We will empirically study these estimators more comprehensively in Section 3.7.

3.5 Propensity-Scored Recommendation Learning

We will now use the unbiased estimators from Section 3.4.3 in an Empirical Risk Minimization (ERM) framework for learning, prove generalization error bounds and derive a matrix factorization method for rating prediction.

3.5.1 ERM for Recommendation with Propensities

Empirical Risk Minimization underlies many successful learning algorithms like SVMs [26], Boosting [100], and Deep Learning [7]. Weighted ERM approaches have been effective for cost-sensitive classification, domain adaptation and covariate shift [137, 9, 111]. We adapt ERM to our setting by realizing that Equation (3.1) corresponds to an expected loss (i.e. risk) over the data generating process $\Pr(O \mid P)$. Given a sample from $\Pr(O \mid P)$, the IPS estimator from Equation (3.10) is the Empirical Risk $\hat{R}(\hat{\Delta})$ that estimates $R(\hat{\Delta})$ for any $\hat{\Delta}$.

Definition 1 (Propensity-Scored ERM for Recommendation)**.** *Given training observations O from Δ with marginal propensities P, given a hypothesis space $\mathcal{H}$ of pre-*

dictors $\hat{\Delta}$, and given a loss function $\delta_{x,y}(\Delta, \hat{\Delta})$, ERM selects the $\hat{\Delta} \in \mathcal{H}$ that optimizes:

$$\hat{\Delta}^{ERM} = \underset{\hat{\Delta} \in \mathcal{H}}{\operatorname{argmin}} \left\{ \hat{R}_{IPS}(\hat{\Delta} \mid P) \right\}. \tag{3.13}$$

Using the *SNIPS* estimator does not change the argmin minimizer. To illustrate the validity of the propensity-scored ERM approach, we state the following generalization error bound (proof in Appendix A.2) which follows analogous proofs for generalization in domain adaptation [24]. We consider only finite $\mathcal{H}$ for the sake of exposition.

Theorem 1 (Propensity-Scored ERM Generalization Error Bound)**.** *For any finite hypothesis space of predictions $\mathcal{H} = \{\hat{\Delta}_1, \ldots \hat{\Delta}_{|\mathcal{H}|}\}$ and loss $0 \le \delta_{x,y}(\Delta, \hat{\Delta}) \le M$, the true risk $R(\hat{\Delta}^{ERM})$ of the empirical risk minimizer $\hat{\Delta}^{ERM}$ from $\mathcal{H}$ using the IPS estimator, given training observations O from Δ with independent Bernoulli propensities P, is bounded with probability $1 - \eta$ by:*

$$R(\hat{\Delta}^{ERM}) \le \hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid P) + \frac{M}{\mathcal{X} \cdot \mathcal{Y}} \sqrt{\frac{\log(2|\mathcal{H}|/\eta)}{2}} \sqrt{\sum_{x,y} \frac{1}{P_{x,y}^2}}. \tag{3.14}$$

3.5.2 Propensity-Scored Matrix Factorization

We now use propensity-scored ERM to derive a matrix factorization method for the problem of rating prediction. Assume a standard rank-d-restricted and ℓ_2-regularized matrix factorization model $\hat{\Delta}_{x,y} = \mathbf{v}_x^T \mathbf{w}_y + a_x + b_y + c$ with a user, item and global offsets as our hypothesis space $\mathcal{H}$. Under this model, propensity-scored ERM leads to the following training objective:

$$\underset{V,W,A}{\operatorname{argmin}} \left[\sum_{(x,y):O_{x,y}=1} \frac{\delta_{x,y}(\Delta, V^T W + A)}{P_{x,y}} + \lambda \left(\|V\|_F^2 + \|W\|_F^2 \right) \right] \tag{3.15}$$

where A encodes the offset terms and $\hat{\Delta}^{ERM} = V^T W + A$. Except for the propensities $P_{x,y}$ that act as weights for each loss term, the training objective is identical to the standard incomplete matrix factorization objective [62, 107, 48] with MSE (using Equation (3.3)) or MAE (using Equation (3.2)). So, we can readily draw upon existing optimization algorithms that can efficiently solve the training problem at scale [38, 133]. For our experiments, we use limited-memory BFGS [16] on a single machine. Our implementation is available online[3].

Conventional incomplete matrix factorization is a special case of Equation (3.15) for MCAR (Missing Completely At Random) data, i.e., all propensities $P_{x,y}$ are equal. Solving this training objective for other $\delta_{x,y}$ that are non-smooth and non-differentiable is more challenging, but possible avenues exist [54, 20]. Finally, note that other recommendation methods (e.g. max-margin approaches [130], non-negative factorization [76]) can be adapted to propensity scoring as well.

3.6 Propensity Estimation for Observational Data

We now turn to the Observational Setting where propensities need to be estimated. One might be worried that we need to perfectly reconstruct all propensities for effective learning. However, as we will show, we merely need estimated propensities that are "better" than the naïve assumption of observations being revealed uniformly, i.e., $P = |\{(x, y) : O_{x,y} = 1\}| / (\mathcal{X} \cdot \mathcal{Y})$ for all users and items. The following characterizes "better" propensities in terms of the bias they induce and their effect on the variability of the learning process.

[3]https://www.cs.cornell.edu/~schnabts/mnar/

Lemma 2 (Bias of IPS Estimator under Inaccurate Propensities)**.** *Let P be the marginal probabilities of observing an entry of the rating matrix Δ, and let $\hat{P}$ be the estimated propensities such that $\hat{P}_{x,y} > 0$ for all x, y. The bias of the IPS estimator Equation* (3.10) *using $\hat{P}$ is:*

$$\text{bias}\left(\hat{R}_{IPS}(\hat{\Delta} \mid \hat{P})\right) = \mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta} \mid \hat{P})\right] - R(\hat{\Delta}) \quad = \quad \sum_{x,y} \frac{\delta_{x,y}(\Delta, \hat{\Delta})}{\mathcal{X} \cdot \mathcal{Y}} \left[1 - \frac{P_{x,y}}{\hat{P}_{x,y}}\right]. \tag{3.16}$$

In addition to bias, the following generalization error bound (proof in Appendix A.4) characterizes the overall impact of the estimated propensities on the learning process.

Theorem 3 (Propensity-Scored ERM Generalization Error Bound under Inaccurate Propensities)**.** *For any finite hypothesis space of predictions $\mathcal{H} = \{\hat{\Delta}_1, \ldots \hat{\Delta}_{|\mathcal{H}|}\}$, the transductive prediction error of the empirical risk minimizer $\hat{\Delta}^{ERM}$, using the IPS estimator with estimated propensities $\hat{P}$ ($\hat{P}_{x,y} > 0$) and given training observations O from Δ with independent Bernoulli propensities P, is bounded by:*

$$\begin{aligned} R(\hat{\Delta}^{ERM}) \quad \leq \quad & \hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P}) + \frac{M}{\mathcal{X} \cdot \mathcal{Y}} \sum_{x,y} \left| 1 - \frac{P_{x,y}}{\hat{P}_{x,y}} \right| \\ & + \frac{M}{\mathcal{X} \cdot \mathcal{Y}} \sqrt{\frac{\log(2|\mathcal{H}|/\eta)}{2}} \sqrt{\sum_{x,y} \frac{1}{\hat{P}^2_{x,y}}}. \end{aligned} \tag{3.17}$$

The bound shows a bias-variance trade-off that does not occur in conventional ERM. In particular, the bound suggests that it may be beneficial to overestimate small propensities, if this reduces variability more than it increases bias.

3.6.1 Propensity Estimation Models.

Recall that our goal is to estimate the probabilities $P_{x,y}$ with which ratings for user x and item y will be observed. In general, the propensities

$$P_{x,y} = \Pr(O_{x,y} = 1 \mid \mathbf{f}, \mathbf{f}^{hid}, \Delta) \tag{3.18}$$

can depend on some observable features $\mathbf{f}$ (e.g., the predicted rating that accompanied the item when it was displayed to the user), unobservable features $\mathbf{f}^{hid}$ (e.g., whether the item was recommended by a friend), and the ratings Δ. It is reasonable to assume that $O_{x,y}$ is independent of the new predictions $\hat{\Delta}$ (and therefore independent of $\delta_{x,y}$) once we take the observable features into account. The following outlines two simple propensity estimation methods, but there are several other techniques [83] to cater to domain-specific needs.

Propensity Estimation via Naive Bayes: The first method estimates the propensity $\Pr(O_{x,y}|\mathbf{f}, \mathbf{f}^{hid}, \Delta)$ by assuming that dependencies between covariates $\mathbf{f}$, $\mathbf{f}^{hid}$ and other ratings $\Delta_{x',y'}$ are negligible. Equation (3.18) then reduces to $\Pr(O_{x,y} \mid \Delta_{x,y})$, similar to the CPT-v missing data model [79]. We can treat $\Delta_{x,y}$ as observed, since we only need the propensities for observed entries to compute IPS and SNIPS. This yields the *Naïve Bayes* propensity estimator:

$$\Pr(O_{x,y} = 1 \mid \Delta_{x,y} = r) = \frac{\Pr(\Delta_{x,y} = r \mid O_{x,y} = 1) \Pr(O_{x,y} = 1)}{\Pr(\Delta_{x,y} = r)}. \tag{3.19}$$

By the Naïve Bayes independence assumptions, $\Pr(O_{x,y} = 1) = \Pr(O = 1)$, where we dropped the subscripts to reflect that parameters are tied across all x and y. Maximum likelihood estimate of this parameter is simply

$$\hat{\Pr}(O = 1) = \frac{|\{(x,y) : O_{x,y} = 1\}|}{\mathcal{X} \cdot \mathcal{Y}}.$$

Similarly, $\Pr(\Delta_{x,y} = r \mid O_{x,y} = 1) = \Pr(\Delta = r \mid O = 1)$ and

$$\hat{\Pr}(\Delta = r \mid O = 1) = \frac{|\{(x,y) : O_{x,y} = 1 \wedge \Delta_{x,y} = r|}{\{(x,y) : O_{x,y} = 1\}|}.$$

Finally, $\Pr(\Delta_{x,y} = r) = \Pr(\Delta = r)$ due to parameter sharing from the independence assumptions. To estimate these r parameters $\Pr(\Delta = r)$, the proportion of different rating values in Δ, we need a small sample of MCAR (Missing Completely At Random) data that is not biased by the observation mechanism P.

Propensity Estimation via Logistic Regression The second propensity estimation approach we explore (which does not require a sample of MCAR data) is based on logistic regression and is commonly used in causal inference [94]. It also starts from Equation (3.18) but aims to find model parameters ϕ such that O becomes independent of unobserved $\mathbf{f}^{hid}$ and Δ, i.e., $\Pr(O_{x,y} \mid \mathbf{f}, \mathbf{f}^{hid}, \Delta) = \Pr(O_{x,y} \mid \mathbf{f}, \phi)$. The main modeling assumption is that there exists a $\phi = (\alpha, \beta, \gamma)$ such that $P_{x,y} = \sigma\left(\alpha^T \mathbf{f}_{x,y} + \beta_x + \gamma_y\right)$. Here, $\mathbf{f}_{x,y}$ is a vector encoding all observable information about a user-item pair (e.g., user demographics, whether an item had a promotional offer, etc.), and $\sigma(\cdot)$ is the sigmoid function. β_x and γ_y are per-user and per-item offsets.

3.7 Empirical Evaluation

We first conduct semi-synthetic experiments to explore the empirical performance and robustness of the proposed methods in both the experimental and the observational setting. Then, we compare against the state-of-the-art joint likelihood method for MNAR data [40] on real-world datasets.

3.7.1 Experiment Setup

In all experiments, we perform model selection for the ℓ_2 regularization parameter λ and the rank of the factorization d via cross-validation as follows. We randomly split the observed MNAR ratings into k folds ($k = 4$ in all experiments), training on $k - 1$ and evaluating on the remaining one using the IPS estimator. Reflecting this additional split requires scaling the propensities in the training folds by $\frac{k-1}{k}$ and those in the validation fold by $\frac{1}{k}$. The hyper-parameters with the best validation set performance are then used to retrain the factorization model on all MNAR data. We report performance on the MCAR test set for real-world datasets or using Equation (3.1) for our semi-synthetic dataset.

3.7.2 How Does Sampling Bias Affect Evaluation?

First, we evaluate how different observation models impact the accuracy of performance estimates. We compare the *Naïve* estimator of Equation (3.5) for MSE, MAE and DCG with their propensity weighted analogs, *IPS* using Equation (3.10) and *SNIPS* using Equation (3.12) respectively. Since this experiment requires experimental control of sampling bias, we created a semi-synthetic dataset and observation model.

ML100K Dataset: The ML100K dataset[4] provides 100K MNAR ratings for 1683 movies by 944 users. To allow ground-truth evaluation against a fully known rating matrix, we complete these partial ratings using standard matrix factorization. The completed matrix, however, gives unrealistically high ratings

[4]http://grouplens.org/datasets/movielens/

to almost all movies. We, therefore, adjust ratings for the final Δ to match a more realistic rating distribution [79] $[p_1, p_2, p_3, p_4, p_5]$ for ratings 1 to 5 as follows: we assign the bottom p_1 fraction of the entries by value in the completed matrix a rating of 1, and the next p_2 fraction of entries by value a rating of 2, and so on. We chose hyper-parameters (rank d and ℓ_2 regularization λ) by using a 90-10 train-test split of the 100K ratings and picking the ones that maximized the accuracy of the completed matrix on the test set.

ML100K Observation Model: If the underlying rating is 4 or 5, the propensity for observing the rating is equal to k. For ratings $r < 4$, the corresponding propensity is $k\alpha^{4-r}$. For each α, we set k so that the expected number of ratings we observe is 5% of the entire matrix. By varying $\alpha > 0$, we vary the MNAR effect: $\alpha = 1$ is missing uniformly at random (MCAR), while $\alpha \to 0$ only reveals 4 and 5 rated items. Note that $\alpha = 0.25$ gives a marginal distribution of observed ratings that resembles the marginals on ML100K ($[0.06, 0.11, 0.27, 0.35, 0.21]$ in ML100K versus $[0.06, 0.10, 0.25, 0.42, 0.17]$ in our model).

Results: Table 3.1 and Table 3.2, described in Section 3.4.4, shows the estimated MAE and DCG@50, respectively, when $\alpha = 0.25$. Next, we vary the severity of the sampling bias by changing $\alpha \in (0, 1]$. Figure 3.2 reports how accurately (in terms of root mean squared estimation error (RMSE)) each estimator predicts the true MSE and DCG respectively. These results are for the Experimental Setting where propensities are known. They are averages over the five prediction matrices $\hat{\Delta}_i$ given in Section 3.4.4 and across 50 trials. Shaded regions indicate a 95% confidence interval.

Over most of the range of α, in particular for the realistic value of $\alpha = 0.25$,

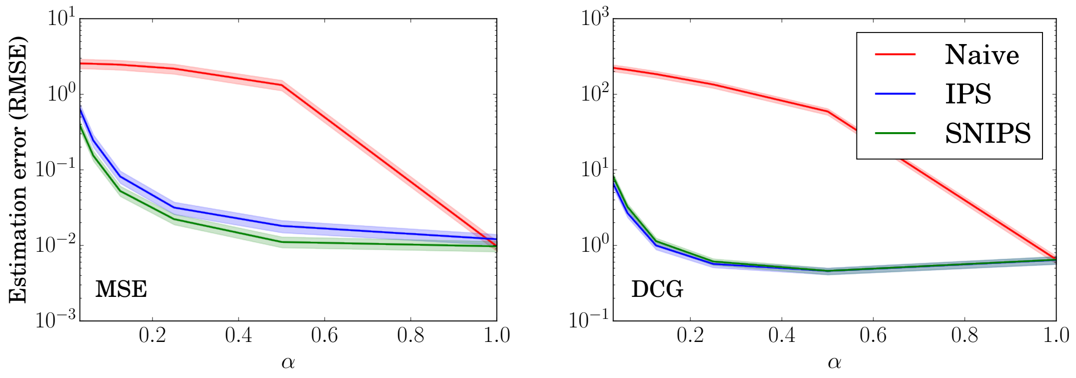


Figure 3.2: RMSE of rating prediction and recommendation quality estimators in the experimental setting as the observed ratings exhibit varying degrees of selection bias.

the *IPS* and *SNIPS* estimators are orders-of-magnitude more accurate than the *Naïve* estimator. Even for severely low choices of α (i.e. very severe selection biases), the gain due to bias reduction of *IPS* and *SNIPS* still outweighs the added variability compared to *Naïve*. When $\alpha = 1$ (MCAR), *SNIPS* is algebraically equivalent to *Naïve*, while *IPS* pays a small penalty due to increased variability from propensity weighting. For MSE, *SNIPS* consistently reduces estimation error over *IPS*, while both are tied for DCG.

3.7.3 How Does Sampling Bias Affect Learning?

Now we explore whether these gains in risk estimation accuracy translate into improved learning via ERM, again in the Experimental Setting. Using the same semi-synthetic ML100K dataset and observation model as in Section 3.7.2, we compare our matrix factorization *MF-IPS* with the traditional unweighted matrix factorization *MF-Naïve*. Both methods use the same factorization model with separate λ selected via cross-validation and $d = 20$. The results are plotted in Figure 3.3 (left), where shaded regions indicate 95% confidence intervals over 30 trials. The propensity weighted matrix factorization *MF-IPS* consistently outperforms standard matrix factorization in terms of MSE. We also conducted experiments for MAE, with similar results.

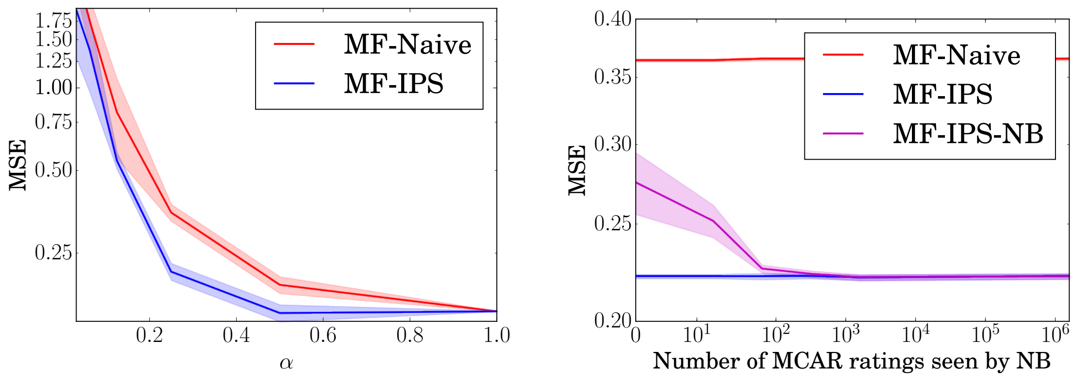


Figure 3.3: Prediction error (MSE) of matrix factorization methods as the observed ratings exhibit varying degrees of selection bias (left) and as propensity estimation quality degrades (right).

3.7.4 Do Inaccurate Propensities Destroy Reliable Evaluation and Learning?

We now switch from the Experimental Setting to the Observational Setting, where propensities need to be estimated. To explore robustness to propensity estimates of varying accuracy, we use the ML100K data and observation model with $\alpha = 0.25$. To generate increasingly bad propensity estimates, we use the Naive Bayes model from Section 3.6.1, but vary the size of the MCAR sample for estimating the marginal ratings $\Pr(\Delta = r)$ via the Laplace estimator,

$$\hat{\Pr}(\Delta = r) = \frac{1 + |(x, y) \in \text{MCAR} : \Delta_{x,y} = r|}{5 + |\text{MCAR}|}.$$

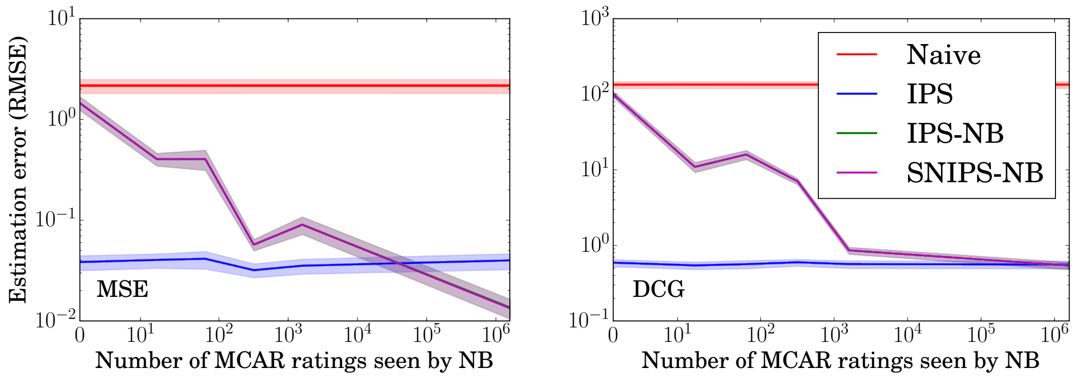


Figure 3.4: RMSE of *IPS* and *SNIPS* as propensity estimates degrade. *IPS* with true propensities and *Naïve* are plotted as a reference.

Figure 3.4 shows how the quality of the propensity estimates impacts evaluation using the same setup as in Section 3.7.2. Under no condition do the *IPS* and *SNIPS* estimator perform worse than *Naive*. Interestingly, *IPS-NB* with estimated propensities can perform even better than *IPS-KNOWN* with known propensities, as can be seen for MSE. This effect is known, partly because the estimated propensities can provide an effect akin to stratification [43, 131].

Figure 3.3 (right) shows how learning performance is affected by inaccurate propensities using the same setup as in Section 3.7.3. We compare the MSE prediction error of *MF-IPS-NB* with estimated propensities to that of *MF-Naïve* and *MF-IPS* with known propensities. The shaded area shows the 95% confidence interval over 30 trials. Again, we see that *MF-IPS-NB* outperforms *MF-Naïve* even for severely degraded propensity estimates, demonstrating the robustness of the approach.

3.7.5 Performance on Real-World Data

Our final experiment studies performance on real-world datasets. We use the following two datasets, which both have a separate test set where users were asked to rate a uniformly drawn sample of items.

Yahoo! R3 Dataset: This dataset[5] [79] contains ratings of songs by users. The MNAR training set provides over 300K ratings for songs that were self-selected by 15400 users. The test set contains ratings by a subset of 5400 users who were asked to rate 10 randomly chosen songs. For this data, we estimate propensities via Naïve Bayes. As an MCAR sample for eliciting the marginal rating distribution, we set aside 5% of the test set and report results only on the remaining 95% test set.

Coat Shopping Dataset: We collected a new dataset[6] simulating MNAR data of customers shopping for a coat in an online store. The training data was generated by giving Amazon Mechanical Turkers a simple web shop interface with facets and paging. They were asked to find the coat in the store that they wanted to buy the most. Afterward, they had to rate 24 of the coats they explored (self-selected) and 16 randomly picked ones on a five-point scale. The dataset contains ratings from 290 Turkers on an inventory of 300 items. The self-selected ratings are the training set, and the uniformly selected ratings are the test set. We learn propensities via logistic regression based on user covariates (gender, age group, location, and fashion-awareness) and item covariates (gender, coat type, color, and "was it promoted by the interface?"). A standard

[5]http://webscope.sandbox.yahoo.com/

[6]https://www.cs.cornell.edu/~schnabts/mnar/

regularized logistic regression implemented using scikit-learn [87] was trained using all pairs of user and item covariates as features and cross-validated to optimize log-likelihood of the self-selected observations.

	YAHOO		COAT	
	MAE	MSE	MAE	MSE
MF-IPS	**0.810**	**0.989**	**0.860**	**1.093**
MF-Naïve	1.154	1.891	0.920	1.202
HL MNAR	1.177	2.175	0.884	1.214
HL MAR	1.179	2.166	0.892	1.220

Table 3.3: Test set MAE and MSE for rating prediction models trained using our proposed approach (*MF-IPS*), conventional matrix factorization (*MF-Naïve*) and the state-of-the-art generative models with two different priors (HL [40]) on the Yahoo and Coat datasets.

Results: Table 3.3 shows that our propensity-scored matrix factorization *MF-IPS* with learned propensities substantially and significantly outperforms the conventional matrix factorization approach, as well as the Bayesian imputation models [40], abbreviated as HL-MNAR and HL-MAR (paired t-test, $p < 0.001$ for all comparisons). This conclusion holds for both MAE and MSE. Furthermore, the performance of *MF-IPS* beats the best published results for the Yahoo dataset in terms of MSE (1.115) and is close in terms of MAE (0.770) (the CTP-v model [79] has better performance [40]). For *MF-IPS* and *MF-Naïve* all hyper-parameters (i.e., $\lambda \in \{10^{-6}, \ldots 1\}$ and $d \in \{5, 10, 20, 40\}$) were chosen by cross-validation. For the HL baselines, we explored $d \in \{5, 10, 20, 40\}$ using software provided by the authors[7] and report the best performance on the test set (running a cross-validation sweep was computationally prohibitive). Our metrics for HL on the Yahoo dataset closely match their reported performance [40].

[7]https://bitbucket.org/jmh233/missingdataicml2014

Compared to the complex generative HL models, we conclude that our discriminative *MF-IPS* performs robustly and efficiently on real-world data. We conjecture that this strength is a result of not requiring any generative assumptions about the validity of the rating model. Furthermore, note that there are several promising directions for further improving performance, like propensity clipping [110], doubly-robust estimation [30, 31], and the use of improved methods for propensity estimation [83].

3.8 Conclusions and Future Work

We proposed an efficient and robust approach to handling selection bias in the evaluation and training of recommender systems based on propensity scoring. The approach is a discriminative alternative to existing joint-likelihood methods which are generative. It, therefore, inherits many of the advantages (e.g., efficiency, predictive performance, no need for latent variables, fewer modeling assumptions) of discriminative methods. The modularity of the approach — separating the estimation of the assignment model from the rating model — also makes it very practical. In particular, any conditional probability estimation method can be plugged in as the propensity estimator, and we conjecture that many existing rating models can be retrofit with propensity weighting without sacrificing scalability.

CHAPTER 4

LEARNING TO RANK USING IMPLICIT FEEDBACK FROM USERS

4.1 Chapter Notes

This chapter describes joint work with Thorsten Joachims and Tobias Schnabel. It is a lightly edited version of a conference publication [57].

We develop a potential outcomes model to understand users' behaviors on search rankings. The aim is to exploit randomness in user actions as a cheap alternative to randomizing rankings for off-policy learning as done in Chapter 5. In the $x \mapsto y \mapsto \delta$ schematic of Section 2.1, $\boldsymbol{y}$ is now a ranking of search results, and we do not have interventional data — this is the observational setting for causal estimation. δ is a known loss function that decomposes into per-document values (e.g., rank-discounted click positions). However, we cannot intervene to collect per-document values (e.g., we cannot force a user to examine a result and observe whether they decide to click or skip it) nor do we know the mechanisms that shape the collected data (e.g., "Why did Alice click a particular result?"). We again treat this as a missing data problem: no-click situations for results in a ranking are confounded by the fact that users are less likely even to examine low-ranked results. We re-use standard click models (see survey [23]) as propensity estimators and show how to incorporate them in traditional Learning-to-Rank algorithms [52]. Empirically we show substantially improved ranking performance in both the real world (on the `https://arxiv.org/` search engine) and semi-synthetic experiments.

4.2 Introduction

Batch training of retrieval systems requires annotated test collections that take substantial effort and cost to amass. While economically feasible for Web Search, eliciting relevance annotations from experts is infeasible or impossible for most other ranking applications (e.g., personal collection search, intranet search). For these applications, implicit feedback from user behavior is an attractive source of data. Unfortunately, existing approaches for Learning-to-Rank (LTR) from implicit feedback — and clicks on search results in particular — have several limitations or drawbacks.

First, the naïve approach of treating a click|no-click as a positive|negative relevance judgment is severely biased. In particular, the order of presentation has a strong influence on where users click [56]. This presentation bias leads to an incomplete and skewed sample of relevance judgments that is far from uniform, thus leading to biased data for learning-to-rank algorithms.

Second, treating clicks as preferences between clicked and skipped documents has been found to be accurate [52, 56], but it can only infer preferences that oppose the presented order. This approach again leads to severely biased data, and learning algorithms trained with these preferences tend to reverse the presented order unless additional heuristics are used [52].

Third, probabilistic click models have been used to model how users produce clicks (a recent survey [23] summarizes these models), and they can take position and context biases into account. By estimating latent parameters of these generative click models, one can infer the relevance of a given document for a given query. However, inferring reliable relevance judgments typically requires that the same query is seen multiple times, which is unrealistic in many retrieval settings (e.g., personal collection search) and for tail queries.

Fourth, allowing the LTR algorithm to randomize what is presented to the user, like in online learning algorithms [92, 45] and batch learning from bandit feedback (BLBF) [113] can overcome the problem of bias in click data in a principled manner. However, requiring that rankings be actively perturbed during system operation, whenever we collect training data, decreases ranking quality and, therefore, incurs a cost compared to observational data collection.

In this chapter, we present a theoretically principled and empirically effective approach for learning from implicit observational feedback that can overcome the limitations outlined above. By drawing on counterfactual estimation techniques from causal inference [50], we first develop an unbiased estimator for evaluating ranking performance using biased feedback data. Based on this estimator, we propose a propensity weighted Empirical Risk Minimization (ERM) approach to LTR, which we implement efficiently in a new learning method we call Propensity SVM-Rank. While our approach uses a click model, the click model is merely used to assign propensities to clicked results in hindsight, not to extract aggregate relevance judgments. Hence, our Propensity SVM-Rank does not require queries to repeat, making it applicable to a large range of ranking scenarios. Finally, our methods can use observational data, and we do not require that the system randomizes rankings during data collection, except optionally for a small pilot experiment to efficiently estimate the propensity model.

When developing our approach, we provide theoretical justification for each step, leading to a rigorous end-to-end approach that does not make unspecified assumptions or employs heuristics. This development provides a principled basis for further improving components of the approach (e.g., the click propensity model, the ranking performance measure, the learning algorithm). We present an extensive empirical evaluation testing the limits of the approach on synthetic click data, finding that it performs robustly over a large range of bias, noise and misspecification levels. Furthermore, we field our method in a real-world application on an operational search engine, finding that it is robust in practice and manages to improve retrieval performance substantially.

4.3 Related Work

There are two groups of approaches for handling biases in implicit feedback for learning-to-rank. The first group assumes the feedback collection step is fixed and tries to interpret the observational data so as to minimize bias effects. Approaches in the second group intervene during feedback collection, trying to present rankings that will lead to less biased feedback data overall.

Approaches in the first group commonly assume some model of user behavior to explain away bias effects. For example, in a cascade model [27], users are assumed to sequentially go down a ranking and click on a document if it is relevant. Clicks, under this model, let us learn preferences between skipped and clicked documents. Learning from these relative preferences lowers the impact of some biases [52]. Other click models [27, 21, 11, 23] have been proposed, and are trained to maximize log-likelihood of observed clicks. In these click modeling approaches, performance on downstream learning-to-rank algorithms is merely an afterthought. In contrast, we separate click propensity estimation and learning-to-rank in a principled way, and we optimize for ranking performance directly. Our framework allows us to plug-in more sophisticated user models in place of the simple click models we use in this work.

The key technique used by approaches in the second group to obtain more reliable click data are randomized experiments. For instance, randomizing documents across all ranks lets us learn unbiased relevances for each document, and swapping adjacent pairs of documents [92] lets us learn pairwise preferences reliably. Similarly, randomized interleaving can detect preferences between different rankers reliably [19]. Different from online learning via bandit algorithms and interleaving [135, 103], batch learning from bandit feedback (BLBF) [113] still uses randomization during feedback collection, and then performs offline learning (we will study this in detail in Chapter 6). We can interpret our problem formulation as being half way between the BLBF setting and learning-to-rank from editorial judgments. In BLBF settings, the loss function is unknown, and we make no assumptions about the loss function. When learning from editorial judgments, components of a search ranking are fully labeled, and the loss function is given. In the current setting, we know the form of the loss function, but labels for only some parts of the ranking are revealed. All approaches that use randomization suffer from two limitations. First, randomization typically degrades ranking quality during data collection; second, deploying non-deterministic ranking functions introduces bookkeeping overhead. In this chapter, the system can be deterministic and we merely exploit and model stochasticity in user behavior. Moreover, our framework allows (but does not require) the use of randomized data collection to mitigate biases and to lower estimator variance.

Our approach uses inverse propensity scoring (IPS), originally employed in survey sampling [47] and causal inference from observational studies [95], but more recently also in whole page optimization [129] and recommender evaluation [72, 101]. We use randomized interventions to estimate propensities in a position discount model. This intervention scheme has precedent — rather than using uniform ranking randomization [128] (with its high performance impact) or swapping adjacent pairs [27], we swap documents in different ranks to the top position [65]. See Section 4.6.3 for details.

Finally, our approach is similar in spirit to recent work [128] where propensity weighting is used to correct for selection bias when discarding queries without clicks during learning-to-rank. The key insight of our work is to recognize that inverse propensity scoring can be employed much more powerfully, to account for position bias, trust bias, contextual effects, document popularity etc. using appropriate click models to estimate the propensity of each click rather than the propensity for a query to receive a click [128].

4.4 Full-Information Learning to Rank

Before we derive our approach for LTR from biased implicit feedback, we first review the conventional problem of LTR from editorial judgments. In conventional LTR, we are given a sample $\mathcal{X}$ of i.i.d. queries $x_i \sim \Pr(x)$ for which we assume the relevances $rel(x, y)$ of all documents y are known. Since all relevances are assumed to be known, we call this the Full-Information Setting. The relevances can be used to compute the *loss* $\delta(\boldsymbol{y} \mid x)$ (e.g., negative DCG) of any ranking $\boldsymbol{y}$ for query x. Aggregating the losses of individual rankings by taking the expectation over the query distribution, we can define the overall *risk* of a ranking system S that returns rankings $S(x)$ as

$$R(S) = \int \delta(S(x) \mid x) d \Pr(x). \tag{4.1}$$

The goal of learning is to find a ranking function $S \in \mathcal{S}$ that minimizes $R(S)$ for the query distribution $\Pr(x)$. Since $R(S)$ cannot be computed directly, it is typically estimated via the *empirical risk*

$$\hat{R}(S) = \frac{1}{|\mathcal{X}|} \sum_{x_i \in \mathcal{X}} \delta(S(x_i) \mid x_i).$$

A common learning strategy is *Empirical Risk Minimization (ERM)* [124], which corresponds to picking the system $\hat{S} \in \mathcal{S}$ that optimizes the empirical risk

$$\hat{S} = \operatorname*{argmin}_{S \in \mathcal{S}} \left\{ \hat{R}(S) \right\},$$

possibly subject to some regularization to control overfitting. There are several LTR algorithms that follow this approach (a recent survey [78] summarizes these approaches), and we use SVM-Rank [52] as a representative algorithm in this chapter.

The relevances $rel(x, y)$ are typically elicited via expert judgments. Apart from being expensive and often infeasible (e.g., in personal collection search), expert judgments come with at least two other limitations. First, since it is clearly impossible to get explicit judgments for all documents, pooling techniques [106] are used such that only the most promising documents are judged. While cutting down on judging effort, this introduces an undesired *pooling bias* because all unjudged documents are typically assumed to be irrelevant. The second limitation is that expert judgments $rel(x, y)$ have to be aggregated over all intents that underlie the same query string, and it can be challenging for a judge to conjecture a distribution of intents to assign an appropriate $rel(x, y)$.

4.5 Partial-Information Learning to Rank

Learning from implicit feedback has the potential to overcome the limitations of full-information LTR mentioned above. By drawing the training signal directly from the user, it naturally reflects the user's intent, since each user acts upon their relevance judgment subject to their specific context and information need. It is, therefore, more appropriate to talk about query instances x_i that include contextual information about the user, instead of query strings x. For a given query instance x_i, we denote with $r_i(y)$ the user-specific relevance of result y for query instance x_i. One may argue that what expert assessors try to capture with $rel(x, y)$ is the mean of the relevances $r_i(y)$ over all the query instances that share the query string, so, using implicit feedback for learning can remove a lot of guesswork about what the distribution of users meant by a query.

However, when using implicit feedback as a relevance signal, unobserved feedback is an even greater problem than missing judgments in the pooling setting. In particular, implicit feedback is distorted by presentation bias, and it is not missing completely at random [77]. To nevertheless derive well-founded learning algorithms, we adopt the following counterfactual model.

For concreteness and simplicity, assume that relevances are binary, $r_i(y) \in \{0, 1\}$, and our performance measure of interest is the sum of the ranks of the relevant results

$$\delta(\boldsymbol{y} \mid x_i, r_i) \quad = \quad \sum_{y \in \boldsymbol{y}} rank(y \mid \boldsymbol{y}) \cdot r_i(y). \tag{4.2}$$

Analogous to Equation (4.1), we can define the risk of a system as

$$R(S) \quad = \quad \int \delta(S(x) \mid x, r) d \Pr(x, r). \tag{4.3}$$

In our counterfactual model, there exists a true vector of relevances r_i for each incoming query instance $(x_i, r_i) \sim \Pr(x, r)$. However, only a part of these relevances is observed for each query instance, while typically most remain unobserved. In particular, given a presented ranking $\bar{\boldsymbol{y}}_i$ we are more likely to observe the relevance signals (e.g., clicks) for the top-ranked results than for results ranked lower down the list. Let o_i denote the binary vector indicating which relevance values were revealed, $o_i \sim \Pr(o \mid x_i, \bar{\boldsymbol{y}}_i, r_i)$. For each element of o_i, denote with $Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)$ the marginal probability of observing the relevance $r_i(y)$ of result y for query x_i, if the user was presented the ranking $\bar{\boldsymbol{y}}_i$. We refer to this probability value as the *propensity* of the observation. We will discuss how o_i and Q can be obtained in Section 4.6.

Using this counterfactual modeling setup, we can get an unbiased estimate of $\delta(\boldsymbol{y} \mid x_i, r_i)$ for any new ranking $\boldsymbol{y}$ (typically different from the presented ranking $\bar{\boldsymbol{y}}_i$) via the inverse propensity scoring (IPS) estimator [47, 95, 50]

$$\begin{aligned}
\hat{\delta}_{IPS}(\boldsymbol{y} \mid x_i, r_i, o_i) &= \sum_{y:o_i(y)=1} \frac{rank(y \mid \boldsymbol{y}) \cdot r_i(y)}{Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)} \\
&= \sum_{\substack{y:o_i(y)=1 \\ \bigwedge r_i(y)=1}} \frac{rank(y \mid \boldsymbol{y})}{Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)}.
\end{aligned}$$

This is an unbiased estimate of $\delta(\boldsymbol{y} \mid x_i, r_i)$ for any $\boldsymbol{y}$, if $Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i) > 0$ for all y that are relevant $r_i(y) = 1$ (but not necessarily for the irrelevant y).

$$\begin{aligned}
\mathbb{E}_{o_i}\left[\hat{\delta}_{IPS}(\boldsymbol{y} \mid x_i, \bar{\boldsymbol{y}}_i, o_i)\right] &= \mathbb{E}_{o_i}\left[\sum_{y:o_i(y)=1} \frac{rank(y \mid \boldsymbol{y}) \cdot r_i(y)}{Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)}\right] \\
&= \sum_{y \in \boldsymbol{y}} \mathbb{E}_{o_i}\left[\frac{o_i(y) \cdot rank(y \mid \boldsymbol{y}) \cdot r_i(y)}{Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)}\right] \qquad (4.4) \\
&= \sum_{y \in \boldsymbol{y}} \frac{Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i) \cdot rank(y \mid \boldsymbol{y}) \cdot r_i(y)}{Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)} \\
&= \sum_{y \in \boldsymbol{y}} rank(y \mid \boldsymbol{y}) r_i(y) \qquad (4.5) \\
&= \delta(\boldsymbol{y} \mid x_i, r_i).
\end{aligned}$$

Equation (4.4) comes from linearity of expectation, and Equation (4.5) uses $Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i) > 0$.

An interesting property of $\hat{\delta}_{IPS}(\boldsymbol{y} \mid x_i, \bar{\boldsymbol{y}}_i, o_i)$ is that only those results y with $[o_i(y) = 1 \wedge r_i(y) = 1]$ (i.e. clicked results, as we will see later) contribute to the estimate. We therefore only need the propensities $Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)$ for relevant results. Since we will eventually need to estimate the propensities $Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)$, an additional requirement for making $\hat{\delta}_{IPS}(\boldsymbol{y} \mid x_i, \bar{\boldsymbol{y}}_i, o_i)$ computable while remaining unbiased is that the propensities only depend on observable information (i.e., unconfoundedness [50]).

To define the empirical risk to optimize during learning, we begin by collecting a sample of n query instances x_i, recording the partially-revealed relevances r_i as indicated by o_i, and the propensities $Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)$ for the observed relevant results in the ranking $\bar{\boldsymbol{y}}_i$ presented by the system. Then, the empirical risk of a system is simply the IPS estimates averaged over query instances:

$$\hat{R}_{IPS}(S) = \frac{1}{n}\sum_{i=1}^{n} \sum_{\substack{y:o_i(y)=1 \\ \wedge\, r_i(y)=1}} \frac{rank(y \mid S(x_i))}{Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)}. \tag{4.6}$$

Since $\hat{\delta}_{IPS}(\boldsymbol{y} \mid x_i, \bar{\boldsymbol{y}}_i, o_i)$ is unbiased for each query instance, the aggregate $\hat{R}_{IPS}(S)$ is also unbiased for $R(S)$ from Equation (4.3),

$$\mathbb{E}[\hat{R}_{IPS}(S)] = R(S).$$

Furthermore, it is easy to verify that $\hat{R}_{IPS}(S)$ converges to the true $R(S)$ under mild additional conditions (i.e., propensities bounded away from 0) as we increase the sample size n of query instances. To see this, observe that $\hat{R}_{IPS}(S)$ is the mean of bounded random variables and the central limit theorem applies. So, we can perform ERM using this propensity weighted empirical risk,

$$\hat{S} = \operatorname*{argmin}_{S \in \mathcal{S}} \left\{ \hat{R}_{IPS}(S) \right\}.$$

Finally, using standard results from statistical learning theory [124], asymptotic consistency of the empirical risk paired with capacity control implies consistency also for ERM. In intuitive terms, this means that given enough training data, the learning algorithm is guaranteed to find the best system in $\mathcal{S}$.

4.6 Feedback Propensity Models

In Section 4.5, we showed that the relevance signal r_i, the observation pattern o_i, and the propensities of the observations $Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)$ are the key components for unbiased LTR from biased observational feedback. We now outline how these quantities can be elicited and modeled in a typical search-engine application. However, the general framework of Section 4.5 extends beyond this particular application, and beyond the particular feedback model below.

4.6.1 Position-Based Propensity Model

Search engine click logs provide a sample of query instances x_i, the presented ranking $\bar{\boldsymbol{y}}_i$ and a (sparse) click-vector where each $c_i(y) \in \{0, 1\}$ indicates whether result y was clicked or not. To derive propensities of observed clicks, we will employ a click propensity model. For simplicity, we consider a straightforward examination model [93], where a click on a search result depends on the probability that a user examines a result (i.e., $e_i(y)$) and then decides to click on it (i.e., $c_i(y)$) in the following way:

$$\Pr(e_i(y) = 1 \mid rank(y \mid \bar{\boldsymbol{y}}_i)) \cdot \Pr(c_i(y) = 1 \mid r_i(y), e_i(y) = 1).$$

In this model, examination depends only on the rank of y in $\bar{\boldsymbol{y}}_i$. So, $\Pr(e_i(y) = 1 \mid rank(y \mid \bar{\boldsymbol{y}}_i))$ can be represented by a vector of examination probabilities p_r, one for each rank r. These examination probabilities can model presentation bias documented in eye-tracking studies [56], where users are more likely to see results at the top of the ranking than those further down.

For the probability of click on an examined result $\Pr(c_i(y) = 1 \mid r_i(y), e_i(y) = 1)$, we first consider the simplest model where clicking is a deterministic noise-free function of the users private relevance assessment $r_i(y)$. Under this model, users click if and only if the result is examined and relevant ($c_i(y) = 1 \leftrightarrow [e_i(y) = 1 \wedge r_i(y) = 1]$). This means that for examined results (i.e., $e_i(y) = 1$), clicking is synonymous with relevance ($e_i(y) = 1 \rightarrow [c_i(y) = r_i(y)]$). Furthermore, it means that we observe the value of $r_i(y)$ perfectly when $e_i(y) = 1$ ($e_i(y) = 1 \rightarrow o_i(y) = 1$). Moreover, we gain no knowledge of the true $r_i(y)$ when a result is not examined ($e_i(y) = 0 \rightarrow o_i(y) = 0$). Therefore, examination equals observation and $Q(o_i(y) \mid x_i, \bar{\boldsymbol{y}}_i, r_i) \equiv \Pr(e_i(y) \mid rank(y \mid \bar{\boldsymbol{y}}_i))$.

Using these equivalences, we can simplify the IPS estimator from Equation (4.6) by substituting p_r as the propensities and by using $c_i(y) = 1 \leftrightarrow [o_i(y) = 1 \wedge r_i(y) = 1]$

$$\hat{R}_{IPS}(S) = \frac{1}{n}\sum_{i=1}^{n}\sum_{y:c_i(y)=1}\frac{rank(y \mid S(x_i))}{p_{rank(y|\bar{\boldsymbol{y}}_i)}}. \tag{4.7}$$

$\hat{R}_{IPS}(S)$ is an unbiased estimate of $R(S)$ under the position-based propensity model if $p_r > 0$ for all ranks. While absence of a click does not imply that the result is not relevant (i.e., $c_i(y) = 0 \nrightarrow r_i(y) = 0$), the IPS estimator has the nice property that such explicit negative judgments are not needed to compute an unbiased estimate of $R(S)$ for the loss in Equation (4.2). Similarly, while absence of a click leaves us unsure about whether the result was examined (i.e., $e_i(y) = ?$), the IPS estimator only needs to know the indicators $o_i(y) = 1$ for results that are also relevant (i.e., clicked results).

Finally, note the conceptual difference in how we use this standard examination model compared to most prior work. We do not try to estimate an average relevance rating $rel(x, y)$ by taking repeat instances of the same query x, but we use the model as a propensity estimator to de-bias individual observed user judgments $r_i(y)$ to be used directly in ERM.

4.6.2 Incorporating Click Noise

In Section 4.6.1, we assumed that clicks reveal the user's true r_i in a noise-free way. This assumption is clearly unrealistic. In addition to the stochasticity in the examination distribution $\Pr(e_i(y) = 1 \mid rank(y \mid \bar{\mathbf{y}}_i))$, we now also consider noise in the distribution that generates the clicks. In particular, we no longer require that a relevant result is clicked with probability 1, and an irrelevant result is clicked with probability 0, but instead, for $1 \geq \epsilon_+ > \epsilon_- \geq 0$,

$$\Pr(c_i(y) = 1 \mid r_i(y) = 1, o_i(y) = 1) = \epsilon_+,$$
$$\Pr(c_i(y) = 1 \mid r_i(y) = 0, o_i(y) = 1) = \epsilon_-.$$

The first line means that users click on a relevant result only with probability ϵ_+, while the second line means that users may erroneously click on an irrelevant result with probability ϵ_-. An alternative and equivalent way of thinking about click noise is that users still click deterministically as in the previous section, but based on a corrupted version $\tilde{r}_i$ of r_i. This viewpoint tells us that all the reasoning regarding observation (examination) events o_i and their propensities p_r still holds and that we still have that $c_i(y) = 1 \rightarrow o_i(y) = 1$. What does change, though, is that we no longer observe the "correct" $r_i(y)$ but instead get feedback according to the noise-corrupted version $\tilde{r}_i(y)$. What happens to our learning process if we estimate risk using Equation (4.7), but now with $\tilde{r}_i$?

Fortunately, the noise does not affect ERM's ability to find the best ranking system given enough data. While using noisy clicks leads to biased empirical risk estimates for the true r_i (i.e., $\mathbb{E}[\hat{R}_{IPS}(S)] \neq R(S)$), in expectation this bias is order preserving for $R(S)$ such that the risk minimizer remains the same.

$$\begin{aligned}
& & & \mathbb{E}[\hat{R}_{IPS}(S_1)] > \mathbb{E}[\hat{R}_{IPS}(S_2)] \\
& & \Leftrightarrow\quad & \mathbb{E}_{x,r,\bar{y}}\left[\mathbb{E}_o \mathbb{E}_{c|o}\left[\sum_{y:c(y)=1} \frac{rank(y \mid S_1(x)) - rank(y \mid S_2(x))}{p_{rank(y|\bar{y})}}\right]\right] > 0 \\
& & \Leftrightarrow\quad & \mathbb{E}_{x,r}\left[\sum_y \Pr(c(y)=1 \mid o(y)=1, r(y)) \Delta\mathrm{rank}(y \mid x)\right] > 0 \\
& & \Leftrightarrow\quad & \mathbb{E}_{x,r}\left[\sum_y \Delta\mathrm{rank}(y \mid x) \cdot (\epsilon_+ r(y) + \epsilon_-(1 - r(y)))\right] > 0 \\
& & \Leftrightarrow\quad & \mathbb{E}_{x,r}\left[\sum_y \Delta\mathrm{rank}(y \mid x) \cdot ((\epsilon_+ - \epsilon_-) r(y) + \epsilon_-)\right] > 0 \\
& * & \Leftrightarrow\quad & \mathbb{E}_{x,r}\left[\sum_y \Delta\mathrm{rank}(y \mid x) \cdot (\epsilon_+ - \epsilon_-) r(y)\right] > 0 \\
& & \Leftrightarrow\quad & \mathbb{E}_{x,r}\left[\sum_y \Delta\mathrm{rank}(y \mid x) \cdot r(y)\right] > 0 \\
& & \Leftrightarrow\quad & R(S_1) > R(S_2),
\end{aligned}$$

where $\Delta\mathrm{rank}(y \mid x)$ is short for $rank(y \mid S_1(x)) - rank(y \mid S_2(x))$ and we use the fact that $\epsilon_- \sum_{y \in \bar{y}} \Delta\mathrm{rank}(y \mid x) = 0$ in the step marked $*$. This implies that our propensity weighted ERM is a consistent approach for finding a ranking function with the best true $R(S)$,

$$\begin{aligned}
\hat{S} &= \underset{S \in \mathcal{S}}{\operatorname{argmin}} \{R(S)\} \\
&= \underset{S \in \mathcal{S}}{\operatorname{argmin}} \left\{\mathbb{E}[\hat{R}_{IPS}(S)]\right\}, \qquad (4.8)
\end{aligned}$$

even when the objective is corrupted by click noise as specified above.

4.6.3 Propensity Estimation

As the last step of defining the click propensity model, we need to address the question of how to estimate its parameters (i.e. the vector of examination probabilities p_r) for a particular search engine. The following shows that we can get estimates using data from a simple intervention [65] but without the strong negative impact of presenting uniformly random results to some users [128].

First, note that it suffices to estimate the p_r up to some positive multiplicative constant, since any such constant does not change how the IPS estimator of Equation (4.7) orders different systems. We therefore merely need to estimate how much p_r changes relative to p_k for some "landmark" rank k. This suggests the following experimental intervention for estimating p_r: before presenting the ranking to the user, swap the result at rank k with the result at rank r. If we denote with y' the results originally in rank k, our click model before and after the intervention indicates that

$$\begin{aligned}
\Pr(c_i(y') = 1 \mid \text{no-swap}) &= p_k \cdot \Pr(c_i(y') = 1 \mid e_i(y') = 1) \\
\Pr(c_i(y') = 1 \mid \text{swap-k-and-r}) &= p_r \cdot \Pr(c_i(y') = 1 \mid e_i(y') = 1)
\end{aligned}$$

where

$$\begin{aligned}
&\Pr(c_i(y') = 1 \mid e_i(y') = 1) \\
&= \sum_{v \in \{0,1\}} \Pr(c_i(y') = 1 \mid r_i(y') = v, e_i(y') = 1) \cdot \Pr(r_i(y') = v)
\end{aligned}$$

is constant regardless of the intervention. This observation means that the click-through rates $\Pr(c_i(y') = 1 \mid \text{swap-k-and-r})$, which we can estimate from the intervention data, are proportional to the parameters p_r for any r. By performing swaps between rank k and other ranks r, we can estimate all the p_r parameters.

This swap-intervention experiment has a much lower adverse impact on system performance than uniform randomization, which was proposed for a different propensity estimation problem [128], and careful consideration of which rank k to choose can further reduce the impact of the swap experiment. From a practical perspective, it may also be unnecessary to estimate p_r for each rank separately. Instead, one may want to interpolate between estimates at well-chosen ranks or employ smoothing. Finally, note that the intervention only needs to be applied to a small subset of the data used for fitting the click propensity model, while the actual data used for training the ERM learning algorithm does not require any interventions.

4.6.4 Alternative Feedback Propensity Models

The click propensity model we defined above is arguably one of the simplest models one can employ for propensity modeling in LTR, and there is broad scope for extensions.

First, one could extend the model by incorporating other biases, for example, trust bias [56] which affects perceived relevance of a result based on its position in the ranking. This can be captured by conditioning click probabilities also on the position $\Pr(c_i(y') = 1 \mid r_i(y'), e_i(y') = 1, rank(y \mid \bar{\mathbf{y}}_i))$. We have already explored that the model can be extended to include trust bias, and the intervention proposed in Section 4.6.3 continues to yield good propensity estimates for this extension. Furthermore, it is possible to model saliency biases [136] by replacing the p_r with a regression function.

Second, we conjecture that a wide range of other click models (e.g., cascade model [27] and others [27, 21, 11, 23]) can be adapted as propensity models. The main requirement is that we can compute marginal click probabilities for the clicked documents in hindsight, which may be feasible for many other models.

Third, we may be able to define and train new types of click models. In particular, for our propensity ERM approach we only need the propensities $Q(o_i(y) = 1 \mid x_i, \bar{\mathbf{y}}_i, r_i)$ for observed and relevant documents to evaluate the IPS estimator, but not for irrelevant documents. This can be substantially easier than a full generative model of how people reveal relevance judgments through implicit feedback. In particular, this model can condition on all the revealed relevances $r_i(y_j)$ in hindsight, and it does not need to treat them as latent variables.

Finally, the ERM learning approach is not limited to binary click feedback, but applies to a large range of feedback settings. For example, the feedback may be explicit star ratings in a movie recommendation system, and the propensities may be the results of self-selection by the users, as we saw in Chapter 3. In such an explicit feedback setting, o_i is fully known, which simplifies propensity estimation substantially.

4.7 Propensity Weighted SVM-Rank

We now derive a concrete learning method that implements the propensity weighted LTR principle from Section 4.5. It is based on SVM-Rank [52, 55], but we conjecture that propensity weighted versions of other LTR methods can be derived as well.

Consider a dataset of n examples of the following form. For each query-(clicked)result pair (x_j, y_j) that is clicked, we compute the propensity $q_i = Q(o_i(y) = 1 \mid x_i, \bar{\mathbf{y}}_i, r_i)$ of the click according to our click propensity model. We also record the candidate set Y_j of all results for query x_j. Typically, Y_j contains a few hundred documents — selected by a stage-one ranker [127] — that we aim to rerank. Note that each click generates a separate training example, even if multiple clicks occur for the same query.

Given this propensity-scored click data, we define Propensity SVM-Rank as a generalization of conventional SVM-Rank. Propensity SVM-Rank learns a linear scoring function $pred(x, y) = \mathbf{w}^T \mathbf{f}(x, y)$ that can be used for ranking results, where $\mathbf{w}$ is a weight vector, and $\mathbf{f}(x, y)$ is a feature vector that describes the match between query x and result y.

Propensity SVM-Rank optimizes the following objective,

$$\begin{aligned}
\hat{\mathbf{w}} \quad &= \quad \underset{\mathbf{w},\xi}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{j=1}^{n}\frac{1}{q_j}\sum_{y \in Y_j}\xi_{jy} \\
s.t. \quad & \quad \forall y \in Y_1 \setminus \{y_1\} : \mathbf{w}^T[\mathbf{f}(x_1, y_1) - \mathbf{f}(x_1, y)] \geq 1 - \xi_{1y} \\
& \quad \vdots \\
& \quad \forall y \in Y_n \setminus \{y_n\} : \mathbf{w}^T[\mathbf{f}(x_n, y_n) - \mathbf{f}(x_n, y)] \geq 1 - \xi_{ny} \\
& \quad \forall j \forall y : \xi_{jy} \geq 0.
\end{aligned}$$

C is a regularization parameter that is typically selected via cross-validation. The training objective optimizes an upper bound on the regularized IPS estimated empirical risk of Equation (4.7) since each line of constraints corresponds to the rank of a relevant document (minus 1). In particular, for any feasible $(\mathbf{w}, \xi)$

$$\begin{aligned} rank(y_i \mid \boldsymbol{y}) - 1 &= \sum_{y \neq y_i} \mathbf{1}\{\mathbf{w}^T[\mathbf{f}(x_i, y) - \mathbf{f}(x_i, y_i)] > 0\} \\ &\leq \sum_{y \neq y_i} \max(1 - \mathbf{w}^T[\mathbf{f}(x_i, y_i) - \mathbf{f}(x_i, y)], 0) \\ &\leq \sum_{y \neq y_i} \xi_{iy}. \end{aligned}$$

We can solve this type of Quadratic Program efficiently via a one-slack formulation [55], and we are using a modified SVM-Rank that incorporates IPS weights $1/q_j$. The latest version of SVM-Rank[1] includes these modifications.

In the empirical evaluation, we compare against the naïve application of SVM-Rank, which minimizes the rank of the clicked documents while ignoring presentation bias. In particular, Naïve SVM-Rank sets all the q_i uniformly to the same constant (e.g., 1).

4.8 Empirical Evaluation

We take a two-pronged approach to evaluating our approach empirically. First, we use synthetically generated click data to explore the behavior of our methods over the whole spectrum of presentation bias severity, click noise, and propensity misspecification. Second, we explore the real-world applicability of our approach by evaluating on an operational search engine using real click-logs from live traffic.

[1]http://www.joachims.org/svm_light/svm_rank.html

4.8.1 Synthetic Data Experiments

To be able to explore the full spectrum of biases and noise, we conducted experiments using click data derived from the Yahoo Learning to Rank Challenge corpus (set 1) [18]. This corpus contains a large number of manually judged queries on a five-point relevance scale, where we binarized relevance by assigning $r_i(y) = 1$ to all documents that got rated 3 and 4, and $r_i(y) = 0$ for ratings $0, 1, 2$. We adopt the train, validation and test splits that are pre-defined in the corpus. This means that queries in the three sets are disjoint, and we never train on any data from queries in the test set. To have a gold standard for reporting test-set performance, we measure performance on the binarized full-information ratings using Equation (4.2).

To generate click data from this full-information dataset of ratings, we first trained a standard Ranking SVM using 1% of the full-information training data to get a ranking function S_0. We employ S_0 as the "Production Ranker", and it is used to "present" rankings $\bar{\boldsymbol{y}}$ when generating the click data. We generate clicks using the rankings $\bar{\boldsymbol{y}}$ and ground-truth binarized relevances from the Yahoo dataset according to the following process. Depending on whether we are generating a training or a validation sample of click data, we first randomly draw a query x from the respective full-information dataset. For this query, we compute $\bar{\boldsymbol{y}} = S_0(x)$ and generate clicks based on the model from Section 4.6. Whenever a click is generated, we record a training example with its associated propensity $Q(o(y) = 1 \mid x, \bar{\boldsymbol{y}}, r)$. We model presentation bias via

$$Q(o(y) = 1 \mid x, \bar{\boldsymbol{y}}, r) = p_{rank(y|\bar{\boldsymbol{y}})} = \left(\frac{1}{rank(y \mid \bar{\boldsymbol{y}})}\right)^{\eta}. \tag{4.9}$$

The parameter η lets us control the severity of the presentation bias. We also introduce noise into the clicks according to the model described in Section 4.6.

When not mentioned otherwise, we use $\eta = 1$, $\epsilon_- = 0.1$, and $\epsilon_+ = 1$, which leads to click data where about 33% of the clicks are noisy clicks on irrelevant results and where the result at rank 10 has a 10% probability of being examined. We explore other bias profiles and noise levels in following experiments.

In all experiments, we select any parameters (e.g., C) of the learning methods via cross-validation on a validation set. The validation set is generated using the same click model as the training set, but using the queries in the validation fold of the Yahoo! dataset. For Propensity SVM-Rank, we always use the (unclipped) IPS estimator Equation (4.7) to estimate validation set performance. Keeping with the proportions of the original Yahoo data, the validation set size is always about 15% the size of the training set.

The primary baseline we compare against is a naïve application of SVM-Rank that simply ignores the bias in the click data. We call this method *Naïve SVM-Rank*. It is equivalent to a standard ranking SVM [52] but is most easily explained as equivalent to Propensity SVM-Rank with all q_j set to 1. Analogously, we use the corresponding naïve version of Equation (4.7) with propensities set to 1 to estimate validation set performance for Naïve SVM-Rank.

4.8.2 How Does Ranking Performance Scale With Training Data?

We first explore how the test-set ranking performance changes as we provide more click data to the learning algorithm. The resulting learning curves are given in Figure 4.1, and the performance of S_0 is plotted as a baseline. The

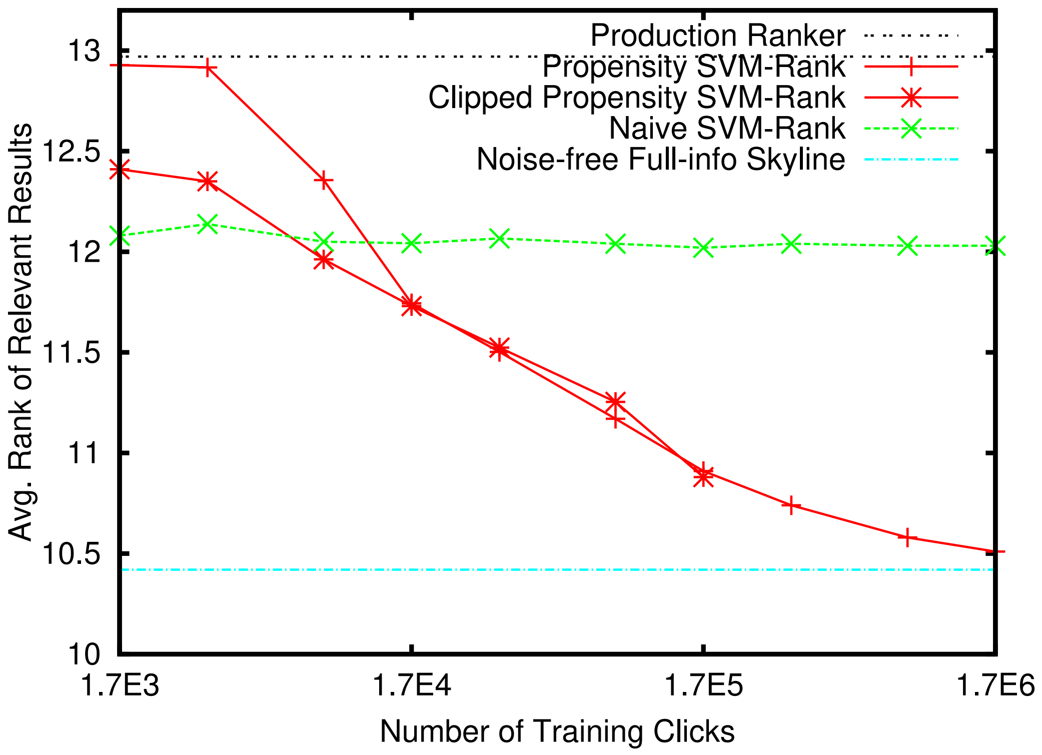


Figure 4.1: The test set performance in terms of Equation (4.2) for Propensity SVM-Rank with and without clipping compared to SVM-Rank naïvely ignoring the bias in clicks ($\eta = 1$, $\epsilon_{-} = 0.1$). The skyline is a Ranking SVM trained on all data without noise in the full-information setting, and the baseline is the production ranker S_0.

click data has presentation bias according to Equation (4.2) with $\eta = 1$ and noise $\epsilon_{-} = 0.1$. For small datasets, results are averaged over 5 draws of the click data.

With increasing amounts of click data, Propensity SVM-Rank approaches the skyline performance of the full-information SVM-Rank trained on the complete training set of manual ratings without noise. This behavior is in stark contrast to Naïve SVM-Rank which fails to account for the bias in the data and does not reach this level of performance. Furthermore, Naïve SVM-Rank cannot make effective use of additional data, and its learning curve is essentially flat. This behavior is consistent with the theoretical insight that estimation error in Naïve

SVM-Rank's empirical risk $\hat{R}(S)$ is dominated by asymptotic bias due to biased clicks, which does not decrease with more data and leads to suboptimal learning. The unbiased risk estimate $\hat{R}_{IPS}(S)$ of Propensity SVM-Rank, however, has estimation error only due to finite sample variance, which is decreased by more data and leads to consistent learning.

While unbiasedness is an important property when click data is plenty, the increased variance of $\hat{R}_{IPS}(S)$ can be a drawback for small datasets. This weakness can be seen in Figure 4.1, where Naïve SVM-Rank outperforms Propensity SVM-Rank for small datasets. We can remedy such behavior using techniques like "propensity clipping" [110], where small propensities are clipped to some threshold value τ to trade bias for variance.

$$\hat{R}_{CIPS}(S) = \frac{1}{n}\sum_{x_i}\sum_{y \in S(x_i)} \frac{rank(y \mid S(x_i)) \cdot r_i(y)}{\max\{\tau, Q(o_i(y) = 1 \mid x_i, \bar{\boldsymbol{y}}_i, r_i)\}}.$$

Figure 4.1 shows the learning curve of Propensity SVM-Rank with clipping, cross-validating both the clipping threshold τ and C. Clipping indeed improves performance for small datasets. While $\tau = 1$ is equivalent to Naïve SVM-Rank, the validation set is too small (and hence, the finite sample error of the validation performance estimate too high) to reliably select this model in every run. In practice, however, we expect click data to be plentiful such that lack of training data is unlikely to be a persistent issue.

4.8.3 How Much Presentation Bias Can Be Tolerated?

We now vary the severity of the presentation bias via η to understand its impact on Propensity SVM-Rank. Figure 4.2 shows that inverse propensity weighting is beneficial whenever substantial bias exists. Furthermore, increasing the amount

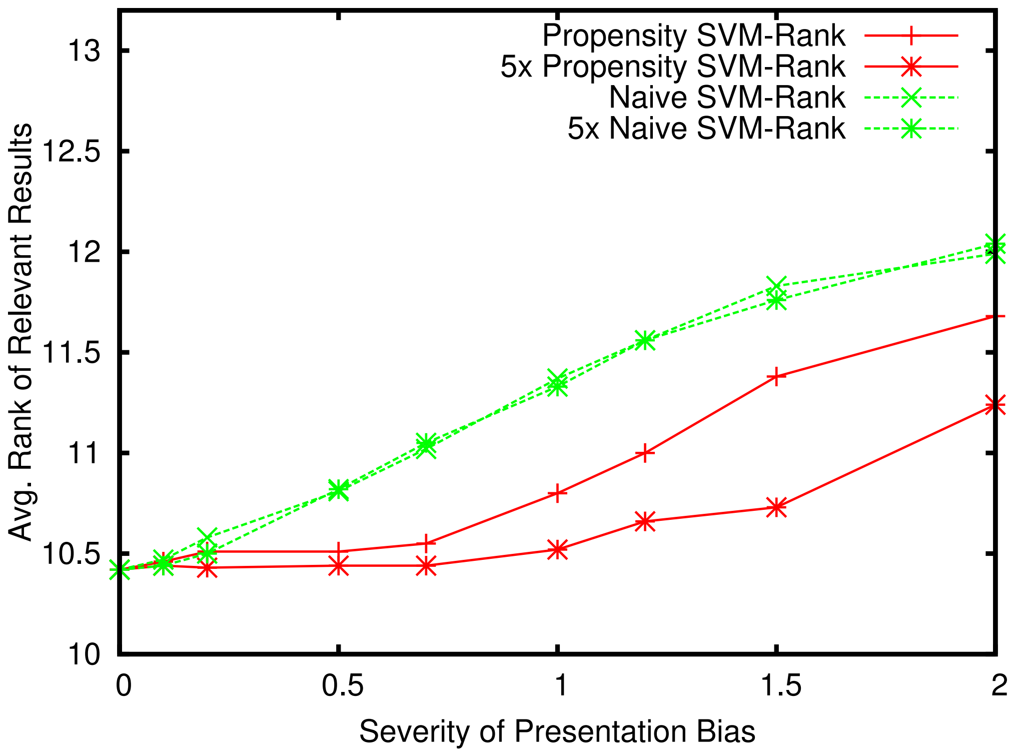


Figure 4.2: The test set performance for Propensity SVM-Rank and Naïve SVM-Rank as presentation bias becomes more severe in terms of η ($n = 45K$ and $n = 225K$. No click-noise was simulated.

of training data by a factor of 5 leads to further improvement for the Propensity SVM-Rank, while the added training data has no effect on Naïve SVM-Rank. This phenomenon is consistent with our arguments from Section 4.5 — more training data does not help when bias dominates estimation error, but it can reduce estimation error from variance in the unbiased risk estimate of Propensity SVM-Rank.

4.8.4 How Does Propensity SVM-Rank Handle Click Noise?

Figure 4.3 shows that Propensity SVM-Rank also enjoys a substantial advantage when it comes to noise. When increasing the noise level in terms of ϵ_- from

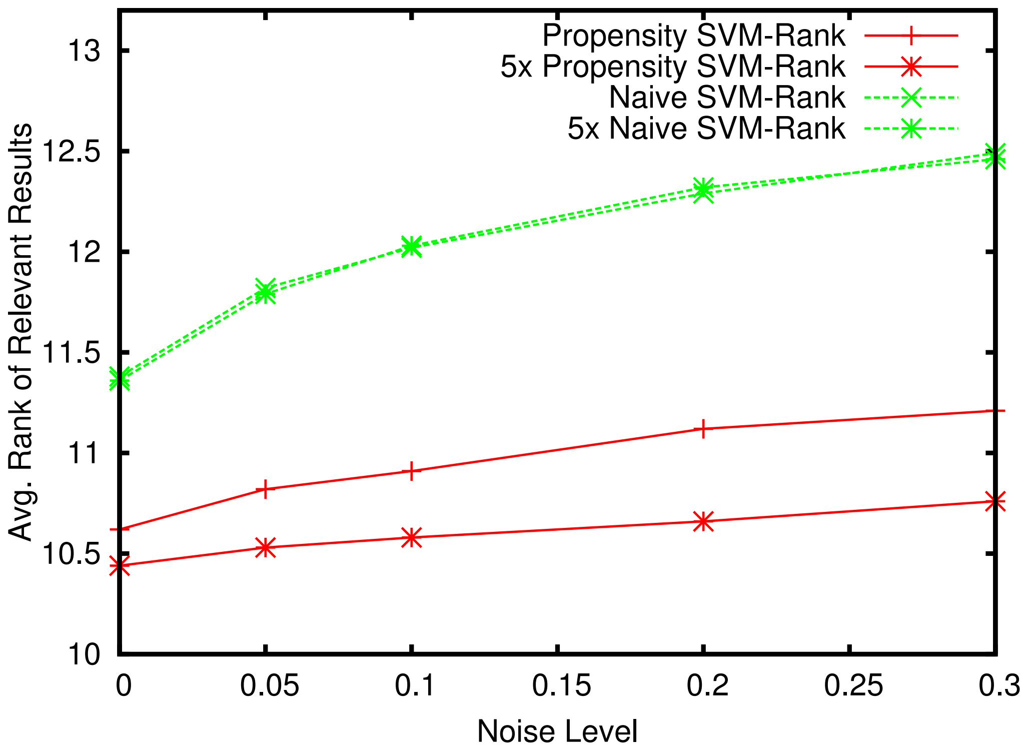


Figure 4.3: The test set performance for Propensity SVM-Rank and Naïve SVM-Rank as the noise level increases in terms of ϵ_- ($n = 170K$ and $n = 850K$, $\eta = 1$).

0 up to 0.3 (resulting in click data where 59.8% of all clicks are on irrelevant documents), Propensity SVM-Rank increasingly outperforms Naïve SVM-Rank. Moreover, the unbiasedness of the empirical risk estimate allows Propensity SVM-Rank to benefit from more data. We defer further study of click-noise models (e.g., $\epsilon_+ < 1$ under our noise model in Section 4.6.2) to future work.

4.8.5 How Does Propensity SVM-Rank Perform With Inaccurate Propensities?

So far all experiments have assumed that Propensity SVM-Rank has access to accurate propensities. In practice, however, propensities need to be estimated and

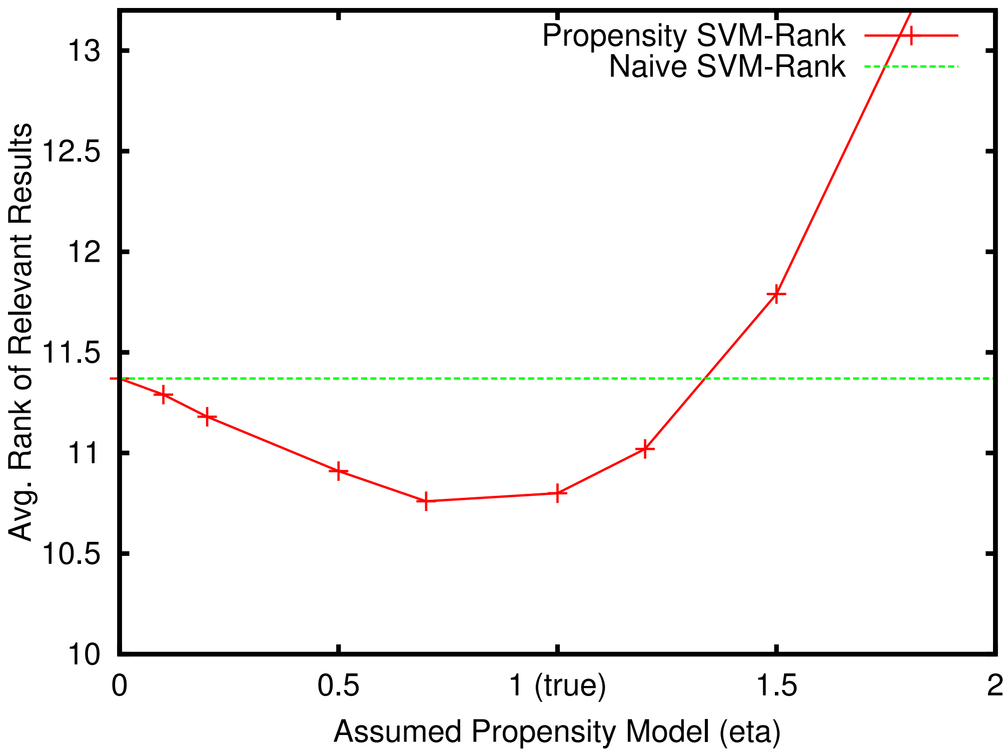


Figure 4.4: The test set performance for Propensity SVM-Rank and Naive SVM-Rank as propensities are misspecified (true $\eta = 1$, $n = 170K$, $\epsilon_- = 0.1$).

are subject to model misspecification. We now evaluate how robust Propensity SVM-Rank is to misspecified propensities. Figure 4.4 shows the performance of Propensity SVM-Rank when the training data is generated with $\eta = 1$, but the propensities used by Propensity SVM-Rank are misspecified using the η given in the x-axis of the plot. The plot shows that even misspecified propensities can give a substantial improvement over naïvely ignoring the bias, as long as the misspecification is "conservative" — i.e., overestimating small propensities is tolerable (which happens when $\eta < 1$), but underestimating small propensities can be harmful (which occurs when $\eta > 1$). This observation is consistent with theory, and clipping is one particular way of overestimating small propensities that can even improve performance. Overall, we conclude that even a mediocre propensity model can improve over the naïve approach — after all, the naïve approach can be thought of as a particularly poor propensity model that implicitly assumes no presentation bias and uniform propensities.

4.8.6 Real-World Experiment

We now examine the performance of Propensity SVM-rank when learning a new ranking function for the Arxiv Full-Text Search[2] based on real-world click logs from this system. The search engine uses a linear scoring function as outlined in Section 4.7. Query-document features $\mathbf{f}(x, y)$ are represented by a 1000-dimensional vector, and the production ranker used for collecting training clicks employs a hand-crafted [92] weight vector $\mathbf{w}$ (denoted Prod). Observed clicks on rankings served by this ranker over a period of 21 days provide implicit feedback data for LTR as outlined in Section 4.7.

To estimate the propensity model, we consider the simple position-based model of Section 4.6.1, and we collect new click data via randomized interventions for 7 days as outlined in Section 4.6.3 with landmark rank $k = 1$. Before presenting the ranking, we take the top-ranked document and swap it with the document at a uniformly at random chosen rank $j \in \{1, \dots 21\}$. The ratio of observed click-through rates (CTR) on the formerly top-ranked document now at position j versus its CTR at position 1 gives a noisy estimate of p_j/p_1 in the position-based click model. We additionally smooth these estimates by interpolating with the overall observed CTR at position j (normalized so that $CTR@1 = 1$). This procedure yields p_r that approximately decay with rank r with the smallest $p_r \simeq 0.12$. For $r > 21$, we impute $p_r = p_{21}$.

[2]http://search.arxiv.org:8081/

	Propensity SVM-Rank		
Interleaving Experiment	wins	loses	ties
against Prod	87	48	83
against Naïve SVM-Rank	95	60	102

Table 4.1: Per-query balanced interleaving [53] results for detecting relative performance between the hand-crafted production ranker used for click data collection (Prod), Naïve SVM-Rank and Propensity SVM-Rank.

We partition the click logs into a train-validation split: the first 16 days are the train set and provide 5437 click-events for SVM-rank, while the remaining 5 days are the validation set with 1755 click events. The hyper-parameter C is picked via cross-validation. Analogous to Section 4.8.1, we use the IPS estimator for Propensity SVM-Rank, and naive estimator with $Q(o(y) = 1 \mid x, \bar{\mathbf{y}}, r) = 1$ for Naïve SVM-Rank. With the best hyper-parameter settings, we re-train on all 21 days worth of data to derive the final weight vectors for either method.

We fielded these learned weight vectors in two online interleaving experiments [19], the first comparing Propensity SVM-Rank against Prod and the second comparing Propensity SVM-Rank against Naïve SVM-Rank. The results are summarized in Table 4.1. We find that Propensity SVM-Rank significantly outperforms the hand-crafted production ranker that was used to collect the click data for training (two-tailed binomial sign test $p = 0.001$ with relative risk 0.71 compared to null hypothesis). Furthermore, Propensity SVM-Rank similarly outperforms Naïve SVM-Rank, demonstrating that even a simple propensity model provides benefits on real-world data (two-tailed binomial sign test $p = 0.006$ with relative risk 0.77 compared to null hypothesis). Note that Propensity SVM-Rank not only significantly, but also substantially outperforms both other rankers in terms of effect size — and the synthetic data experiments suggest that additional training data will further increase its advantage.

4.9 Conclusions and Future Work

This chapter introduced a principled approach for learning-to-rank under biased feedback data. Drawing on counterfactual modeling techniques from causal inference, we presented a theoretically sound Empirical Risk Minimization framework for LTR. We instantiated this framework with a Propensity Weighted Ranking SVM, and provided extensive empirical evidence that the resulting learning method is robust to selection biases, noise and model misspecification. Furthermore, our real-world experiments on a live search engine showed that the approach leads to substantial retrieval improvements, without any heuristic or manual interventions in the learning process.

Beyond the specific learning methods and propensity models we proposed, our work may have an even bigger impact through its theoretical contribution to developing the general counterfactual model for LTR, thus articulating the key components necessary for LTR under biased feedback. First, the insight that propensity estimates are crucial for ERM learning opens a wide area of research on designing better propensity models. Second, the theory demonstrates that LTR methods should optimize propensity weighted ERM objectives, raising the question of which other learning methods beyond the Ranking SVM can be adapted to the Propensity ERM approach. Third, we conjecture that propensity weighted ERM approaches can also be developed for pointwise and listwise LTR methods.

Beyond learning from implicit feedback, propensity-weighted ERM techniques may prove useful even for optimizing offline IR metrics on manually annotated test collections. First, they can eliminate pooling bias, since the use of sampling during judgment elicitation puts us in a controlled setting where propensities are known by design. Second, propensities estimated via click models can enable click-based IR metrics like click-DCG to better correlate with the test set DCG in manually annotated test collections.

Part II

Learning from Logged Interventional Feedback

CHAPTER 5

BETTER ESTIMATORS FOR SLATE RECOMMENDATION

5.1 Chapter Notes

This chapter describes joint work with Akshay Krishnamurthy, Alekh Agarwal, Miroslav Dudík, John Langford, Damien Jose and Imed Zitouni. It is a lightly edited version of a workshop paper [118].

We estimate online metrics (e.g., revenue in the game recommendation example of Section 2.1) in situations where $\boldsymbol{y}$ is a combinatorial structure (recall the $x \mapsto y \mapsto \delta$ schematic). Rankings of documents in IR (Chapter 4) is a special case of combinatorial $\boldsymbol{y}$. This problem is an instance of causal estimation in the Off Policy setting (Section 2.3). Unlike Chapter 4, δ does not have a known or observable decomposition over $\boldsymbol{y}$ (e.g., the time it takes Alice to find and purchase a game in a carousel of recommendations is not any observable decomposition of per-recommendation values). Nevertheless, we reason about decompositions of $\boldsymbol{y}$ to construct an off-policy estimator. Remarkably we prove that this estimator is unbiased if there exists any (even unknown, unobservable, non-unique) decomposition of δ at all. We then employ this estimator to learn a ranking system using the classic Pointwise Learning-to-Rank (LTR) approach (see LTR survey [78]). Theoretically, we characterize the sample complexity (i.e., the number of samples needed to estimate to a certain accuracy) of our estimator. Empirically we see the first demonstration of high quality off policy estimation in both the real world and semi-synthetic combinatorial settings.

5.2 Introduction

In recommendation systems for e-commerce, online advertising, 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 [12, 71]. This task is called *off-policy evaluation* and standard approaches, namely *inverse propensity scores* (IPS) [47, 30], require unrealistically large amounts of past data to evaluate whole-page metrics that depend on multiple recommended items, such as when showing ranked lists. Therefore, the industry standard for evaluating new policies is to simply deploy them in weeks-long A/B tests [60]. Replacing or supplementing A/B tests with accurate off-policy evaluation, running in seconds instead of weeks, would revolutionize the process of developing better recommendation systems. For instance, we could perform automatic *policy optimization* (i.e., learn a policy that scores well on whole-page metrics), a task which is currently plagued with bias and an expensive trial-and-error cycle.

The data we collect in these recommendation applications provides only partial information, which is formalized as *contextual bandits* [5, 30, 66]. We study a combinatorial generalization of contextual bandits, where for each *context* a policy selects a list, called a *slate*, consisting of component *actions*. 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, and actions are the individual documents. Example metrics are page-level measures such as time-to-success, NDCG or more general measures of user satisfaction.

The key challenge in off-policy evaluation and optimization is the fact that a new policy, called the *target policy*, recommends different slates than those with recorded metrics in our logs. Without structural assumptions on the relationship between slates and observed metrics, we can only hope to evaluate the target policy if its chosen slates occur in the logged past data with a decent probability. Unfortunately, the number of possible slates is combinatorially large, e.g., when recommending ℓ of m items, there are $m^{\Omega(\ell)}$ ordered sets, so the likelihood of even one match in past data with a target policy is miniscule, leading to a complete breakdown of fully general techniques such as IPS.

To overcome this limitation, some authors [12, 113] restrict their logging and target policies to a parameterized stochastic policy class. Others assume specific parametric (e.g., linear) models relating the observed metrics to the features describing a slate [4, 98, 34, 22, 90]. Another paradigm, called *semi-bandits*, assumes that the slate-level metric is a linear combination of *observed* action-level metrics [59, 63].

We seek to evaluate arbitrary policies, while avoiding strong assumptions about user behavior, as in parametric bandits, or the nature of feedback, as in semi-bandits. We relax restrictions of both parametric and semi-bandits. Like semi-bandits, we assume that the slate-level metric is a sum of action-level metrics that depend on the context, the action, and the position on the slate, but not on the other actions in the slate. Unlike semi-bandits, these per-action metrics are *unobserved by the decision maker*. This model also means that the slate-level metric is linearly related to the unknown vector listing all the per-action metrics in each position. However, this vector of per-action metric values can depend arbitrarily on each context, which precludes fitting a single linear model of rewards (with dimensionality independent of the number of contexts) as usually done in linear bandits.

This chapter outlines the following contributions:

1. The *additive decomposition assumption* (ADA): an assumption about the feedback structure in combinatorial contextual bandits, which generalizes contextual, linear, and semi-bandits.
2. The *pseudoinverse estimator* (PI) for off-policy evaluation: a general purpose estimator (using linear bandit machinery) for any stochastic logging policy, unbiased under ADA. The number of logged samples needed for evaluation with error ϵ when choosing ℓ out of m items is typically $O(\ell m/\epsilon^2)$ — an exponential gain over the $m^{\Omega(\ell)}$ complexity of other unbiased estimators. We provide distribution-dependent bounds based on the overlap between logging and target policies.
3. Experiments on a real-world search ranking dataset: 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 Figure 5.1).
4. Off-policy optimization: We provide a simple procedure for learning to rank (L2R) using the PI estimator. Our procedure tunes L2R models directly to online metrics by leveraging pointwise supervised L2R approaches, without requiring pointwise feedback.

Without contexts, several authors have studied a similar linear dependence of the reward on action-level metrics [28, 98]. Their approaches compete with the *best fixed slate*, whereas we focus on evaluating arbitrary *context-dependent* policies. While they also use the pseudoinverse estimator in their analysis (e.g.,

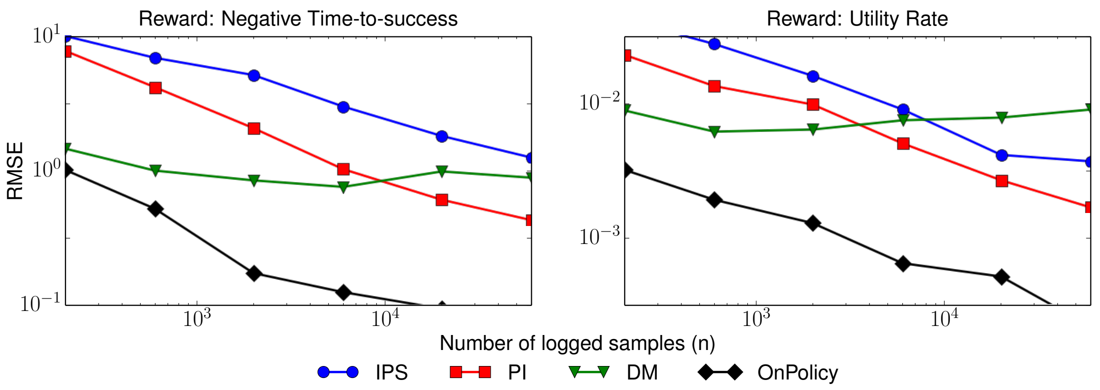


Figure 5.1: Off-policy evaluation results for two whole-page user satisfaction metrics on proprietary search engine data is reported. Average RMSE over 50 runs on a log-log scale. Our method (pseudoinverse or PI) achieves the best performance for moderate data sizes. The unbiased IPS method suffers high variance, and direct modeling (DM) suffers high bias. OnPolicy is the expensive alternative of deploying the policy. *Improvements of PI are significant, with p-values in text*. Details in Section 5.6.3.

Dani, Hayes and Kakade introduce it in Lemma 3.2 [28]), its role is different. They construct specific distributions to optimize the explore-exploit trade-off, while we provide guarantees for off-policy evaluation with arbitrary logging distributions, requiring a very different analysis and conclusions.

5.3 Setting and Notation

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

1. The decision maker observes a *context* x drawn from a distribution $\Pr(X)$ over some space X;
2. Based on the context, the decision maker chooses a *slate* $\mathbf{y} = (y_1, \dots y_\ell)$ consisting of *actions* y_j, where a position j is called a *slot*, the number of slots

is ℓ, actions at position j come from some space $Y_j(x)$, and the slate $\mathbf{y}$ is chosen from a set of allowed slates $Y(x) \subseteq Y_1(x) \times \cdots \times Y_\ell(x)$;

3. Given the context and slate, the environment draws a reward $\delta \in [-1, 1]$ from a distribution $\Pr(\Delta \mid x, \mathbf{y})$. Rewards in different rounds are independent, conditioned on contexts and slates.

The context space X can be infinite, but the set of actions is of finite size. For simplicity, we assume $|Y_j(x)| = m_j$ for all contexts $x \in X$ and define $m := \max_j m_j$ 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* π that specifies a conditional distribution $\pi(\mathbf{y} \mid x)$ (a deterministic policy is a special case). The *value* of policy π, denoted $V(\pi)$, is defined as the expected reward when following π:

$$V(\pi) := \mathbb{E}_x \mathbb{E}_{\mathbf{y} \sim \pi(\cdot \mid x)} \mathbb{E}_{\delta \sim \Pr(\cdot \mid x, \mathbf{y})} [\delta] \,.$$

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

To finish this section, we introduce notation for the expected reward for a context and slate, called the *slate value*, and denote as $V(x, \mathbf{y}) := \mathbb{E}_{\delta \sim \Pr(\cdot \mid x, \mathbf{y})}[\delta]$.

Example 1 (Cartesian product)**.** Consider whole-page optimization of a news portal where the reward is the whole-page advertising revenue. The context x is the user profile; the slate is the news portal page with slots corresponding to news sections or topics,[1] and actions are the news articles. It is natural to assume

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

that each article can only appear in one of the sections, so that $Y_j(x) \cap Y_k(x) = \emptyset$ if $j \neq k$. The set of valid slates is the Cartesian product $Y(x) = \prod_{j \le \ell} Y_j(x)$. The number of valid slates is exponential in ℓ, namely, $|Y(x)| = \prod_{j \le \ell} m_j$.

Example 2 (Ranking)**.** Consider information retrieval in web search. Here the context x is the user query along with user profile, time of day, etc. Actions correspond to search items (such as web pages). The policy chooses ℓ of m items, where the set $Y(x)$ of m items for a context x is selected from a large corpus by a fixed filtering step (e.g., a database query). We have $Y_j(x) = Y(x)$ for all $j \le \ell$, but the allowed slates $Y(x)$ have no repeated actions. The slots $j \le \ell$ correspond to positions on the search results page. The number of valid slates is exponential in ℓ since $|Y(x)| = m!/(m - \ell)! = m^{\Omega(\ell)}$. A reward could be the *negative time-to-success*, i.e., negative of the time taken by the user to find a relevant item, typically capped at some threshold if nothing relevant is found.

5.4 Off-Policy Evaluation and Optimization

In the *off-policy* setting, we have access to the *logged data* $(x_1, \mathbf{y}_1, \delta_1), \ldots (x_n, \mathbf{y}_n, \delta_n)$ collected using a past policy π_0, called the *logging policy*. *Off-policy evaluation* is the task of estimating the value of a new policy π, called the *target policy*, using the logged data. *Off-policy optimization* is the harder task of finding a policy $\hat{\pi}$ that improves upon the performance of π_0 and achieves a large reward. We mostly focus on off-policy evaluation and show how to use it as a subroutine for off-policy optimization in Section 5.6.2.

There are two standard approaches for off-policy evaluation. The *direct method* (DM) partitions the logged data and uses one subset to train a model

$\hat{\delta}(x, \mathbf{y})$ to predict the expected reward for a given context and slate. $V(\pi)$ is then estimated on the remaining data as

$$\hat{V}_{\mathrm{DM}}(\pi) = \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \sum_{\mathbf{y} \in Y(x_i)} \hat{\delta}(x_i, \mathbf{y}) \pi(\mathbf{y} \mid x_i). \tag{5.1}$$

The sum over $\mathbf{y}$ in Equation (5.1) can be estimated with a small sample (for stochastic π) and simplifies to a single $\mathbf{y}$ for deterministic π. The direct method is frequently biased because the reward model $\hat{\delta}(x, \mathbf{y})$ is typically misspecified.

The second approach, which is provably unbiased (under modest assumptions), is the *inverse propensity score* (IPS) estimator [47]. The IPS estimator reweights the logged data according to ratios of slate probabilities under the target and logging policy. It has two common variants:

$$\hat{V}_{\mathrm{IPS}}(\pi) = \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \delta_i \cdot \frac{\pi(\mathbf{y}_i|x_i)}{\pi_0(\mathbf{y}_i|x_i)}, \quad \hat{V}_{\mathrm{wIPS}}(\pi) = \sum_{i=1}^{n} \delta_i \cdot \frac{\pi(\mathbf{y}_i|x_i)}{\pi_0(\mathbf{y}_i|x_i)} \Big/ \left(\sum_{i=1}^{n} \frac{\pi(\mathbf{y}_i|x_i)}{\pi_0(\mathbf{y}_i|x_i)}\right). \tag{5.2}$$

The two estimators differ only in their normalizer. The IPS estimator is unbiased, whereas the weighted IPS (wIPS) is only asymptotically unbiased, but usually achieves smaller error due to smaller variance. Unfortunately, the variance of both estimators grows linearly with the magnitude of $\pi(\mathbf{y} \mid x)/\pi_0(\mathbf{y} \mid x)$, which can be as bad as $\Omega(|Y(x)|)$. This is prohibitive when $|Y(x)| = m^{\Omega(\ell)}$.

5.5 Our Approach and Assumptions

To reason about slates, we consider vectors in $\mathbb{R}^{\ell m}$ whose components are indexed by pairs (j, y) of slots and possible actions in them. A slate is then described by an indicator vector $\mathbf{1}_{\mathbf{y}} \in \mathbb{R}^{\ell m}$ whose entry at position (j, y) is equal to 1 if the slate $\mathbf{y}$ has action y in the slot j, i.e., $\mathbf{1}\{\mathbf{y}_j = y\}$.

At the foundation of our approach is an assumption relating the slate value to its component actions:

Assumption 1 (ADA)**.** A combinatorial contextual bandit problem satisfies the *additive decomposition assumption* (ADA) if for each context $x \in X$ there exists a (possibly unknown) *intrinsic reward* vector $\phi_x \in \mathbb{R}^{\ell m}$ such that the slate value decomposes as $V(x, \mathbf{y}) = \mathbf{1}_{\mathbf{y}}^T \phi_x = \sum_{j=1}^{\ell} \phi_x(j, \mathbf{y}_j)$.

ADA only posits the existence of intrinsic rewards, not their observability. This assumption distinguishes it from semi-bandits where $\{\phi_x(j, \mathbf{y}_j)\}_{j=1}^{\ell}$ can be observed for the $\mathbf{y}_j$'s chosen by π_0 in context x.

The slate value is described by a linear relationship between $\mathbf{1}_{\mathbf{y}}$ and the unknown "parameters" ϕ_x, but we do not require that ϕ_x be easy to fit from features describing contexts and actions, which is the key departure from the direct method and parametric bandits.

While ADA rules out some kinds of interactions among different actions on a slate,[2] its ability to vary intrinsic rewards arbitrarily across contexts can capture many common metrics in information retrieval, such as the *normalized discounted cumulative gain* (NDCG) [15], a common reward metric in web ranking:

Example 3 (NDCG)**.** For a given slate $\mathbf{y}$ we first define DCG:

$$\mathrm{DCG}(x, \mathbf{y}) := \textstyle\sum_{j=1}^{\ell} \frac{2^{rel(x, \mathbf{y}_j)} - 1}{\log_2(j+1)},$$

where $rel(x, y) \geq 0$ is the relevance of document y on query x. We define $\mathrm{NDCG}(x, \mathbf{y}) := \mathrm{DCG}(x, \mathbf{y}) / \mathrm{DCG}^{\star}(x)$ where $\mathrm{DCG}^{\star}(x) = \max_{\mathbf{y} \in Y(x)} \mathrm{DCG}(x, \mathbf{y})$, so NDCG takes values in $[0, 1]$. Thus, NDCG satisfies ADA with $\phi_x(j, y) = \left(2^{rel(x,y)} - 1\right) / \left(\mathrm{DCG}^{\star}(x) \log_2(j+1)\right)$.

[2] We discuss limitations of ADA and directions to overcome them in Section 5.7.

In addition to ADA, we also make the standard assumption that the logging policy puts a non-zero probability on all slates that can be potentially chosen by the target policy. This assumption is also required for the unbiasedness of IPS. Otherwise, off-policy evaluation is impossible [65].

Assumption 2 (ABS)**.** The off-policy evaluation problem satisfies the *absolute continuity* assumption if $\pi_0(\mathbf{y} \mid x) > 0$ whenever $\pi(\mathbf{y} \mid x) > 0$ with probability one over $x \sim \Pr(X)$.

5.5.1 The Pseudoinverse Estimator

Our estimator uses certain moments of the logging policy π_0, called *marginal values* and denoted $\boldsymbol{\theta}_{\pi_0,x} \in \mathbb{R}^{\ell m}$, and their empirical estimates called *marginal rewards* and denoted $\hat{\boldsymbol{\theta}}_i \in \mathbb{R}^{\ell m}$:

$$\boldsymbol{\theta}_{\pi_0,x} := \mathbb{E}_{\pi_0}[\delta \mathbf{1}_{\mathbf{y}} \mid x] \quad \text{and} \quad \hat{\boldsymbol{\theta}}_i := \delta_i \mathbf{1}_{\mathbf{y}_i}.$$

Recall that π_0 is viewed here as a distribution over triples $(x, \mathbf{y}, \delta)$. In words, the components $\boldsymbol{\theta}_{\pi_0,x}(j, y)$ accumulate the rewards only when the policy π_0 chooses a slate $\mathbf{y}$ with $\mathbf{y}_j = y$. The random variable $\hat{\boldsymbol{\theta}}_i$ estimates $\boldsymbol{\theta}_{\pi_0,x}$ at x_i by the observed reward for the slate $\mathbf{y}_i$ displayed for x_i in our logs. The key insight is that the marginal value $\boldsymbol{\theta}_{\pi_0,x}(j, y)$ provides an indirect view of $\phi_x(j, y)$, occluded by the effect of actions in slots $k \neq j$. Specifically, from ADA and the definition of $\boldsymbol{\theta}_{\pi_0,x}$, we obtain

$$\boldsymbol{\theta}_{\pi_0,x}(j, y) = \pi_0(\mathbf{y}_j = y \mid x)\phi_x(j, y) + \sum_{k \neq j} \sum_{y' \in Y_k(x)} \pi_0(\mathbf{y}_j = y, \mathbf{y}_k = y' \mid x)\phi_x(k, y'). \tag{5.3}$$

Equation (5.3) represents a linear relationship between $\boldsymbol{\theta}_{\pi_0,x}$ and ϕ_x, which is concisely described by a matrix $\boldsymbol{\Gamma}_{\pi_0,x} \in \mathbb{R}^{\ell m \times \ell m}$, with

$$\Gamma_{\pi_0,x}(j, y;\, k, y') := \begin{cases} \pi_0(\mathbf{y}_j = y \mid x) & \text{if } j = k \text{ and } y = y', \\ \pi_0(\mathbf{y}_j = y, \mathbf{y}_k = y' \mid x) & \text{if } j \neq k, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, $\boldsymbol{\theta}_{\pi_0,x} = \boldsymbol{\Gamma}_{\pi_0,x}\phi_x$. If $\boldsymbol{\Gamma}_{\pi_0,x}$ was invertible, we could write $\phi_x = \boldsymbol{\Gamma}^{-1}_{\pi_0,x}\boldsymbol{\theta}_{\pi_0,x}$ and use ADA to obtain $V(x, \mathbf{y}) = \mathbf{1}_{\mathbf{y}}^T \boldsymbol{\Gamma}^{-1}_{\pi_0,x}\boldsymbol{\theta}_{\pi_0,x}$. We could then replace $\boldsymbol{\theta}_{\pi_0,x_i}$ by its unbiased estimate $\hat{\boldsymbol{\theta}}_i$ to get an unbiased estimate of $V(x_i, \mathbf{y})$. In reality, $\boldsymbol{\Gamma}_{\pi_0,x}$ is not invertible. However, it turns out that the above strategy still works, we just need to replace the inverse by the pseudoinverse: [3]

Theorem 4. *If ADA holds and $\pi_0(\mathbf{y} \mid x) > 0$, then $V(x, \mathbf{y}) = \mathbf{1}_{\mathbf{y}}^T \boldsymbol{\Gamma}^{\dagger}_{\pi_0,x}\boldsymbol{\theta}_{\pi_0,x}$.*

This strategy gives rise to the *pseudoinverse estimator* (PI):

$$\hat{V}_{\mathrm{PI}}(\pi) = \frac{1}{n}\sum_{i=1}^{n}\sum_{\mathbf{y}\in Y(x_i)} \pi(\mathbf{y} \mid x_i)\mathbf{1}_{\mathbf{y}}^T \boldsymbol{\Gamma}^{\dagger}_{\pi_0,x_i}\hat{\boldsymbol{\theta}}_i = \frac{1}{n}\sum_{i=1}^{n} \delta_i \cdot \mathbf{q}^T_{\pi,x_i}\boldsymbol{\Gamma}^{\dagger}_{\pi_0,x_i}\mathbf{1}_{\mathbf{y}_i}. \tag{5.4}$$

In Equation (5.4), we have expanded the definition of $\hat{\boldsymbol{\theta}}_i$ and introduced the notation $\mathbf{q}_{\pi,x}$ for the expected slate indicator under π conditioned on x, $\mathbf{q}_{\pi,x} := \mathbb{E}_\pi[\mathbf{1}_{\mathbf{y}} \mid x]$. The sum over $\mathbf{y}$ required to obtain $\mathbf{q}_{\pi,x_i}$ in Equation (5.4) can be estimated with a small sample for stochastic π and simplifies to a single $\mathbf{y}$ for deterministic π, analogous to the direct method.

Theorem 4 immediately yields the unbiasedness of $\hat{V}_{\mathrm{PI}}$:

[3] A variant of Theorem 4 is proved in a different context by Dani, Hayes, and Kakade [28]. Our proof, alongside proofs of all other statements in this chapter, is in Appendix B.

Theorem 5. *If ADA and ABS hold, then the estimator $\hat{V}_{\mathrm{PI}}$ is unbiased, i.e., $\mathbb{E}_{\pi_0^n}[\hat{V}_{\mathrm{PI}}] = V(\pi)$, where the expectation is over the n logged examples sampled i.i.d. from π_0.*

To better motivate the PI estimator and its suitability for off-policy evaluation, we now present some examples where the estimator can be further simplified and takes intuitive forms. We begin by showing that it can be viewed as a direct extension of IPS to combinatorial settings.

Example 4 (PI when $\ell = 1$)**.** When the slate consists of a single slot, the policies recommend a single action chosen from some set $Y(x)$ for a context x. In this case, PI coincides with IPS since

$$\mathbf{\Gamma}_{\pi_0,x} = \mathrm{diag}(\pi_0(y \mid x))_{y\in Y(x)},\ \mathbf{\Gamma}^{\dagger}_{\pi_0,x} = \mathrm{diag}(1/\pi_0(y \mid x))_{y\in Y(x)},\ \text{and } \mathbf{q}_{\pi,x} = (\pi(y \mid x))_{y\in Y(x)}.$$

Our second example considers a self-evaluation setting, where $\pi = \pi_0$ and highlights that PI is maximally data efficient in this extreme.

Example 5 (PI when $\pi = \pi_0$)**.** When the target policy coincides with the logging policy, the estimator simplifies to the average of rewards: $\hat{V}_{\mathrm{PI}}(\pi) = \frac{1}{n}\sum_{i=1}^{n}\delta_i$ (see Appendix B.3). For $\ell = 1$, this follows from the previous example, but it is non-trivial to show for $\ell \geq 2$.

Finally, we take two particular structured action sets discussed in Example 1 and 2. In both cases, we simplify the PI estimator to highlight the sources of improvement over the vanilla IPS estimator.

Example 6 (PI for a Cartesian product with uniform logging)**.** The PI estimator for the Cartesian product slate space when π_0 is uniform simplifies to

$$\hat{V}_{\mathrm{PI}}(\pi) = \frac{1}{n}\sum_{i=1}^{n}\delta_i \cdot \left(\sum_{j=1}^{\ell}\frac{\pi(\mathbf{y}_{ij}|x_i)}{1/m_j} - \ell + 1\right),$$

by Proposition 5 in Appendix B.4.1. 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 $O(\ell m)$, whereas the IPS terms are $m^{\Omega(\ell)}$.

Example 7 (PI for rankings with $\ell = m$ and uniform logging)**.** In this case, the PI estimator equals

$$\hat{V}_{\mathrm{PI}}(\pi) = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \delta_i \cdot \big(\sum_{j=1}^{\ell} \tfrac{\pi(\mathbf{y}_{ij}|x_i)}{1/(m-1)} - m + 2\big),$$

by Proposition 6 in Appendix B.4.1. Individual terms are $O(\ell m) = O(m^2)$.

5.5.2 Deviation Analysis

We have shown that the pseudoinverse estimator is unbiased given ADA and have also given examples when it improves exponentially over IPS, the existing state-of-the-art for off-policy evaluation. We next derive a distribution-dependent bound on the finite sample error and use it to obtain an exponential improvement over IPS for a broader class of logging distributions.

Our deviation bound is obtained by an application of Bernstein's inequality, which requires bounding the variance and range of the terms appearing in Equation (5.4), namely $\delta_i \cdot \mathbf{q}_{\pi,x_i}^T \mathbf{\Gamma}_{\pi_0,x_i}^{\dagger} \mathbf{1}_{\mathbf{y}_i}$. We bound their variance and range, respectively, by the following distribution-dependent quantities:

$$\sigma^2 := \mathbb{E}_x\left[\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{q}_{\pi,x}\right], \quad \rho := \sup_x \sup_{\mathbf{y}:\pi_0(\mathbf{y}|x)>0} \left|\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}}\right| \; . \tag{5.5}$$

The quantity σ^2 bounds the variance whereas ρ bounds the range. They capture the "average" and "worst-case" mismatch between the logging and target policy. They equal one when $\pi = \pi_0$ (see Appendix B.3), and in general, yield the following deviation bound:

Theorem 6. *Assume that ADA and ABS hold, and let σ^2 and ρ be defined as in Equation* (5.5). *Then, for any $\eta \in (0, 1)$, with probability at least $1 - \eta$,*

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

Observe that this finite sample bound is structurally different from the regret bounds developed for combinatorial bandits. The bound incorporates the extent of overlap between π and π_0 so that we have a higher confidence in our estimates when the logging and evaluation policies are similar — these characterizations are important for off-policy evaluation.

In Appendix B.4, Proposition 4, we show that $\sigma^2 \le \rho$, so to bound $\hat{V}_{\mathrm{PI}}$, it suffices to bound ρ. While this leads to less precise bounds in general, it is a natural choice since ρ is often significantly easier to bound than σ^2, often in a distribution-agnostic manner. We next show such a bound for a broad class of logging policies defined as follows:

Definition 2. *Let ν denote the uniform policy, that is, $\nu(\mathbf{y} \mid x) = 1/|Y(x)|$ and let π be an arbitrary stochastic policy. We say that the policy π_κ is κ-*uniform *for some $\kappa \in (0, 1]$ if for all contexts x, actions $\mathbf{y}, \mathbf{y}'$, we have*

$$\pi_\kappa(\mathbf{y} \mid x) = \kappa\nu(\mathbf{y} \mid x) + (1 - \kappa)\pi(\mathbf{y} \mid x).$$

For the Cartesian product slate space, this means that $\pi_\kappa(\mathbf{y}_j = y, \mathbf{y}_k = y' \mid x) \ge \kappa/(m_j m_k)$ for $j \ne k$. For rankings, $\pi_\kappa(\mathbf{y}_j = y, \mathbf{y}_k = y' \mid x) \ge \kappa/(m(m-1))$

for $j \neq k$. These lower bounds on the entries of $\mathbf{\Gamma}_{\pi_0,x}$ will prove to be very useful when deriving finite sample bounds. The κ-uniform definition captures ϵ-greedy policies which are a popular class of logging policies [14] in practice.

Proposition 2. *Assume that valid slates form a Cartesian product space as in Example 1 or are rankings as in Example 2. Then for any κ-uniform logging policy, we have $\rho \leq \kappa^{-1}\ell m$.*

Thus, using the fact that $\sigma^2 \leq \rho$, Proposition 2 and Theorem 6 yield $|\hat{V}_{\mathrm{PI}}(\pi) - V(\pi)| \leq O(\sqrt{\kappa^{-1}\ell m/n})$, or equivalently $O(\kappa^{-1}\ell m/\eta^2)$ logging samples are needed to achieve accuracy η.

5.6 Experiments

We now empirically evaluate the performance of the pseudoinverse estimator in the ranking scenario of Example 2. We first show that our approach compares favorably to baselines in a semi-synthetic evaluation on a public data set under the NDCG metric, which satisfies ADA as discussed in Example 3. On the same data, we further use the pseudoinverse estimator for *off-policy optimization*, that is, to learn ranking policies, competing against a supervised baseline 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 are *a priori* unlikely to satisfy ADA. More detailed results are deferred to Appendices B.5, B.6, and B.7.

5.6.1 Semi-Synthetic Evaluation

Our semi-synthetic evaluation uses labeled data from the LETOR4.0 MQ2008 dataset [91] to create a contextual bandit problem. Queries provide the contexts x and actions y are the available documents. The dataset contains 784 queries, 5—121 documents per query and relevance labels $rel(x, y) \in \{0, 1, 2\}$ for each query-document pair. Each pair (x, y) has a 47-dimensional feature vector $\mathbf{f}(x, y)$, which can be partitioned into title features $\mathbf{f}_{\text{title}}$, and body features $\mathbf{f}_{\text{body}}$.

To derive a logging policy and a distinct target policy, we first train two lasso regression models, called $pred_{\text{title}}$ and $pred_{\text{body}}$, to predict relevances from $\mathbf{f}_{\text{title}}$ and $\mathbf{f}_{\text{body}}$, respectively. To create the logged data queries x are sampled uniformly, and the set $Y(x)$ consists of top m documents according to $pred_{\text{title}}$. The logging policy π_0 samples from a multinomial distribution over documents in $Y(x)$, parameterized by $\alpha \geq 0$: $p_\alpha(y \mid x) \propto \exp(\alpha \cdot pred_{\text{title}}(x, y))$. Slates are constructed slot-by-slot, sampling *without replacement* according to p_α. Choosing $\alpha \in [0, \infty)$ interpolates between uniformly random and deterministic logging. Our target policy π selects the top ℓ documents according to $pred_{\text{body}}$ to sequentially populate the slate. The slate reward is the NDCG metric defined in Example 3.

We compare our estimator PI with the direct method (DM) and weighted IPS (wIPS, see Equation (5.2)), which out-performed IPS. Our implementation of DM concatenates per-slot features $\mathbf{f}(x, y)$ to produce a slate-level feature vector $\mathbf{f}(x, \mathbf{y})$, training a reward predictor on the first $n/2$ examples and evaluating π using Equation (5.1) on the other $n/2$ examples. We experimented with regression trees, ridge and lasso regression for DM, and always report results for the choice with the smallest RMSE at $n = 10^6$ examples. We also include an aspirational baseline, ONPOLICY. This "skyline" corresponds to deploying the target policy as in an A/B test and returning the average of observed rewards [60]. This approach is the expensive alternative we wish to avoid. We plot the root mean square error (RMSE) of the estimators as a function of increasing data size over at least 20 independent runs.

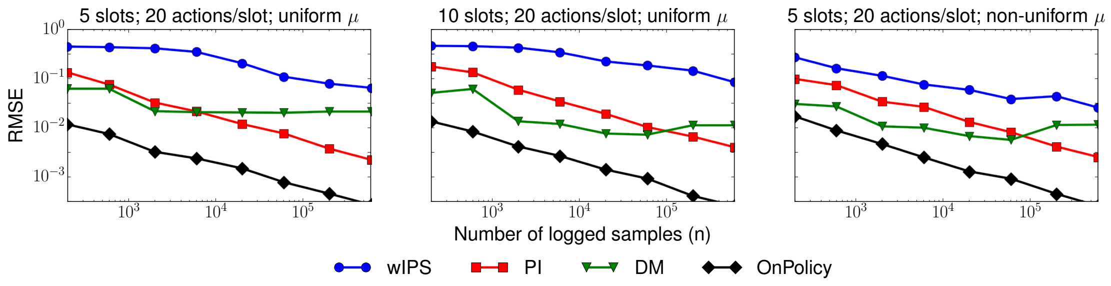


Figure 5.2: RMSE of different off-policy value estimators under uniform logging ($\alpha = 0$) and non-uniform logging ($\alpha = 10$) for three different slate recommendation problems.

In Figure 5.2, the first two plots study the RMSE of estimators for two choices of m and ℓ, given the uniform logging policy $\pi_0 \equiv \nu$ (i.e., $\alpha = 0$). In both cases, the pseudoinverse estimator outperforms wIPS by a factor of 10 or more with high statistical significance, $p < 10^{-8}$ for both plots and all n. The pseudoinverse estimator eventually also outperforms the biased DM with statistical significance, with $p \leq 7.3 \times 10^{-4}$ for both plots at $n \geq 600$K. The cross-over point occurs fairly early ($n \approx 10$K) for the smaller slate space, but is one order larger ($n \approx 100$K) for the biggest slate space. Note that DM's performance can deteriorate with more data because it optimizes the fit to the reward distribution of π_0, which is different from that of π.

As expected, OnPolicy performs the best, requiring between 10x and 100x less data. However, OnPolicy requires fixing the target policy π for each data collection, while off-policy methods like PI take advantage of massive amounts of logged data to evaluate arbitrary policies. As an aside, since the user feedback in these experiments is simulated, we can also simulate semi-bandit feedback (which reveals the intrinsic reward of each shown action) for off-policy evaluation. This simulation is a purely hypothetical baseline: with only page-level feedback, one cannot implement a semi-bandit solution. We compare against this hypothetical baseline in Appendix B.6.

In Figure 5.2 (right panel), we study the effect of the overlap between the logging and target policies, by taking $\alpha = 10$, which results in a better alignment between the logging and target policies. While the RMSE of the pseudoinverse estimator is largely unchanged, both wIPS and DM show some improvement. wIPS enjoys a smaller variance due to a lower range of importance weights, while DM enjoys a lower bias due to closer training and target distributions. PI continues to be statistically better than wIPS, with $p \leq 10^{-8}$ for all n, and eventually also better than DM, with $p \leq 4.4 \times 10^{-4}$ starting at $n \geq 200$K. See Appendices B.5 and B.6 for more results and the complete set of p-values.

5.6.2 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 our targets for regression. *Thus, the pseudoinverse estimator enables pointwise L2R even without relevance labels.* Given a contextual bandit dataset $\{(x_i, \mathbf{y}_i, \delta_i)\}_{i \le n}$ collected by the logging policy π_0, we begin by creating the estimates of ϕ_{x_i}: $\hat{\boldsymbol{\phi}}_i = \boldsymbol{\Gamma}^{\dagger}_{\pi_0, x_i} \hat{\boldsymbol{\theta}}_i$, turning the i-th contextual bandit example into ℓm 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).

We used the MQ2008 dataset from Section 5.6.1 and created a contextual bandit problem with 5 slots and 20 candidate documents, with a uniformly random logging policy. We chose a standard 5-fold split and always trained on bandit data from 4 folds and evaluated using the supervised data on the fifth. We compare our approach, titled PI-OPT, against the supervised approach, trained to predict the *gains*, equal to $2^{rel(x,y)} - 1$, computed using annotated relevance judgments in the training fold (we found that predicting raw relevances was inferior). Both PI-OPT and SUP train regression trees. We observe that PI-OPT is consistently competitive with SUP after seeing about 1K samples containing slate-level feedback, and gets a test NDCG of 0.450 at 1K samples, 0.451 at 10K samples, and 0.456 at 100K samples. SUP achieves a test NDCG of 0.453 by using approximately 12K annotated relevance judgments. We posit that PI-OPT is competitive with SUP because it optimizes the target metric directly, while SUP uses an imperfect target surrogate. See Appendix B.7 for detailed results.

5.6.3 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 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 candidate slates. To control the query distribution ($\Pr(X)$) in our experiment, we generate a larger dataset by bootstrap sampling, repeatedly choosing a query uniformly at random and a slate uniformly at random from those shown for this query. Hence, the conditional probability of any slate for a given query matches the frequencies in the original data.

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 a sufficiently long duration. TTS values are capped and scaled to $[0, 1]$. UTILITYRATE is a more complex page-level metric of a user's 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 that was trained to predict clicks and uses these 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 by design. Since we know the query distribution, we can calculate the target policy's value $V(\pi)$ exactly, and measure RMSE relative to this true value.

We compare our estimator (PI) with three baselines similar to those from Section 5.6.1: DM, IPS, and ONPOLICY. DM uses regression trees over roughly 20K slate-level features.

Figure 5.1 from Section 5.2 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. For TTS, the gains over IPS are significant with $p \leq 3.7 \times 10^{-5}$ after 2K samples and for DM with $p \leq 1.5 \times 10^{-3}$ after 20K samples. For UTILITYRATE, the improvements on IPS are significant with $p < 10^{-8}$ at 60K examples, and over DM with $p \leq 4.3 \times 10^{-7}$ after 20K examples. The complete set of p-values is in Appendix B.5.

5.7 Conclusions and Future Work

In this chapter, we have introduced a new assumption (ADA), a new estimator (PI) that exploits this assumption, and demonstrated their significant theoretical and practical merits.

In our experiments, we saw examples of bias-variance trade-off with off-policy estimators. At small sample sizes, the pseudoinverse estimator still has a non-trivial variance. In these regimes, the biased direct method can often be practically useful due to its low variance (if its bias is sufficiently small). Such well-performing albeit biased estimators can be incorporated into the pseudoinverse estimator via the doubly-robust approach [30, 31].

Experiments with real-world data in Section 5.6.3 demonstrate that even when ADA does not hold, the estimators based on ADA can still be applied and tend to be superior to alternatives. We view ADA similarly to the IID assumption: while it is probably often violated in practice, it leads to practical algorithms that remain robust under misspecification. Similarly to the IID assumption, we are not aware of ways for easily testing whether ADA holds.

One promising approach to relax ADA 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 a refined sample-complexity analysis remains open.

CHAPTER 6

BATCH LEARNING FROM LOGGED BANDIT FEEDBACK

6.1 Chapter Notes

This chapter describes joint work with Thorsten Joachims. It is a lightly edited combination of a journal article [113], a conference publication [115] and a workshop paper [114].

Recall the $x \mapsto y \mapsto \delta$ schematic. In this chapter, we make no assumptions about the structure of δ. We focus on the Off-Policy setting (see Section 2.3). We formalize the data collected through $x \mapsto y \mapsto \delta$ interactions as logged contextual bandit feedback. The goal is to build a general learning machine (see Section 2.4) that can find good interaction policies using logged bandit data — Batch Learning from Bandit Feedback (BLBF). We derive new data-dependent variance regularizers for use with classic learning algorithms [64, 122]. Theoretically, we prove generalization error bounds of these variance-regularized algorithms analogous to Structural Risk Minimization (Section 2.4). Empirically we show in semi-synthetic experiments (following the methodology sketched in Section 1.2) that variance regularization allows these new algorithms to improve their performance in off-policy learning substantially.

6.2 Introduction

Log data is one of the most ubiquitous forms of data available, as it can be recorded from a variety of systems (e.g., search engines, recommender systems, ad placement) at little cost. The interaction logs of such systems typically contain a record of the input to the system (e.g., features describing the user), the prediction made by the system (e.g., a recommended list of news articles) and the feedback (e.g., the number of ranked articles the user read) [71]. The feedback, however, provides only partial information — "bandit feedback" — limited to the particular prediction shown by the system. The feedback for all the other predictions the system could have made is typically not known. This missing information makes learning from log data fundamentally different from supervised learning, where "labels" (e.g., the best ranking of news articles for that user) together with a loss function provide full-information feedback.

In this chapter, we address the problem of learning from logged bandit feedback. Unlike the well-studied problem of online learning from bandit feedback [17], Batch Learning with Bandit Feedback (BLBF) does not require interactive experimental control over the system. Furthermore, it enables the reuse of existing data and offline cross-validation techniques for model selection (e.g., "which features to use?", "which learning algorithm to use?", etc.).

To design algorithms for batch learning from bandit feedback, *counterfactual* estimators [12] of a system's performance can be used to estimate how other systems would have performed if they had been in control of choosing predictions. Such estimators have been developed recently for the off-policy evaluation problem [31, 72, 70], where data collected from the interaction logs of one interactive system is used to evaluate another system. Learning in such a setting is closely related to the problem of off-policy reinforcement learning [112] — we would like to know how well a new system (policy) would perform *if it had been* used in the past.

Our approach to counterfactual learning centers around the insight that, to perform robust learning, it is not sufficient to have just an unbiased estimator of the off-policy system's performance. We must also reason about how the variances of these estimators differ across the hypothesis space, and pick the hypothesis that has the best possible guarantee (tightest conservative bound) for its performance. We first prove generalization error bounds for a *stochastic hypothesis* family using an empirical Bernstein argument [82]. These bounds complement recent approaches to deriving confidence intervals for counterfactual estimators [12, 119]. By relating the generalization error to the empirical sample variance of different hypotheses, we can effectively penalize the hypotheses with large variance during training using a data-dependant regularizer. In analogy to Structural Risk Minimization for full-information feedback [124], the constructive nature of these bounds suggests a general principle — Counterfactual Risk Minimization (CRM) — for designing methods for BLBF.

Using the CRM principle, we derive a new learning algorithm — Policy Optimizer for Exponential Models (POEM) — for structured output prediction. The training objective is decomposed using repeated variance linearization, and optimizing it using AdaGrad [29] yields a fast and efficient algorithm. We evaluate POEM on several multi-label classification problems, verify that its empirical performance supports the theory, and demonstrates substantial gain in generalization performance over the state-of-the-art. We then use POEM in a real-world experiment for learning a high precision classifier for information retrieval using logged click data.

The remainder of this chapter is structured as follows. We review existing approaches in Section 6.3. The learning setting is detailed in Section 6.4 and contrasted with supervised learning. In Section 6.5, we derive the Counterfactual Risk Minimization learning principle and provide a rule of thumb for setting hyper-parameters. In Section 6.6, we instantiate the CRM principle for structured output prediction using exponential models and construct an efficient decomposition of the objective for stochastic optimization. Empirical evaluations are reported in Section 6.7, and a real-world application is described in Section 6.8. We conclude with future directions and discussion in Section 6.9.

6.3 Related Work

Existing approaches for batch learning from logged bandit feedback fall into two categories. The first approach is to reduce the problem to supervised learning. In principle, since the logs give us an incomplete view of the feedback for different predictions, one could first use regression to estimate a feedback oracle for unseen predictions, and then use any supervised learning algorithm using this feedback oracle. Such a two-stage approach is known not to generalize well [8]. More sophisticated techniques using the Offset Tree algorithm [8] allow us to perform batch learning when the space of possible predictions is small. In contrast, our approach generalizes structured output prediction, with exponentially-large prediction spaces. In experiments, we apply our approach to multi-label classification problems. When the number of labels is K, the number of possible predictions is 2^K. A direct application of the Offset tree algorithm requires $O(2^K)$ space and only guarantees regret $O((2^K - 1)r)$ where r is the "regret" of an underlying binary classifier. Our approach directly tackles the problem using popular structured prediction models, with computation and space complexity that mirrors these supervised structured prediction algorithms.

The second approach to batch learning from bandit feedback uses propensity scoring [95] to derive unbiased estimators from the interaction logs [12]. These estimators are used for a small set of candidate policies, and the best candidate is picked via exhaustive search. In contrast, our approach can be optimized via gradient descent, over hypothesis families (of infinite size) that are equally as expressive as those used in supervised learning. In particular, we build on recent work that develops confidence bounds for counterfactual estimators [12, 119] using empirical Bernstein bounds. Our key insight is that these confidence intervals are not merely observable but can be efficiently optimized during training. Other recent bounds derived from Renyi divergences [24] can analogously be co-opted in our approach to counterfactual learning.

Our approach builds on counterfactual estimators that have been developed for off-policy evaluation. The inverse propensity scoring approach can work well when we have a good model of the historical algorithm [110, 70, 73], and doubly robust estimators [31, 30] are even more effective when we additionally have a good model of the feedback. In our work, we focus on the inverse propensity scoring estimator, but the results we derive also hold for the doubly robust estimators.

In the current work, we concentrate on the case where the historical algorithm was a stationary, stochastic policy. Techniques like exploration scavenging [65] and bootstrapping [81] allow us to perform counterfactual evaluation even when the historical algorithm was deterministic or adaptive.

Our strategy of picking the hypothesis with the tightest conservative bound on performance mimics similar successful approaches in other problems like supervised learning [124], risk averse multi-armed bandits [36], regret minimizing contextual bandits [66] and reinforcement learning [37]. Beyond the problem of batch learning from bandit feedback, our approach can have implications for several applications that require learning from logged bandit feedback data: warm-starting multi-armed bandits [105] and contextual bandits [110], pre-selecting retrieval functions for search engines [45], policy evaluation for contextual bandits [72], and reinforcement learning [119] to name a few.

6.4 Learning Setting: Batch Learning from Bandit Feedback

Consider a structured output prediction problem that takes as input $x \in \mathcal{X}$ and outputs a prediction $y \in \mathcal{Y}$. For example, in multi-label document classification, x could be a news article and y a bit vector indicating the labels assigned to this article. The inputs are assumed drawn from a fixed but unknown distribution $\Pr(\mathcal{X})$, $x \overset{i.i.d.}{\sim} \Pr(\mathcal{X})$. Consider a hypothesis space $\mathcal{H}$ of *stochastic policies*. A hypothesis $\pi(\mathcal{Y} \mid x) \in \mathcal{H}$ defines a probability distribution over the output space $\mathcal{Y}$ and the hypothesis makes predictions by *sampling*, $y \sim \pi(\mathcal{Y} \mid x)$. Note that this definition also includes deterministic hypotheses, where the distributions assign probability 1 to a single y. For notational convenience, denote $\pi(\cdot \mid x)$ by $\pi(x)$, and the probability assigned to a particular y as $\pi(y \mid x)$. We will abuse notation slightly and use $(x, y) \sim \pi$ to refer to samples drawn from the joint distribution, $x \sim \Pr(\mathcal{X}), y \sim \pi(\mathcal{Y} \mid x)$. When it is evident from the context, we will drop $(x, y) \sim \pi$ and simply write π.

In interactive learning systems, we only observe feedback $\delta(x, y)$ for the y sampled from $\pi(x)$. In this work, feedback $\delta : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ is a cardinal loss that is only observed at the sampled data points. Small values for $\delta(x, y)$ indicate user satisfaction with y for x, while large values indicate dissatisfaction. The setting extends naturally to noisy losses, where we think of individual feedbacks $\delta \sim \Pr(\cdot \mid x, y)$ as conditioned on the input x and presented output y. All subsequent results hold with $\delta(x, y) := \mathbb{E}_{\delta \sim \Pr(\cdot|x,y)}[\delta]$.

The expected loss — called risk — of a hypothesis $R(\pi)$ is defined as,

$$R(\pi) = \mathbb{E}_{x \sim \Pr(\mathcal{X})} \mathbb{E}_{y \sim \pi(x)} \left[\delta(x, y) \right] = \mathbb{E}_{\pi} \left[\delta(x, y) \right].$$

The goal of the system is to minimize risk, or equivalently, maximize expected user satisfaction. The aim of learning is to find a hypothesis $\hat{\pi} \in \mathcal{H}$ that has minimum risk.

We wish to re-use the interaction logs of these systems for batch learning. Assume that its historical algorithm acted according to a *stationary* policy $\pi_0(x)$ (also called logging policy). The data collected from this system is

$$\mathcal{D} = \{(x_1, y_1, \delta_1), \ldots (x_n, y_n, \delta_n)\},$$

where $y_i \sim \pi_0(x_i)$ and $\delta_i \equiv \delta(x_i, y_i)$ (or $\delta_i \sim \Pr(\cdot \mid x_i, y_i)$ in the noisy feedback case).

Sampling Bias: $\mathcal{D}$ cannot be used to estimate $R(\pi)$ for a new hypothesis π using the estimator typically used in supervised learning. We ideally need either full information about $\delta(x_i, \cdot)$ or need samples $y \sim \pi(x_i)$ to directly estimate $R(\pi)$. This observation explains why, in practice, model selection over a small set of candidate systems is typically done via A/B tests, where the candidates are deployed to collect new data sampled according to $y \sim \pi(x)$ for each hypothesis π. A relative comparison of the assumptions, hypotheses, and principles used in supervised learning versus our learning setting is outlined in Table 6.1. Fundamentally, batch learning with bandit feedback is hard because $\mathcal{D}$ is both *biased* (predictions favored by the historical algorithm will be over-represented) and *incomplete* (feedback for other predictions will not be available) for learning.

	Supervised	Batch w/bandit
Distribution	$(x, y^*) \sim \Pr(\mathcal{X} \times \mathcal{Y})$	$x \sim \Pr(\mathcal{X}), y \sim \pi_0(x)$
Data $\mathcal{D}$	$\{x_i, {y^*}_i\}$	$\{x_i, y_i, \delta_i, p_i\}$
Hypothesis	$y = S(x)$	$y \sim \pi(\mathcal{Y} \mid x)$
Loss	$\Delta(y^*, \cdot)$ known	$\delta(x, \cdot)$ unknown
Objective: argmin	$\hat{R}(S) + C \cdot \mathrm{Reg}(S)$	$\hat{R}^M(\pi) + C \cdot \mathrm{Reg}(\pi) + \lambda \cdot \sqrt{\frac{\hat{Var}(\pi)}{n}}$

Table 6.1: Comparison of assumptions, hypotheses and learning principles for supervised learning and Batch Learning from Bandit Feedback (BLBF).

6.5 Learning Principle: Counterfactual Risk Minimization

The distribution mismatch between π_0 and any hypothesis $\pi \in \mathcal{H}$ can be addressed using importance sampling, which corrects the sampling bias [86] as:

$$R(\pi) = \mathbb{E}_\pi\left[\delta(x, y)\right] = \mathbb{E}_{\pi_0}\left[\delta(x, y)\frac{\pi(y \mid x)}{\pi_0(y \mid x)}\right].$$

This equation motivates the propensity scoring approach [95] in causal inference. During the operation of the logging policy, we keep track of the propensity, $p \equiv \pi_0(y \mid x_i)$ of π_0 to generate y. From these propensity-augmented logs

$$\mathcal{D} = \{(x_1, y_1, \delta_1, p_1), \ldots (x_n, y_n, \delta_n, p_n)\},$$

where $p_i \equiv \pi_0(y_i \mid x_i)$, we can derive an unbiased estimate of $R(\pi)$ via Monte Carlo approximation,

$$\hat{R}(\pi) = \frac{1}{n}\sum_{i=1}^{n} \delta_i \frac{\pi(y_i \mid x_i)}{p_i}. \tag{6.1}$$

At first thought, one may think that directly estimating $\hat{R}(\pi)$ over $\pi \in \mathcal{H}$ and picking the empirical minimizer is a valid learning strategy. Unfortunately, there are several pitfalls.

First, this strategy is not invariant to additive transformations of the loss and will give degenerate results if the loss is not appropriately scaled. In Section 6.5.3, we develop the intuition for why this is so and derive the optimal scaling of δ. For now, assume that $\forall x, \forall y, \delta(x, y) \in [-1, 0]$.

Second, this estimator has unbounded variance, since $p_i \simeq 0$ in $\mathcal{D}$ can cause $\hat{R}(\pi)$ to be arbitrarily far away from the true risk $R(\pi)$. This drawback can be fixed by "clipping" the importance sampling weights [51, 110, 12, 24]

$$\begin{aligned} R^M(\pi) &= \mathbb{E}_{\pi_0}\left[\delta(x, y) \min\left\{M, \frac{\pi(y \mid x)}{\pi_0 y \mid x}\right\}\right], \\ \hat{R}^M(\pi) &= \frac{1}{n}\sum_{i=1}^{n} \delta_i \min\left\{M, \frac{\pi(y_i \mid x_i)}{p_i}\right\}. \end{aligned} \tag{6.2}$$

$M \geq 1$ is a hyper-parameter chosen to trade-off bias and variance in the estimate, where smaller values of M induce larger bias in the estimate. Optimizing $\hat{R}^M(\pi)$ through exhaustive enumeration over $\mathcal{H}$ yields the Inverse Propensity Scoring (IPS) training objective

$$\hat{\pi}^{IPS} = \underset{\pi \in \mathcal{H}}{\operatorname{argmin}} \left\{\hat{R}^M(\pi)\right\}. \tag{6.3}$$

This objective captures the essence of previous offline policy optimization approaches [12, 110]. These approaches differ from Equation (6.3) in the specific way the importance sampling weights are clipped, the choice of $\mathcal{H}$ and frame the optimization problem as a maximization of counterfactual rewards as opposed to minimization of counterfactual risk.

Third, importance sampling typically estimates $\hat{R}^M(\pi)$ of different hypotheses $\pi \in \mathcal{H}$ with vastly different variances. Consider two hypotheses π_1 and π_2, where π_1 is very similar to π_0, but where π_2 samples predictions y that were not well explored by π_0. Importance sampling gives us low-variance estimates for $\hat{R}^M(\pi_1)$, but highly variable estimates for $\hat{R}^M(\pi_2)$. Intuitively, if we can develop variance-sensitive confidence bounds over the hypothesis space, optimizing a conservative confidence bound should find a π whose $R(\pi)$ will not be much worse, with high probability.

6.5.1 Generalization Error Bound

A standard analysis [124] would give a bound that is agnostic to the variance introduced by importance sampling. Following our intuition above, we derive a higher order bound that includes the variance term using empirical Bernstein bounds [82]. To develop such a generalization error bound, we first need a concept of capacity for stochastic hypothesis classes. Our strategy is to define an auxiliary deterministic function class $\mathcal{F}_{\mathcal{H}}$ for $\mathcal{H}$ and directly use covering numbers for $\mathcal{F}_{\mathcal{H}}$ conditioned on sample $\mathcal{D}$ to characterize the capacity of $\mathcal{H}$. We start by defining the auxiliary deterministic function class $\mathcal{F}_{\mathcal{H}}$.

Definition 3. *For any stochastic class $\mathcal{H}$, define an auxiliary function class $\mathcal{F}_{\mathcal{H}} = \{f_\pi : \mathcal{X} \times \mathcal{Y} \mapsto [0,1]\}$. Each $\pi \in \mathcal{H}$ corresponds to a function $f_\pi \in \mathcal{F}_{\mathcal{H}}$,*

$$f_\pi(x,y) = 1 + \frac{\delta(x,y)}{M} \min\left\{M, \frac{\pi(y \mid x)}{\pi_0 y \mid x}\right\}. \tag{6.4}$$

Based on this auxiliary function class $\mathcal{F}_{\mathcal{H}}$, we will study the convergence of $\hat{R}^M(\pi) \to R^M(\pi)$. A key insight is the following relationship between π and f_π.

Lemma 7. *For any stochastic hypothesis π, the clipped risk $R^M(\pi)$ and the expected value of f_π under the data generating distribution are related as*

$$\mathbb{E}_{\pi_0}\left[f_\pi(x,y)\right] = 1 + \frac{R^M(\pi)}{M}. \tag{6.5}$$

Proof. Note that f_π is a deterministic and bounded function. From the definition of f_π and by linearity of expectation,

$$\begin{aligned}\mathbb{E}_{\pi_0}\left[f_\pi(x,y)\right] &= \mathbb{E}_{\pi_0}\left[1 + \frac{\delta(x,y)}{M}\min\left\{M, \frac{\pi(y \mid x)}{\pi_0 y \mid x}\right\}\right] \\ &= 1 + \frac{1}{M}\mathbb{E}_{\pi_0}\left[\delta(x,y)\min\left\{M, \frac{\pi(y \mid x)}{\pi_0 y \mid x}\right\}\right] \\ &= 1 + \frac{R^M(\pi)}{M}.\end{aligned}$$

□

As a consequence of Lemma 7, we can use classic notions of capacity for $\mathcal{F}_\mathcal{H}$ to reason about the convergence of $\hat{R}^M(\pi) \rightarrow R^M(\pi)$. Recall the covering number $\mathcal{N}_\infty(\epsilon, \mathcal{F}, n)$ for a function class $\mathcal{F}$.[1] Define an ϵ–cover $\mathcal{N}(\epsilon, A, \|\cdot\|_\infty)$ for a set $A \subseteq \mathbb{R}^n$ to be the size of the smallest cardinality subset $A_0 \subseteq A$ such that A is contained in the union of balls of radius ϵ centered at points in A_0, in the metric induced by $\|\cdot\|_\infty$. The covering number is,

$$\mathcal{N}_\infty(\epsilon, \mathcal{F}, n) = \sup_{(x_i,y_i)\in(\mathcal{X}\times\mathcal{Y})^n} \mathcal{N}(\epsilon, \mathcal{F}(\{(x_i,y_i)\}), \|\cdot\|_\infty),$$

where $\mathcal{F}(\{(x_i,y_i)\})$ is the function class conditioned on sample $\{(x_i,y_i)\}$,

$$\mathcal{F}(\{(x_i,y_i)\}) = \{(f(x_1,y_1), \ldots f(x_n,y_n)) : f \in \mathcal{F}\}.$$

Our measure of the capacity of our stochastic class $\mathcal{H}$ to "fit" a sample of size n shall be $\mathcal{N}_\infty(\frac{1}{n}, \mathcal{F}_\mathcal{H}, 2n)$.

For a compact notation, define the random variable $z_\pi \equiv \delta(x,y)\min\left\{M, \frac{\pi(y|x)}{\pi_0(y|x)}\right\}$ with mean $\bar{z}_\pi = R^M(\pi)$. The sample $\mathcal{D}$ contains n i.i.d. random variables ${z_\pi}^i \equiv$

[1]Recent texts [1, 82] and their references provide an excellent overview of covering numbers.

$\delta_i \min\{M, \frac{\pi(y_i|x_i)}{p_i}\}$. Define the sample mean and variance of ${z_\pi}^i$

$$\hat{\bar{z}}_\pi \equiv \frac{1}{n}\sum_{i=1}^{n} {z_\pi}^i = \hat{R}^M(\pi),$$
$$\hat{\boldsymbol{Var}}(z_\pi) \equiv \frac{1}{n-1}\sum_{i=1}^{n}({z_\pi}^i - \hat{\bar{z}}_\pi)^2.$$

Theorem 8. *With probability at least* $1-\eta$ *in the random vector* $(x_1, y_1)\dots(x_n, y_n) \overset{i.i.d.}{\sim} \pi_0$*, with observed losses* $\delta_1, \dots \delta_n$*, for* $n \geq 16$ *and a stochastic hypothesis space* $\mathcal{H}$ *with capacity* $\mathcal{N}_\infty(\frac{1}{n}, \mathcal{F}_\mathcal{H}, 2n)$*,*

$$\forall \pi \in \mathcal{H} : R(\pi) \leq \hat{R}^M(\pi) + \sqrt{18\frac{\hat{\boldsymbol{Var}}(z_\pi)\boldsymbol{Q}_\mathcal{H}(n,\eta)}{n}} + M\frac{15\boldsymbol{Q}_\mathcal{H}(n,\eta)}{n-1},$$

$$\text{where } \boldsymbol{Q}_\mathcal{H}(n,\eta) \equiv \log(\frac{10 \cdot \mathcal{N}_\infty(\frac{1}{n}, \mathcal{F}_\mathcal{H}, 2n)}{\eta}), \quad 0 < \eta < 1.$$

The proof of Theorem 8 is provided in Appendix C.1.

6.5.2 CRM Principle

The generalization error bound from Theorem 8 is constructive in the sense that it motivates a general principle for designing machine learning methods for batch learning from bandit feedback. In particular, a learning algorithm following this principle should optimize the estimate $\hat{R}^M(\pi)$ as well as the empirical standard deviation, where the latter serves as a *data-dependent regularizer*.

$$\hat{pi}^{CRM} = \underset{\pi \in \mathcal{H}}{\operatorname{argmin}} \left\{ \hat{R}^M(\pi) + \lambda \sqrt{\frac{\hat{\boldsymbol{Var}}(z_\pi)}{n}} \right\}. \tag{6.6}$$

$M \geq 1$ and $\lambda \geq 0$ are regularization hyper-parameters. When $\lambda = 0$, we recover the Inverse Propensity Scoring objective of Equation (6.3). In analogy to Structural Risk Minimization [124], we call this principle *Counterfactual Risk Minimization*, since both pick the hypothesis with the tightest upper bound on the true risk $R(\pi)$.

6.5.3 Conservative Loss Scaling

When performing supervised learning with true labels y^* and a loss function $\Delta(y^*, \cdot)$, empirical risk minimization using the standard estimator is invariant to additive translation and multiplicative (by a positive constant) scaling of Δ. Bandit learning with the risk estimators $\hat{R}(\pi)$ and $\hat{R}^M(\pi)$, however, crucially requires $\delta(\cdot, \cdot) \in [-1, 0]$.

Consider, for example, the case of $\delta(\cdot, \cdot) \geq 0$. The training objectives in Equation (6.3) (IPS) and Equation (6.6) (CRM) become degenerate! A hypothesis $\pi \in \mathcal{H}$ that completely avoids the sample $\mathcal{D}$ (i.e. $\forall i = 1, \dots n, \pi(y_i \mid x_i) = 0$) trivially achieves the best possible $\hat{R}^M(\pi)$ $(= 0)$ with 0 empirical variance. This degeneracy arises partially because when $\delta(\cdot, \cdot) \geq 0$, the objectives optimize a *lower* bound on $R(\pi)$, whereas what we need is an *upper* bound.

For any bounded loss $\delta(\cdot, \cdot) \in [\triangledown, \vartriangle]$, we have, $\forall x$

$$\mathbb{E}_{y \sim \pi(x)} \left[\delta(x, y)\right] \leq \vartriangle + \mathbb{E}_{y \sim \pi_0 x} \left[(\delta(x, y) - \vartriangle) \min \left\{ M, \frac{\pi(y \mid x)}{\pi_0(y \mid x)} \right\} \right].$$

Since the optimization objectives in Equation (6.3) and Equation (6.6) are unaffected by a constant positive scale factor (e.g., $\vartriangle - \triangledown$), we should transform $\delta \mapsto \delta'$ to derive a conservative training objective,

$$\delta' \equiv \{\delta - \vartriangle\} / \{\vartriangle - \triangledown\}.$$

Such a transformation captures the following assumption: for an input $x \in \mathcal{D}$, if a new hypothesis $\pi \neq \pi_0$ samples an unexplored y not seen in $\mathcal{D}$, in the worst case it will incur a loss of $\triangle$. This assumption is clearly very conservative, and we foresee future work that relaxes this using additional assumptions about $\delta(\cdot, \cdot)$ and $\mathcal{Y}$.

6.5.4 Selecting Hyper-Parameters

We propose selecting the hyper-parameters $M \geq 1$ and $\lambda \geq 0$ via cross validation. However, we must be careful not to set M too small or λ too big. The estimated risk $\hat{R}^M(\pi) \in [-M, 0]$, while the variance penalty $\sqrt{\frac{\hat{Var}(z_\pi)}{n}} \in \left[0, \frac{M}{2\sqrt{n}}\right]$. If M is too small, all the importance sampling weights will be clipped, and all hypotheses will have the same biased estimate of risk $M\hat{R}^M(\pi_0)$. Similarly, if $\lambda \gg 0$, a hypothesis $\pi \in \mathcal{H}$ that completely avoids $\mathcal{D}$ (i.e. $\forall i = 1, \ldots n, \pi(y_i \mid x_i) = 0$) has $\hat{R}^M(\pi)$ $(= 0)$ with 0 empirical variance. So, it will achieve the best possible training objective of 0. As a rule of thumb, we can calibrate M and λ so that, for some $\pi \in \mathcal{H}$, the estimator is unbiased, and the objective is non-trivially negative. When $\pi_0 \in \mathcal{H}$, $M \simeq \max\{p_i\}/\min\{p_i\}$ and $\lambda : \left\{\hat{R}^M(\pi_0) + \lambda\sqrt{\frac{\hat{Var}(z_{\pi_0})}{n}}\right\} < 0$ are natural choices.

6.5.5 When Is Counterfactual Learning Possible?

The bounds in Theorem 8 are with respect to the randomness in π_0. Known impossibility results for counterfactual evaluation using π_0 [65] also apply to counterfactual learning. In particular, if π_0 was deterministic, or even stochastic but without full support over $\mathcal{Y}$, it is easy to engineer examples involving the unexplored $y \in \mathcal{Y}$ that guarantee sub-optimal learning even as $|\mathcal{D}| \to \infty$. Similarly, lower bounds for learning under covariate shift [24] also apply to counterfactual learning. Finally, a stochastic π_0 with heavier tails need not always allow more effective learning. From importance sampling theory [86], what matters is how well π_0 explores the regions of $\mathcal{Y}$ with favorable losses.

6.6 Learning Algorithm: POEM

We now use the CRM principle to derive an efficient algorithm for structured output prediction using linear rules. Classic learning methods for structured output prediction based on full-information feedback, like structured support vector machines [122] and conditional random fields [64], predict using

$$S_{\mathbf{w}}^{sup}(x) = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \left\{ \mathbf{w}^T \mathbf{f}(x, y) \right\}, \tag{6.7}$$

where $\mathbf{w}$ is a d-dimensional weight vector, and $\mathbf{f}(x, y)$ is a d-dimensional joint feature map. For example, in multi-label document classification, for a news article x and a possible assignment of labels y represented as a bit vector, $\mathbf{f}(x, y)$ could simply be a concatenation $\overline{x} \otimes y$ of the bag-of-words features of the document ($\overline{x}$), one copy for each of the assigned labels in y. Several efficient inference algorithms have been developed to solve Equation (6.7).

The POEM algorithm we derive uses the same parameterization of the hypothesis space as in Equation (6.7). However, it considers the following expanded class of Stochastic Soft-max Rules based on this parameterization, which contains the deterministic rule in Equation (6.7) as a limiting case.

6.6.1 Stochastic Soft-max Rules

Consider the following stochastic family $\mathcal{H}_{lin}$, parametrized by $\mathbf{w}$. A hypothesis $\pi_{\mathbf{w}}(x) \in \mathcal{H}_{lin}$ samples y from the distribution

$$\pi_{\mathbf{w}}(y \mid x) = \exp(\mathbf{w}^T \mathbf{f}(x, y))/\mathbb{Z}(x).$$

$\mathbb{Z}(x) = \sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^T \mathbf{f}(x, y'))$ is the partition function. This distribution is the "soft-max" variant of the "hard-max" rules from Equation (6.7). Additionally, for a *temperature* multiplier $\alpha > 1, \mathbf{w} \mapsto \alpha\mathbf{w}$ induces a more "peaked" distribution $\pi_{\alpha\mathbf{w}}$ that preserves the modes of $\pi_{\mathbf{w}}$, and is a "more deterministic" variant of $\pi_{\mathbf{w}}$.

$\pi_{\mathbf{w}}$ lies in the exponential family of distributions, and has a simple gradient,

$$\nabla\pi_{\mathbf{w}}(y \mid x) = \pi_{\mathbf{w}}(y \mid x)\left\{\mathbf{f}(x, y) - \mathbb{E}_{y' \sim \pi_{\mathbf{w}}(x)}\left[\mathbf{f}(x, y')\right]\right\}.$$

This observation allows us to implement the CRM principle of Equation (6.6) by using gradient descent to search through $\mathcal{H}_{lin}$ tractably. This search procedure is the core of the Policy Optimizer for Exponential Models (POEM) algorithm.

6.6.2 POEM Training Objective

Consider a bandit structured output data set $\mathcal{D} = \{(x_1, y_1, \delta_1, p_1), \ldots (x_n, y_n, \delta_n, p_n)\}$. In multi-label document classification, this data could be collected from an interactive labeling system, where each y indicates the labels predicted by the system for a document x. The feedback $\delta(x, y)$ could be how many labels (but not which ones) were correct. To perform learning, first we scale the losses as outlined in Section 6.5.3. Next, instantiating the CRM principle of Equation (6.6) for $\mathcal{H}_{lin}$, (using notation analogous to that in Theorem 8, adapted for $\mathcal{H}_{lin}$), yields the POEM training objective.

$$\hat{\mathbf{w}} = \operatorname*{argmin}_{\mathbf{w} \in \mathbb{R}^d} \hat{\bar{z}}_{\mathbf{w}} + \lambda \sqrt{\frac{\hat{Var}(z_{\mathbf{w}})}{n}}, \tag{6.8}$$

$$\text{with} \qquad {z_{\mathbf{w}}}^i \equiv \delta_i \min\{M, \frac{\exp(\mathbf{w}^T \mathbf{f}(x_i, y_i))}{p_i \cdot \mathbb{Z}(x_i)}\},$$

$$\hat{\bar{z}}_{\mathbf{w}} \equiv \frac{1}{n} \sum_{i=1}^{n} {z_{\mathbf{w}}}^i,$$

$$\hat{Var}(z_{\mathbf{w}}) \equiv \frac{1}{n-1} \sum_{i=1}^{n} ({z_{\mathbf{w}}}^i - \hat{\bar{z}}_{\mathbf{w}})^2.$$

While the objective in Equation (6.8) is not convex in $\mathbf{w}$ (even for $\lambda = 0$), we find that batch and stochastic gradient descent find $\hat{\mathbf{w}}$ that have good generalization error (e.g., L-BFGS [16] out of the box). The key subroutine that enables us to perform efficient gradient descent is a tractable way to compute ${z_{\mathbf{w}}}^i$ and $\nabla_{\mathbf{w}}({z_{\mathbf{w}}}^i)$ — both depend on $\mathbb{Z}(x_i)$.

$$\nabla_{\mathbf{w}}({z_{\mathbf{w}}}^i) = \begin{cases} 0 & \text{if } \frac{\exp(\mathbf{w}^T \mathbf{f}(x_i, y_i))}{p_i \cdot \mathbb{Z}(x_i)} \geq M \\ {z_{\mathbf{w}}}^i \left\{ \mathbf{f}(x_i, y_i) - \sum_{y'} \left[\mathbf{f}(x_i, y') \frac{\exp(\mathbf{w}^T \mathbf{f}(x_i, y'))}{\mathbb{Z}(x_i)} \right] \right\} & \text{otherwise.} \end{cases} \tag{6.9}$$

For the special case when $\mathbf{f}(x, y) = \bar{x} \otimes y$, where y is a bit vector $\in \{0, 1\}^\ell$, $\mathbb{Z}(x)$ has a simple decomposition.

$$\exp(\mathbf{w}^T \mathbf{f}(x, y)) = \prod_{j=1}^{\ell} \exp(y_j \mathbf{w}_j^T x),$$

$$\mathbb{Z}(x) = \prod_{j=1}^{\ell} (1 + \exp(\mathbf{w}_j^T x)),$$

where ℓ is the length of the bit vector representation of y. For the general case, several approximation schemes have been developed to handle $\mathbb{Z}(x)$ for supervised training of graphical models and we can directly co-opt these for batch learning under bandit feedback as well.

6.6.3 POEM Iterated Variance Majorization Algorithm

We could use standard batch gradient descent methods to minimize the POEM training objective. In particular, prior work [134, 68] has established theoretically sound modifications to L-BFGS for non-smooth non-convex optimization. The following develops a stochastic method that can be much faster.

At first glance, the POEM training objective in Equation (6.8), specifically the variance term resists stochastic gradient optimization in the presented form. To remove this obstacle, we now develop a Majorization-Minimization scheme, similar in spirit to recent approaches to multi-class SVMs [123] that can be shown to converge to a local optimum of the POEM training objective. In particular, we will show how to decompose $\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}})}$ as a sum of differentiable functions (e.g., $\sum_i {z_{\mathbf{w}}}^i$ or $\sum_i \{{z_{\mathbf{w}}}^i\}^2$) so that we can optimize the overall training objective at scale using stochastic gradient descent.

Proposition 3. *For any* $\mathbf{w}_0$ *such that* $\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0}) > 0$*,*

$$\begin{aligned}
\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}})} &\leq A_{\mathbf{w}_0} \sum_{i=1}^{n} {z_{\mathbf{w}}}^i + B_{\mathbf{w}_0} \sum_{i=1}^{n} \{{z_{\mathbf{w}}}^i\}^2 + C_{\mathbf{w}_0} \\
&= G(\mathbf{w}; \mathbf{w}_0). \\
A_{\mathbf{w}_0} &\equiv \frac{-\hat{\bar{z}}_{\mathbf{w}_0}}{(n-1)\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}, \\
B_{\mathbf{w}_0} &\equiv \frac{1}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}, \\
C_{\mathbf{w}_0} &\equiv \frac{n\{\hat{\bar{z}}_{\mathbf{w}_0}\}^2}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}} + \frac{\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}{2}.
\end{aligned}$$

Proposition 3 is proved in Appendix C.2.

Iteratively minimizing $\mathbf{w}^{t+1} = \text{argmin}_{\mathbf{w}} G(\mathbf{w}; \mathbf{w}^t)$ ensures that the sequence of iterates $\mathbf{w}^1, \dots \mathbf{w}^{t+1}$ are successive minimizers of $\sqrt{\hat{Var}(z_{\mathbf{w}})}$. Hence, during an epoch t, POEM proceeds by sampling uniformly $(x_i, y_i, \delta_i, p_i) \sim \mathcal{D}$, computing ${z_{\mathbf{w}}}^i, \nabla {z_{\mathbf{w}}}^i$ and, for learning rate η, updating

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\{\nabla {z_{\mathbf{w}}}^i + \lambda \sqrt{n}(A_{\mathbf{w}^t} \nabla {z_{\mathbf{w}}}^i + 2B_{\mathbf{w}^t} {z_{\mathbf{w}}}^i \nabla {z_{\mathbf{w}}}^i)\}.$$

After each epoch, $\mathbf{w}^{t+1} \leftarrow w$, and iterated minimization proceeds until convergence (e.g. by monitoring performance on a validation set using Equation (6.1)).

The full algorithm is summarized as Algorithm 1. Software implementing POEM is available at `http://www.cs.cornell.edu/~adith/POEM/` for download, as is all the code and data needed to run each of the experiments reported in Section 6.7. Standalone implementations of POEM tuned for scalable performance for bandit multi-class problems (and wrappers to handle several other problem types) is also available at `http://www.cs.cornell.edu/~adith/Criteo/`.

6.7 Empirical Evaluation

We now empirically evaluate the prediction performance and computational efficiency of POEM on a broad range of scenarios. To be able to control these experiments effectively, we derive bandit feedback from existing full-information data sets. As the learning task, we consider multi-label classification with input $x \in \mathbb{R}^p$ and prediction $y \in \{0, 1\}^\ell$. Popular supervised algorithms that solve this problem include Structured SVMs [122] and Conditional Random Fields [64]. In the simplest case, CRF essentially performs logistic regression for each of the ℓ labels independently. As outlined in Section 6.6, we use a joint feature

Algorithm 1 POEM pseudocode. An alternative version can use separate samplers for estimating ${z_{\mathbf{w}}}^i$ and $\{{z_{\mathbf{w}}}^i\}^2$ on Line 24.

```
 1: procedure LossGradient($\mathcal{D}_s$, $\mathbf{w}$)          ▹ Returns ${z_{\mathbf{w}}}^i, \nabla_{\mathbf{w}}({z_{\mathbf{w}}}^i)$ for $i \in \mathcal{D}_s$
 2:     for $(x_i, y_i, \delta_i, p_i) \in \mathcal{D}_s$ do
 3:         $z^i \leftarrow {z_{\mathbf{w}}}^i$.
 4:         $g^i \leftarrow \nabla_{\mathbf{w}}({z_{\mathbf{w}}}^i)$.                    ▹ Equation (6.9)
        return $\vec{z}, \vec{g}$.
 5: procedure ABC($\mathcal{D}$, $\mathbf{w}$, $\lambda$)          ▹ Returns $A_{\mathbf{w}}, B_{\mathbf{w}}, C_{\mathbf{w}}$ from Proposition (3)
 6:     $\vec{z}, \vec{g} \leftarrow LossGradient(\mathcal{D}, \mathbf{w})$.
 7:     $R \leftarrow \sum_i z_i/n$.
 8:     $V \leftarrow \sqrt{\sum_i (z_i - R)^2/(n-1)}$.
 9:     $A \leftarrow 1 - \frac{\lambda\sqrt{n}R}{(n-1)V}$.
10:     $B \leftarrow \frac{\lambda}{2(n-1)V\sqrt{n}}$.
11:     $C \leftarrow \frac{\lambda V}{2\sqrt{n}} + \frac{\lambda\sqrt{n}R^2}{2(n-1)V}$.
        return $A, B, C$.
12: procedure SGD($\mathcal{D}$, $\lambda$, $\mu$)                  ▹ $L2$ regularizer $\mu$
13:     $\mathbf{w} \leftarrow [0]_d$.                              ▹ Initial param
14:     $\mathbf{h} \leftarrow [1]_d$.                              ▹ AdaGrad history
15:     while True do
16:         Shuffle $\mathcal{D}$.
17:         $A, B, C \leftarrow ABC(\mathcal{D}, \mathbf{w}, \lambda)$.
18:         for $\mathcal{D}_s \subset \mathcal{D}$ do                  ▹ Mini-batch $|\mathcal{D}_s| = b$
19:             $\vec{z}, \vec{g} \leftarrow LossGradient(\mathcal{D}_s, \mathbf{w})$.
20:             $\bar{z} = \sum_i z_i/b$.
21:             $\bar{g} = \sum_i g_i/b$.
22:             $h_i \leftarrow h_i + {\bar{g}_i}^2$.
23:             $j_i \leftarrow \bar{g}_i/\sqrt{h_i}$.
24:             $\vec{\nabla} \leftarrow A\vec{j} + 2\mu\mathbf{w} + 2B\bar{z}\vec{j}$.
25:             if $||\vec{\nabla}|| \simeq 0$ then return $\mathbf{w}$.       ▹ Gradient norm convergence
26:             if $\bar{z} >$ avg. validation $\bar{z}$ then return $\mathbf{w}$.   ▹ Progressive validation
27:             $\mathbf{w} \leftarrow \mathbf{w} - \eta\vec{\nabla}$.                ▹ Step size $\eta$
```

map: $\mathbf{f}(x, y) = x \otimes y$. We conducted experiments on different multi-label datasets collected from the LibSVM repository[2], with different ranges for p (features), ℓ (labels) and n (samples) represented as summarized in Table 6.2.

[2]http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html

Name	p(# features)	ℓ(# labels)	n_{train}	n_{test}
Scene	294	6	1211	1196
Yeast	103	14	1500	917
TMC	30438	22	21519	7077
LYRL	47236	4	23149	781265

Table 6.2: Corpus statistics for different multi-label datasets from the LibSVM repository. LYRL was post-processed so that only top-level categories were treated as labels.

Experiment Methodology: We employ the Supervised $\mapsto$ Bandit conversion [66, 8] method. Here, we take a supervised data set

$$\mathcal{D}^* = \{(x_1, {y^*}_1) \ldots (x_n, {y^*}_n)\},$$

and simulate a bandit feedback data set from a logging policy π_0 by sampling $y_i \sim \pi_0 x_i$ and collecting feedback $\delta_i \equiv \Delta({y^*}_i, y_i)$. Feedback for all other possible actions $\Delta({y^*}_i, \cdot)$ is withheld. In principle, we could use any arbitrary stochastic policy as π_0. We choose a CRF trained on 5% of $\mathcal{D}^*$ as π_0 using default hyper-parameters since they provide probability distributions amenable to sampling. In all the multi-label experiments, $\Delta(y^*, y)$ is the Hamming loss between the supervised label y^* versus the sampled label y for input x. Hamming loss is just the number of incorrectly assigned labels (adding both false positives and false negatives). To create bandit feedback $\mathcal{D} = \{(x_i, y_i, \delta_i, p_i \equiv \pi_0(y_i \mid x_i))\}$, we take four passes through $\mathcal{D}^*$ and sample labels from π_0. Note that each supervised label is worth $\simeq |\mathcal{Y}| = 2^\ell$ bandit feedback labels. We can explore different learning strategies (e.g., IPS, CRM, etc.) on $\mathcal{D}$ and obtain learned weight vectors $\mathbf{w}^{ips}, \mathbf{w}^{crm}$, etc. On the supervised test set, we then report the expected loss per instance $\mathcal{R} = \frac{1}{n_{test}} \sum_i \mathbb{E}_{y \sim \pi_{\mathbf{w}}(x_i)} \Delta({y^*}_i, y)$ and compare the generalization error ($\mathcal{R}$) of these learning strategies.

Baselines and Learning Methods: The expected Hamming loss of π_0 is the baseline to beat. The lower the loss, the better it is. The naïve, variance-agnostic approach to counterfactual learning [12, 110] can be generalized to handle parametric multi-label classification by optimizing Equation (6.8) with $\lambda = 0$. We optimize it either using L-BFGS (IPS($\mathcal{B}$)) or stochastic optimization (IPS($\mathcal{S}$)). POEM($\mathcal{S}$) uses our Iterative-Majorization approach to variance regularization as outlined in Section 6.6.3, while POEM($\mathcal{B}$) is an L-BFGS variant. Finally, we report results from a supervised CRF trained on all of $\mathcal{D}^*$ as a skyline, despite its unfair advantage of having access to the full-information examples.

We keep aside 25% of $\mathcal{D}$ as a validation set — we use the unbiased counterfactual estimator from Equation (6.1) for selecting hyper-parameters. $\lambda = c\lambda^*$, where λ^* is the calibration factor from Section 6.5.4 and $c \in \{10^{-6}, \dots 1\}$ in multiples of 10. The clipping constant M is similarly set to the ratio of the 90*%ile* to the 10*%ile* propensity score observed in the training set of $\mathcal{D}$. The reported results are not sensitive to this choice of M, any reasonably large clipping constant suffices (e.g. even a simple, problem independent choice of $M = 100$ works well). When optimizing any objective over $\mathbf{w}$, we always begin the optimization from $\mathbf{w} = 0$, which is equivalent to $\pi_{\mathbf{w}} =$ Uniform($\mathcal{Y}$). We use mini-batch AdaGrad [29] with batch size $b = 100$ and step size $\eta = 1$ to adapt our learning rates for the stochastic approaches and use progressive validation [10] and gradient norms to detect convergence. Finally, the entire experiment set-up is run 10 times (i.e. π_0 trained on randomly chosen 5% subsets, $\mathcal{D}$ re-created, and test set performance of different approaches collected) and we report the averaged test set expected error across runs.

6.7.1 Does Variance Regularization Improve Generalization?

$\mathcal{R}$	Scene	Yeast	TMC	LYRL
π_0	1.543	5.547	3.445	1.463
IPS($\mathcal{B}$)	1.193	4.635	2.808	0.921
POEM($\mathcal{B}$)	1.168	4.480	2.197	0.918
IPS($\mathcal{S}$)	1.519	4.614	3.023	1.118
POEM($\mathcal{S}$)	1.143	4.517	2.522	0.996
CRF	0.659	2.822	1.189	0.222

Table 6.3: Test set Hamming loss, $\mathcal{R}$ for different approaches to multi-label classification on different data sets, averaged over 10 runs. POEM is significantly better than IPS on each data set and each optimization variant (one-tailed paired difference t-test at a significance level of 0.05).

Results are reported in Table 6.3. We statistically test the performance of POEM against IPS (batch variants are paired together, and the stochastic variants are paired together) using a one-tailed paired difference t-test at a significance level of 0.05 across 10 runs of the experiment, and find POEM to be significantly better than IPS on each data set and each optimization variant. On all datasets, POEM learns a model that is substantially better than π_0. This outcome suggests that the CRM principle is practically useful for designing learning algorithms, and that variance regularization is indeed beneficial.

6.7.2 How Computationally Efficient Is POEM?

Table 6.4 shows the time taken (in CPU seconds) to run each method on each data set, averaged over different validation runs when performing hyper-parameter grid search. Some of the timing results are skewed by outliers, e.g., when under very weak regularization, CRFs tend to take longer to converge. However, it is still clear that the stochastic variants are able to recover good pa-

Time(s)	Scene	Yeast	TMC	LYRL
IPS($\mathcal{B}$)	2.58	47.61	136.34	21.01
IPS($\mathcal{S}$)	1.65	2.86	49.12	13.66
POEM($\mathcal{B}$)	75.20	94.16	949.95	561.12
POEM($\mathcal{S}$)	4.71	5.02	276.13	120.09
CRF	4.86	3.28	99.18	62.93

Table 6.4: Average time in seconds for each validation run for different approaches to multi-label classification. CRF is implemented by scikit-learn [87]. On all data sets, stochastic approaches are much faster than batch gradients.

rameter settings in a fraction of the time of batch L-BFGS optimization, and this is even more pronounced when the number of labels grows — the run-time in such problem instances is dominated by computation of $\mathbb{Z}(x)$.

6.7.3 Can Deterministic Predictions from Learned Stochastic Policies Generalize Well?

For the policies learned by POEM as shown in Table 6.3, Table 6.5 reports the averaged performance of the deterministic predictor derived from them. For a learned weight vector $\mathbf{w}$, this simply amounts to applying Equation (6.7). In practice, this method of generating predictions can be substantially faster than sampling since computing the argmax does not require computation of the partition function $\mathbb{Z}(x)$ which can be expensive in structured output prediction. From Table 6.5, we see that the loss of the deterministic predictor is typically not far from the loss of the stochastic policy, and often better.

$\mathcal{R}$	Scene	Yeast	TMC	LYRL
POEM($\mathcal{S}$)	1.143	4.517	2.522	0.996
POEM($\mathcal{S}$)$_{map}$	1.143	4.065	2.299	0.880

Table 6.5: Mean Hamming loss of MAP predictions from the policies learned in Table 6.3. POEM$_{map}$ is significantly better than POEM on all data sets except Scene (one-sided paired difference t-test, significance level 0.05).

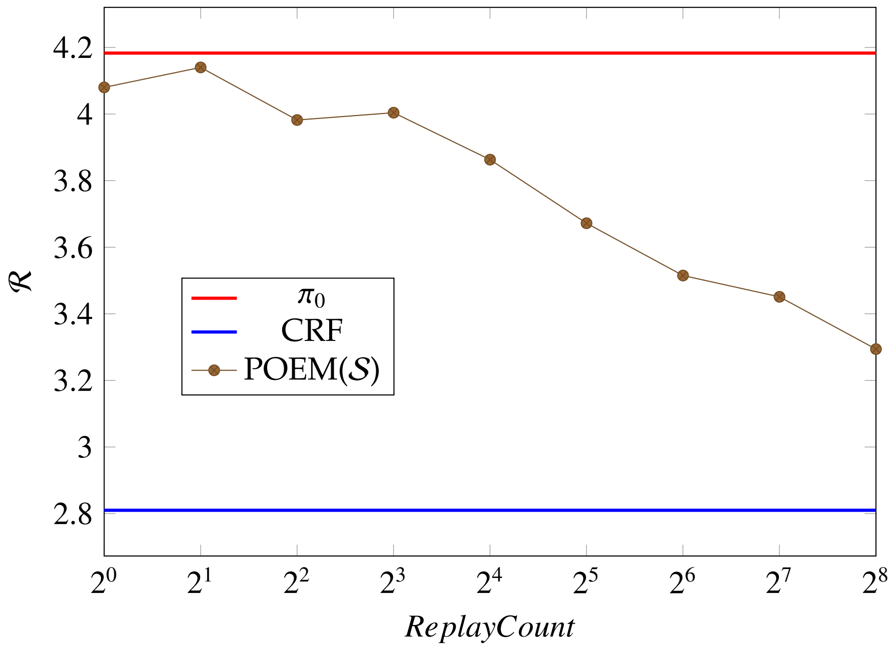


Figure 6.1: Generalization performance of POEM($\mathcal{S}$) as a function of n on the Yeast dataset.

6.7.4 How Does Generalization Improve With Dataset Size?

As we collect more data under π_0, our generalization error bound indicates that prediction performance should eventually approach that of the optimal model in the hypothesis space. We can simulate $n \to \infty$ by replaying the training data multiple times, collecting samples $y \sim \pi_(x)$. In the limit, we would observe every possible y in the bandit feedback data set, since $\pi_0(x)$ has a non-zero probability of exploring each prediction y. However, the learning rate may be slow, since the exponential model family has very thin tails, and hence may not be an ideal logging distribution to explore well. Holding all other details of the experiment setup fixed, we vary the number of times we replayed the training set (*ReplayCount*) to collect samples from π_0 and report the performance of POEM($\mathcal{S}$) on the Yeast dataset in Figure 6.1. As expected, the performance of POEM improves with increasing sample size. Note that even with $ReplayCount = 2^8$, POEM($\mathcal{S}$) is learning from much less information than the CRF, where each supervised label conveys 2^{14} bandit label feedbacks.

6.7.5 How Does Quality of the Logging Policy Affect Learning?

In this experiment, we change the fraction of the training set $\beta \cdot n_{train}$ that was used to train the logging policy — and as β is increased, the quality of π_0 improves. Intuitively, there's a trade-off: better π_0 probably samples correct predictions more often and so produces a higher quality $\mathcal{D}$ to learn from, but it should also be harder to beat π_0.

We vary β from 1% to 100% while keeping all other conditions identical to the original experiment setup in Figure 6.2. Note that results in other experiments correspond to $\beta = 5\%$. We find that POEM($\mathcal{S}$) is able to find a hypothesis at least as good as π_0 consistently. Moreover, even $\mathcal{D}$ collected from a poor quality π_0 ($0.5 \leq \beta \leq 0.2$) allows POEM($\mathcal{S}$) to learn an improved policy efficiently.

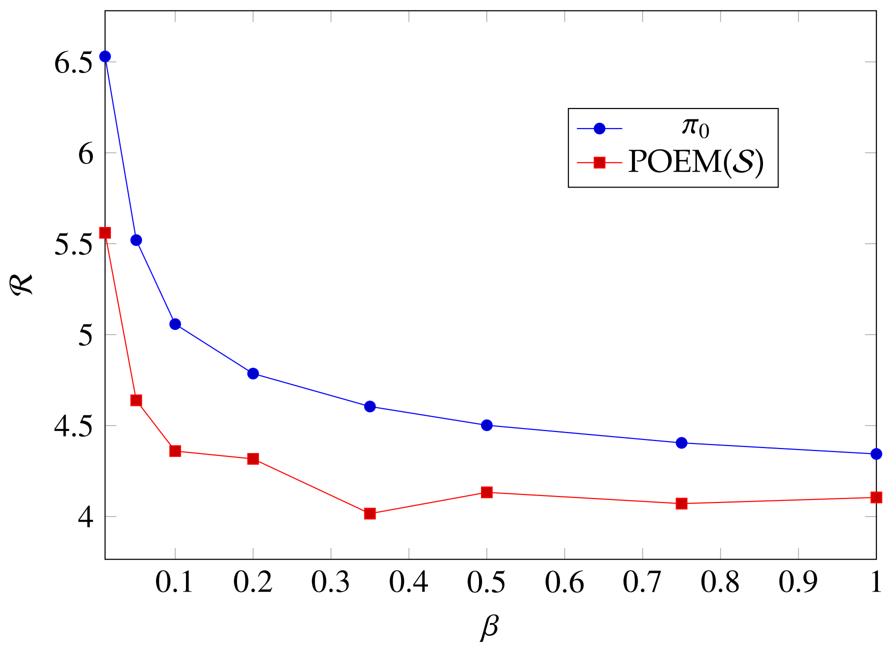


Figure 6.2: The performance of POEM($\mathcal{S}$) on the Yeast dataset as π_0 is improved. The fraction β of the supervised training set used to train π_0 is varied to control the quality of π_0. π_0 performance does not reach CRF when $\beta = 1$ because we do not tune hyper-parameters, and we report its expected loss, not the loss of its Maximum A Posteriori prediction.

6.7.6 How Does Stochasticity of the Logging Policy Affect Learning?

Finally, the theory suggests that counterfactual learning is only possible when π_0 is sufficiently stochastic (the generalization bounds of Theorem 8 hold with high probability in the samples drawn from π_0). Does CRM degrade gracefully when this assumption is violated? We test this by introducing the *temperature* multiplier $\mathbf{w} \mapsto \alpha\mathbf{w}, \alpha > 0$ (as discussed in Section 6.6) into the logging policy. For $\pi_0 = \pi_{\mathbf{w}}$, we scale $\mathbf{w} \mapsto \alpha\mathbf{w}$, to derive a "less stochastic" variant of π_0, and generate $\mathcal{D} \sim \pi_{\alpha\mathbf{w}}$. We report the performance of POEM($\mathcal{S}$) on the LYRL data set in Figure 6.3 as we change $\alpha \in [0.5, \ldots, 32]$, compared against π_0, and the deterministic predictor — π_0 *map* — derived from π_0. So long as there is some minimum amount of stochasticity in π_0, POEM($\mathcal{S}$) is still able to find a $\mathbf{w}$ that improves upon π_0 and π_0 *map*. The margin of improvement is typically greater when π_0 is more stochastic. Even when π_0 is barely stochastic ($\alpha \geq 2^4$), performance of POEM($\mathcal{S}$) simply recovers π_0 *map*, suggesting that the CRM principle indeed achieves robust learning.

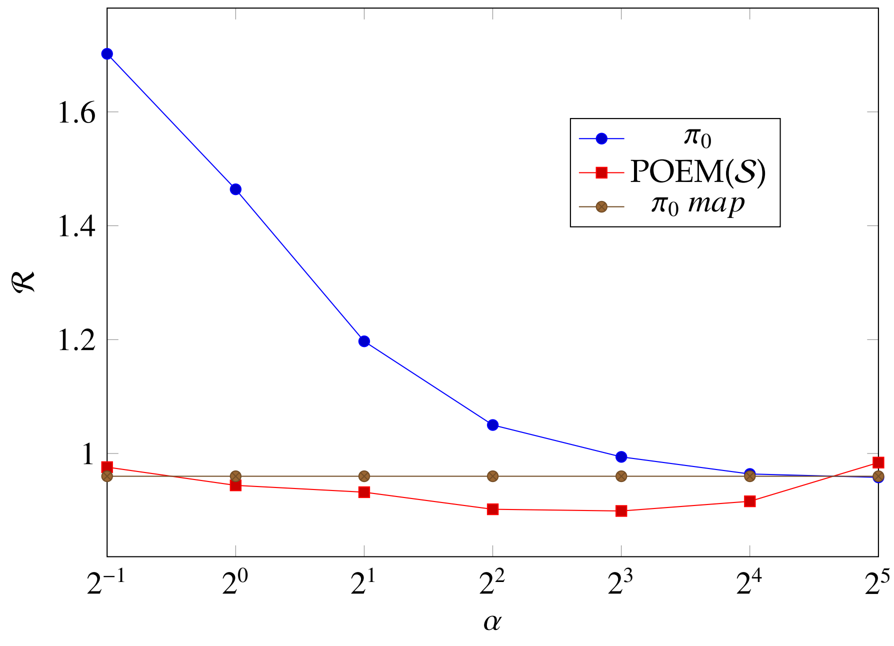


Figure 6.3: The performance of POEM($\mathcal{S}$) on the LYRL data set as π_0 becomes less stochastic. For $\alpha \geq 2^5$, $\pi_0 \equiv \pi_0$ *map* (within machine precision).

We observe the same trends (Figure 6.1, Figure 6.2 and Figure 6.3) across all data sets and optimization variants. They also remain unchanged when we include ℓ_2-regularization (analogous to supervised CRFs to capture the capacity of $\mathcal{H}_{lin}$ to overfit the training data).

6.8 Real-World Application

We now demonstrate how POEM (and in general the CRM principle) can be instantiated effectively in real world settings. Bloomberg[3], the financial and media company in New York, had the following challenging retrieval problem: the task was to train a high-precision classifier that could reliably pick the best answer doc^* (or none, if none answered the query) from a pool of candidate answers $\mathcal{Y}(x)$ for query x, where $\mathcal{Y}(x)$ was generated by an existing high-recall retrieval function. The challenge lay in collecting supervised labeled data that could be used to train this high-precision classifier.

Before we started our experiment with POEM, an existing high-precision classifier was already in operation. It was trained using a few labeled examples (x, doc^*), but scaling up the system to achieve improved accuracy appeared challenging given the cost of acquiring new (x, doc^*) pairs that mimicked what the system saw during its operation. However, it was possible to collect logs of the system, where each entry contained a query x and the features $\mathbf{f}(x, \text{doc})$ describing each candidate answer $\text{doc} \in \mathcal{Y}(x)$. The high-precision classifier could be modeled as a logistic regression classifier with weights $\mathbf{w}$ and a threshold τ. Each candidate was scored using $\mathbf{w}$, $\mathit{pred}(\text{doc}) = \mathbf{w}^T\mathbf{f}(x, \text{doc})$. If the highest scoring candidate $\mathit{pred}(\text{doc}^*) \geq \tau$, it was selected as the answer and otherwise the system abstained.

This existing system could easily be adapted to provide $\mathcal{D}$ as needed by POEM. For each x, a dummy $\text{doc}_0 \in \mathcal{Y}(x)$ is added to the candidate pool to model abstention. During the operation of the system, answers are *sampled* according to $\frac{\exp(\alpha \cdot \mathit{pred}(\text{doc}))}{\mathbb{Z}(x)}$. $\mathbb{Z}(x)$ is the partition function to ensure this is a

[3]https://www.bloomberg.com/

valid sampling distribution, $\mathbb{Z}(x) = \sum_{\mathrm{doc} \in \mathcal{Y}(x) \cup \mathrm{doc}_0} \exp(\alpha \cdot \mathit{pred}(\mathrm{doc}))$. Abstention is modeled by the fact that doc_0 is sampled with probability proportional to $\exp(\alpha \cdot \mathit{pred}(\mathrm{doc}_0))$. α is a temperature constant so that the system can be tuned to sample abstentions at roughly the same rate as its deterministic counterpart. Finally, the end-result feedback ($\delta \in \{\texttt{thumbs-up}, \texttt{thumbs-down}\}$) was logged and provided bandit feedback for the presented answer.

This data set was much easier to collect during the system run compared to annotating each x in the logs with the best possible doc^* that would have answered the query. We argue that this is a general, practical, alternative approach to training retrieval systems: use any strategy with a very high recall to construct $\mathcal{Y}$, then use the parameters $\mathbf{w}$ estimated using the CRM principle to search through this $\mathcal{Y}$ and find a precise answer.

On a small pilot study, we acquired $\mathcal{D}$ with $\simeq$ 4000 $(x, \mathrm{doc}, \frac{\exp(\alpha \cdot \mathit{pred}(\mathrm{doc}))}{\mathbb{Z}(x)}, \delta)$ tuples in the training set and $\simeq$ 500 tuples in the validation and test sets. We verified that the existing high-precision classifier was statistically significantly better than random baselines for the problem. POEM($\mathcal{S}$) is trained on this log data by performing gradient descent with $\mathbf{w}$ initialized to $\mathbf{w}_0 = 0$ and validating $\lambda = c\lambda^*, \quad c \in \left[10^{-6}, \ldots 1\right]$ as described in Section 6.5.4 and Section 6.7. POEM($\mathcal{S}$) found a $\hat{\mathbf{w}}$ that improved δ feedback over the existing system by over 30%, as estimated using the unbiased counterfactual estimator of Equation (6.1) on the test set. Without using the variance regularizer, the IPS($\mathcal{S}$) found a $\hat{\mathbf{w}}$ that degraded the system performance by 3.5% estimated counterfactually in the same way. This shows that POEM and the CRM principle can bring potential benefit even in binary-feedback multi-class classification settings where classic supervised learning approaches lack available data.

6.9 Conclusions and Future Work

Counterfactual risk minimization serves as a robust principle for designing algorithms that can learn from a batch of bandit feedback interactions. The key insight for CRM is to expand the classical notion of a hypothesis class to include stochastic policies, reason about variance in the risk estimator, and derive a generalization error bound over this hypothesis space. The practical take away is a simple, data-dependent regularizer that guarantees robust learning. Following the CRM principle, we developed the POEM (Policy Optimizer for Exponential Models) learning algorithm for structured output prediction. POEM can optimize over rich policy families (exponential models corresponding to soft-max linear rules in supervised learning), and deal with massive output spaces as efficiently as classical supervised methods.

The CRM principle more generally applies to supervised learning with non-differentiable losses, since the objective does not require the gradient of the loss function. We also foresee extensions of the algorithm to handle ordinal or co-active feedback models for $\delta(\cdot, \cdot)$, and extensions of the generalization error bound to include adaptive or deterministic π_0, etc.

CHAPTER 7

THE SELF-NORMALIZED ESTIMATOR FOR COUNTERFACTUAL LEARNING

7.1 Chapter Notes

This chapter describes joint work with Thorsten Joachims. It is a lightly edited version of a conference publication [116].

We will continue to study the Batch Learning from Bandit Feedback (BLBF) setting [8, 113] that was introduced in Chapter 6. The conventional counterfactual risk estimator used in prior works on BLBF exhibits severe anomalies that can lead to degeneracies when used in ERM. In particular, the estimator exhibits a new form of *Propensity Overfitting* that causes severely biased risk estimates for the ERM minimizer. By introducing multiplicative control variates, we will replace this risk estimator with a *Self-Normalized Risk Estimator* that provably avoids these degeneracies. Empirical evaluation in semi-synthetic experiments (following the methodology sketched in Section 1.2) confirms that the desirable theoretical properties of the Self-Normalized Risk Estimator translate into improved generalization performance and robustness.

7.2 Related work

All the estimators we study in this chapter are instances of importance sampling for Monte Carlo approximation and can be traced back to "What-If simulations" [121]. We additionally show that importance sampling can overfit in hitherto unforeseen ways with the capacity of the hypothesis space during counterfactual learning. We call this new overfitting *Propensity Overfitting*.

Classic variance reduction techniques for importance sampling are also useful for counterfactual evaluation and learning. For instance, importance weights can be "clipped" [51] to trade-off bias against variance in the estimators [12]. Additive control variates give rise to regression estimators [73] and doubly robust estimators [31, 30]. Our proposal uses *multiplicative* control variates. These are widely used in financial applications [13] and policy iteration for reinforcement learning [102]. In particular, we study the self-normalized estimator [121] which is superior to the vanilla estimator when fluctuations in the weights dominate the variance [42]. We additionally show that the self-normalized estimator neatly addresses propensity overfitting.

7.3 The Propensity Overfitting Problem

The CRM objective of Equation (6.6) penalizes those $\pi \in \mathcal{H}$ that are "far" from the logging policy π_0 (as measured by their empirical variance $\hat{\boldsymbol{Var}}(z_\pi)$). We can intuitively understand this penalty as a safeguard against overfitting. However, overfitting in BLBF is more nuanced than in conventional supervised learning. In particular, the unbiased risk estimator of Equation (6.1) has two anomalies. Even if $\delta(\cdot,\cdot) \in [\triangledown, \vartriangle]$, the value of $\hat{R}(\pi)$ estimated on a finite sample need not lie in that range. Furthermore, if $\delta(\cdot,\cdot)$ is translated by a constant $\delta(\cdot,\cdot) + C$, $R(\pi)$ becomes $R(\pi) + C$ by linearity of expectation — but the unbiased estimator on a finite sample need not equal $\hat{R}(\pi) + C$. In short, this risk estimator is not *equivariant* [42]. The various thresholding schemes for importance sampling [51, 110, 12, 24] only exacerbate this effect. These anomalies leave us vulnerable to a peculiar kind of overfitting, as we see in the following example.

Example 8. For the input space of integers $\mathcal{X} = \{1, \dots k\}$ and the output space $\mathcal{Y} = \{1, \dots k\}$, define

$$\delta(x, y) = \begin{cases} -2 & \text{if } y = x \\ -1 & \text{otherwise.} \end{cases}$$

The hypothesis space $\mathcal{H}$ is the set of all deterministic functions $S : \mathcal{X} \mapsto \mathcal{Y}$.

$$\pi_S(y \mid x) = \begin{cases} 1 & \text{if } S(x) = y \\ 0 & \text{otherwise.} \end{cases}$$

Data is drawn uniformly, $x \sim$ Uniform($\mathcal{X}$) and $\pi_0(\mathcal{Y} \mid x) =$ Uniform($\mathcal{Y}$) for all x. The hypothesis π^* with minimum true risk is π_{S^*} with $S^*(x) = x$, which has risk $R(\pi^*) = -2$.

When drawing a training sample $\mathcal{D} = ((x_1, y_1, \delta_1, p_1), \dots (x_n, y_n, \delta_n, p_n))$, let us first consider the special case where all x_i in the sample are distinct. This case is quite likely if n is small relative to k. In this case $\mathcal{H}$ contains a hypothesis π_{overfit}, which assigns $S(x_i) = y_i$ for all i. This hypothesis has the following empirical risk as estimated by Equation (6.1):

$$\hat{R}(\pi_{\text{overfit}}) = \frac{1}{n}\sum_{i=1}^{n} \delta_i \frac{\pi_{\text{overfit}}(y_i \mid x_i)}{p_i} = \frac{1}{n}\sum_{i=1}^{n} \delta_i \frac{1}{1/k} \leq \frac{1}{n}\sum_{i=1}^{n} -1 \frac{1}{1/k} = -k.$$

Clearly this risk estimate shows severe overfitting since it can be arbitrarily lower than the true risk $R(\pi^*) = -2$ of the best hypothesis π^* with appropriately chosen k (or, more generally, the choice of π_0). This situation is in stark contrast to overfitting in full-information supervised learning, where at least the overfitted risk is bounded by the lower range of the loss function. Note that the empirical risk $\hat{R}(\pi^*)$ of π^* concentrates around -2. ERM will, hence, select π_{overfit} over π^*.

Even if we are not in the special case of having a sample with all distinct x_i, this type of overfitting still exists. In particular, if there are only ℓ distinct x_i in $\mathcal{D}$, then there still exists a π_{overfit} with $\hat{R}(\pi_{\text{overfit}}) \leq -k\frac{\ell}{n}$. Finally, note that this type of overfitting behavior is not an artifact of this example. Section 7.6 shows that this behavior is ubiquitous in all the datasets we explored.

Maybe this problem could be avoided by transforming the loss? For example, suppose we translate the loss by adding 2 to δ so that now all loss values become non-negative. This transformation results in the new loss function $\delta'(x, y)$ taking values 0 and 1. In conventional supervised learning an additive translation of the loss does not change the empirical risk minimizer. Suppose we draw a sample $\mathcal{D}$ in which not all possible values y for x_i are observed for all x_i in the sample (again, such a sample is likely for sufficiently large k). Now there are many hypotheses $\pi_{\text{overfit}'}$ that predict one of the unobserved y for each x_i, basically avoiding the training data.

$$\hat{R}(\pi_{\text{overfit}'}) = \frac{1}{n}\sum_{i=1}^{n} \delta_i \frac{\pi_{\text{overfit}'}(y_i \mid x_i)}{p_i} = \frac{1}{n}\sum_{i=1}^{n} \delta_i \frac{0}{1/k} = 0.$$

Again we are faced with overfitting since many overfit hypotheses are indistinguishable from the true risk minimizer π^* with true risk $R(\pi^*) = 0$ and empirical risk $\hat{R}(\pi^*) = 0$.

These examples indicate that this overfitting occurs regardless of how the loss is transformed. Intuitively, this type of overfitting occurs since the risk estimate according to Equation (6.1) can be minimized not only by putting large probability mass $\pi(y \mid x)$ on the examples with low loss $\delta(x, y)$, but by maximizing (for $\delta \leq 0$) or alternatively, minimizing (for $\delta \geq 0$) the sum of the weights

$$\hat{T}(\pi) \quad = \quad \frac{1}{n}\sum_{i=1}^{n}\frac{\pi(y_i \mid x_i)}{p_i}. \tag{7.1}$$

For this reason, we call this type of overfitting *Propensity Overfitting*. This overfitting is in stark contrast to overfitting in supervised learning, which we call *Loss Overfitting*. Intuitively, Loss Overfitting occurs because the capacity of $\mathcal{H}$ fits spurious patterns of low δ in the data. In Propensity Overfitting, the capacity in $\mathcal{H}$ allows overfitting the propensities — for positive δ, hypotheses that avoid $\mathcal{D}$ are selected; for negative δ, hypotheses that over-represent $\mathcal{D}$ are selected.

The variance regularization of CRM combats both Loss Overfitting and Propensity Overfitting by optimizing a more informed generalization error bound. However, the empirical variance estimate is also affected by Propensity Overfitting — especially for positive losses. Can we avoid Propensity Overfitting more directly?

7.4 Control Variates and the Self-Normalized Estimator

To avoid Propensity Overfitting, we must first detect when and where it is occurring. For this, we draw on diagnostic tools used in importance sampling [86]. Note that for any $\pi \in \mathcal{H}$, the sum of propensity weights $\hat{T}(\pi)$ from Equation (7.1) always has expected value 1 under the conditions required for the unbiased estimator of Equation (6.1).

$$\mathbb{E}\left[\hat{T}(\pi)\right] = \frac{1}{n}\sum_{i=1}^{n}\int \frac{\pi(y_i \mid x_i)}{\pi_0(y_i \mid x_i)}\pi_0(y_i \mid x_i)\Pr(x_i)dy_i dx_i = \frac{1}{n}\sum_{i=1}^{n}\int 1 \Pr(x_i)dx_i = 1. \tag{7.2}$$

This observation means that we can identify hypotheses that suffer from Propensity Overfitting based on how far $\hat{T}(\pi)$ deviates from its expected value of 1. Since $\frac{\pi(y|x)}{\pi_0(y|x)}$ is likely correlated with $\delta(x,y)\frac{\pi(y|x)}{\pi_0(y|x)}$, a large deviation in $\hat{T}(\pi)$ suggests a large deviation in $\hat{R}(\pi)$ and consequently a bad risk estimate.

How can we use the knowledge that $\forall \pi \in \mathcal{H} : \mathbb{E}\left[\hat{T}(\pi)\right] = 1$ to avoid degenerate risk estimates in a principled way? While one could use concentration inequalities to explicitly detect and eliminate overfit hypotheses based on $\hat{T}(\pi)$, we use control variates to derive an improved risk estimator that directly incorporates this knowledge.

Control Variates: Control variates — random variables whose expectation is known — are a classic tool used to reduce the variance of Monte Carlo approximations [86]. Let $U(Y)$ be a control variate with known expectation $\mathbb{E}_Y[U(Y)] = \boldsymbol{u} \neq 0$, and let $\mathbb{E}_Y[V(Y)]$ be an expectation that we would like to estimate based on independent samples of Y. Employing $U(Y)$ as a multiplicative control variate, $\mathbb{E}_Y[V(Y)] = \frac{\mathbb{E}[V(Y)]}{\mathbb{E}[U(Y)]}\boldsymbol{u}$. This observation motivates the estimator

$$\hat{V}^{SN} = \frac{\sum_{i=1}^{n} V(Y_i)}{\sum_{i=1}^{n} U(Y_i)}\boldsymbol{u}, \tag{7.3}$$

which is called the *Self-Normalized estimator* in the importance sampling literature [121, 61, 97]. This estimator has substantially lower variance if $V(Y)$ and $U(Y)$ are correlated [42].

Self-Normalized Risk Estimator: Let us use $T(\pi)$ as a control variate when estimating $R(\pi)$, yielding

$$\hat{R}^{SN}(\pi) = \frac{\sum_{i=1}^{n} \delta_i \frac{\pi(y_i|x_i)}{p_i}}{\sum_{i=1}^{n} \frac{\pi(y_i|x_i)}{p_i}}. \tag{7.4}$$

Hesterberg reports that this estimator tends to be more accurate than the unbiased estimator of Equation (6.1) when fluctuations in the sampling weights dominate the fluctuations in $\delta(x,y)$ [42].

Observe that the estimate is just a convex combination of the δ_i observed in the sample. If $\delta(\cdot,\cdot)$ is now translated by a constant $\delta(\cdot,\cdot) + C$, both the true risk $R(\pi)$ and the finite sample estimate $\hat{R}^{SN}(\pi)$ get shifted by C. Hence $\hat{R}^{SN}(\pi)$ is *equivariant*, unlike $\hat{R}(\pi)$ [42]. Moreover, $\hat{R}^{SN}(\pi)$ is always bounded within the range of δ. So the overfitted risk due to ERM will now be bounded by the lower range of the loss, analogous to full-information supervised learning.

Finally, while the self-normalized risk estimator is not unbiased ($\mathbb{E}\left[\frac{\hat{R}^{SN}(\pi)}{\hat{T}(\pi)}\right] \neq \frac{R(\pi)}{\mathbb{E}[\hat{T}(\pi)]}$ in general), it is strongly consistent and approaches the desired expectation when n is large.

Theorem 9. *Let $\mathcal{D}$ be drawn $(x_i, y_i, \delta_i, p_i) \overset{i.i.d.}{\sim} \pi_0$, from a π_0 that has full support over $\mathcal{Y}$ for all x. Then,*

$$\forall \pi \in \mathcal{H} : \Pr(\lim_{n\to\infty} \hat{R}^{SN}(\pi) = R(\pi)) = 1.$$

Proof. The numerator of $\hat{R}^{SN}(\pi)$ in Equation (7.4) are *i.i.d.* observations with mean $R(\pi)$. Strong law of large numbers gives $\Pr(\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} \delta_i \frac{\pi(y_i|x_i)}{p_i} = R(\pi)) = 1$. Similarly, the denominator has *i.i.d.* observations with mean 1. So, the strong law of large numbers implies $\Pr(\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} \frac{\pi(y_i|x_i)}{p_i} = 1) = 1$. Hence, $\Pr(\lim_{n\to\infty} \hat{R}^{SN}(\pi) = R(\pi)) = 1$. $\square$

In summary, the self-normalized risk estimator $\hat{R}^{SN}(\pi)$ in Equation (7.4) resolves all the problems of the unbiased estimator $\hat{R}(\pi)$ from Equation (6.1) identified in Section 7.3.

7.5 Learning Method: Norm-POEM

We now derive a learning algorithm, called Norm-POEM (Normalized Policy Optimizer for Exponential Models), for structured output prediction. The algorithm is analogous to POEM [113] in its choice of hypothesis space and its application of the CRM principle, but it replaces the conventional estimator of Equation (6.1) with the self-normalized estimator of Equation (7.4).

Hypothesis Space: Norm-POEM follows prior work [113, 64] and learns stochastic linear rules $\pi\mathbf{w} \in \mathcal{H}_{lin}$ parametrized by $\mathbf{w}$ that operate on a d-dimensional joint feature map $\mathbf{f}(x, y)$.

$$\pi\mathbf{w}(y \mid x) = \exp(\mathbf{w}^T\mathbf{f}(x, y))/\mathbb{Z}(x).$$

$\mathbb{Z}(x) = \sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^T\mathbf{f}(x, y'))$ is the partition function.

Empirical Variance Estimate: To instantiate the CRM objective from Equation (6.6), we need an empirical variance estimate $\hat{\boldsymbol{Var}}(\hat{R}^{SN}(\pi))$ for the self-normalized risk estimator. We use an approximate variance estimate [97, Section 4.3] for the ratio estimator of Equation (7.3). Using the Normal approximation argument [86, Equation 9.9],

$$\hat{\boldsymbol{Var}}(\hat{R}^{SN}(\pi)) = \frac{\sum_{i=1}^{n}(\delta_i - \hat{R}^{SN}(\pi))^2(\frac{\pi(y_i|x_i)}{p_i})^2}{(\sum_{i=1}^{n}\frac{\pi(y_i|x_i)}{p_i})^2}. \tag{7.5}$$

Using the delta method to approximate the variance [61] yields the same formula. To invoke asymptotic normality of the estimator (and indeed, for reliable importance sampling estimates), we require the true variance of the self-normalized estimator $\boldsymbol{Var}(\hat{R}^{SN}(\pi))$ to exist. We can guarantee this by thresholding the importance weights, analogous to $\hat{R}^{M}(\pi)$ of Equation (6.2).

The benefits of the self-normalized estimator come at a computational cost. The risk estimator of POEM had a simpler empirical variance formula which could be approximated by Taylor expansion and optimized using stochastic gradient descent. The variance of Equation (7.5) does not admit stochastic optimization. Surprisingly, in our experiments in Section 7.6, we find that the improved robustness of Norm-POEM permits fast convergence during training even without stochastic optimization.

Training Objective of Norm-POEM. The objective is now derived by substituting the self-normalized risk estimator of Equation (7.4) and its sample variance estimate from Equation (7.5) into the CRM objective Equation (6.6) for the hypothesis space $\mathcal{H}_{lin}$. By design, $\pi_{\mathbf{w}}$ lies in the exponential family of distributions. So, the gradient of the resulting objective can be tractably computed whenever the partition function $\mathbb{Z}(x)$ is tractable. Doing so yields a non-convex objective in the parameters $\mathbf{w}$ which we optimize using L-BFGS [16]. The choice of L-BFGS for non-convex and non-smooth optimization is well supported [68, 134]. Analogous to POEM, the hyper-parameters M (clipping to prevent unbounded variance) and λ (strength of variance regularization) can be calibrated via counterfactual evaluation (using either the vanilla IPS estimate of Equation (6.1) or the self-normalized estimate of Equation (7.4)) on a held-out validation set. In summary, the per-iteration cost of optimizing the Norm-POEM objective has the same complexity as the per-iteration cost of POEM with L-BFGS. It requires the same set of hyper-parameters. Moreover, it can be done tractably whenever the corresponding supervised CRF can be learned efficiently. Software implementing Norm-POEM is available at `http://www.cs.cornell.edu/~adith/POEM/`.

7.6 Experiments

We will now empirically verify if the self-normalized estimator as used in Norm-POEM can indeed guard against propensity overfitting and attain robust generalization performance. The experiment setup is identical to Section 6.7 of Chapter 6. POEM uses the CRM principle instantiated with the unbiased estimator of Equation (6.1) while Norm-POEM uses the self-normalized estimator of Equation (7.4). We report performance of BLBF approaches without ℓ_2-regularization here; we observed Norm-POEM dominated POEM even when both methods had their ℓ_2-regularizers cross-validated. Since the choice of optimization method could be a confounder, we use L-BFGS for all methods and experiments. We used a more fine-grained grid of hyper-parameter choices than Section 6.7 — λ (variance regularization) and M (clipping constant) for BLBF approaches, and ℓ_2-regularization for the skyline CRF that performs supervised learning on the full-information training set.

7.6.1 What is the Generalization Performance of Norm-POEM?

The key question is whether the appealing theoretical properties of the self-normalized estimator lead to better generalization performance. In Table 7.1, we report the test set loss for Norm-POEM and POEM averaged over 10 runs. On each run, π_0 has different performance (trained on random 5% subsets), but Norm-POEM consistently beats POEM. The setup is identical to Section 6.7.1, the tiny differences in performance of CRF compared to Table 6.3 is due to hyper-parameter selection over a more fine-grained grid of choices. Variability in POEM and Norm-POEM arises from the different quality of π_0 in each trial of the experiment.

$\mathcal{R}$	Scene	Yeast	TMC	LYRL
π_0	1.511	5.577	3.442	1.459
POEM	1.200	4.520	2.152	0.914
Norm-POEM	1.045	3.876	2.072	0.799
CRF	0.657	2.830	1.187	0.222

Table 7.1: The test set Hamming loss for BLBF approaches averaged over 10 runs. Norm-POEM significantly outperforms POEM on all four datasets (one-tailed paired difference t-test at a significance level of 0.05).

7.6.2 How Does Generalization of Norm-POEM Improve With Dataset Size?

The plot below Figure 7.1 shows how generalization performance improves with more training data for a single run of the experiment on the Yeast dataset. We achieve this by varying the number of times we replay the training set to collect samples from π_0 (*ReplayCount*), analogous to Section 6.7.4. Norm-POEM consistently outperforms POEM for all training sample sizes.

7.6.3 Does Norm-POEM Avoid Propensity Overfitting?

While the previous results indicate that Norm-POEM achieves good performance, it remains to be verified that this improved performance is indeed due to improved control over Propensity Overfitting. Table 7.2 (left) shows the average $\hat{T}(\hat{\pi})$ for the hypothesis $\hat{\pi}$ selected by each approach. Indeed, $\hat{T}(\hat{\pi})$ is close to its known expectation of 1 for Norm-POEM, while it is severely biased for POEM. Furthermore, the value of $\hat{T}(\hat{\pi})$ depends heavily on how the losses δ

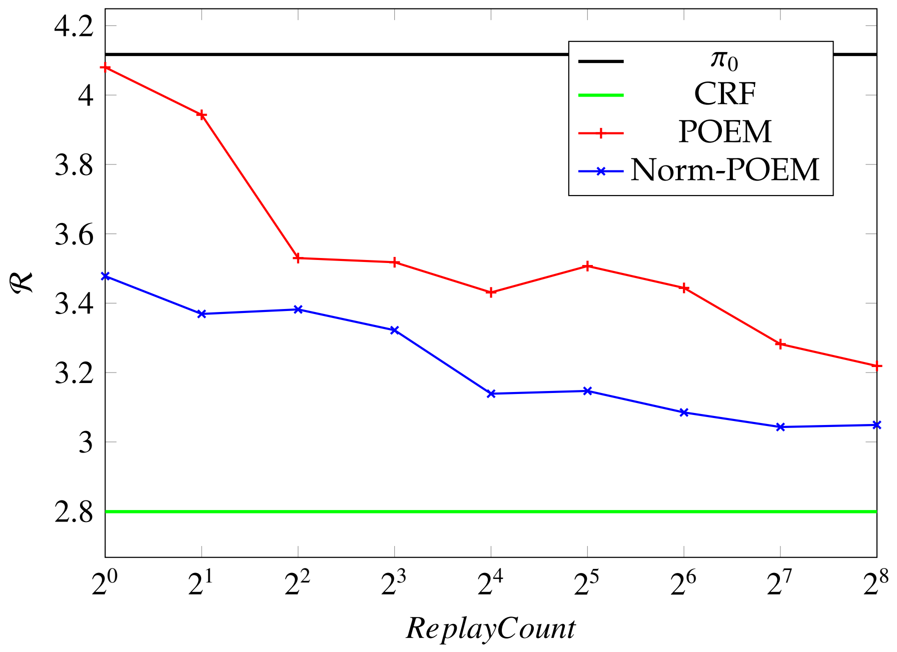


Figure 7.1: Test set Hamming loss for various BLBF approaches as $n \to \infty$ on the Yeast dataset. All approaches will converge to CRF performance in the limit, but the rate of convergence is slow since π_0 is thin-tailed.

are translated for POEM, as predicted by theory. As anticipated by our earlier observation that the self-normalized estimator is equivariant, Norm-POEM is unaffected by translations of δ. Table 7.2 (right) shows that the same goes for the prediction error on the test set. Norm-POEM is consistently good while POEM fails catastrophically (for instance, on the TMC dataset, POEM is worse than random guessing).

7.6.4 Is CRM Variance Regularization Still Necessary?

It may be possible that the improved self-normalized estimator no longer requires variance regularization. The loss of the unregularized estimator is reported (Norm-IPS) in Table 7.3. We see that variance regularization still helps.

	$\hat{T}(\hat{\pi})$				$\mathcal{R}(\hat{\pi})$			
	Scene	Yeast	TMC	LYRL	Scene	Yeast	TMC	LYRL
POEM-δ_+	0.274	0.028	0.000	0.175	2.059	5.441	17.305	2.399
POEM-δ_-	1.782	5.352	2.802	1.230	1.200	4.520	2.152	0.914
NPOEM-δ_+	0.981	0.840	0.941	0.945	1.058	3.881	2.079	0.799
NPOEM-δ_-	0.981	0.821	0.938	0.945	1.045	3.876	2.072	0.799

Table 7.2: Mean of the unclipped importance weights $\hat{T}(\hat{\pi})$ (left) and test set Hamming loss $\mathcal{R}$ (right), averaged over 10 runs for POEM and Norm-POEM (NPOEM). δ_+ and δ_- indicate whether the loss was translated to be always positive or always negative.

$\mathcal{R}$	Scene	Yeast	TMC	LYRL
Norm-IPS	1.072	3.905	3.609	0.806
Norm-POEM	1.045	3.876	2.072	0.799

Table 7.3: The test set Hamming loss for Norm-POEM and the variance agnostic Norm-IPS averaged over the same 10 runs as Table 7.1. On Scene, TMC and LYRL, Norm-POEM is significantly better than Norm-IPS (one-tailed paired difference t-test at a significance level of 0.05).

7.6.5 How Computationally Efficient Is Norm-POEM?

The empirical runtime of Norm-POEM is surprisingly faster than POEM. Even though normalization increases the per-iteration computation cost, optimization tends to converge in fewer iterations than for POEM. We find that POEM picks a hypothesis with large $||\mathbf{w}||$, attempting to assign a probability of 1 to all training points with negative losses. However, Norm-POEM converges to a much shorter $||\mathbf{w}||$. The loss of an instance *relative* to others in sample $\mathcal{D}$ governs how Norm-POEM tries to fit it. This behavior is another nice consequence of the fact that the overfitted risk of $\hat{R}^{SN}(\pi)$ is bounded and small. Overall, the runtime of Norm-POEM is on the same order of magnitude as that of a full-information CRF as reported in Table 7.4. Norm-POEM has runtime that is competitive with the runtimes reported for POEM with stochastic optimization [113] (see Section 6.7.2) while providing substantially better generalization performance.

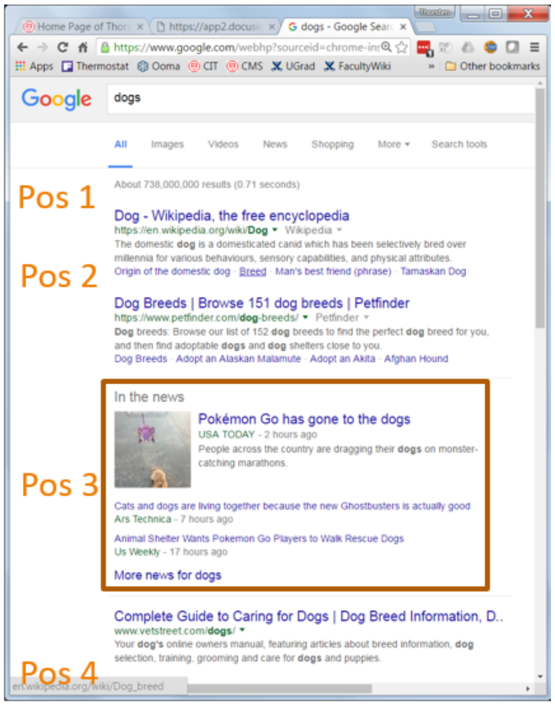
Time(s)	Scene	Yeast	TMC	LYRL
POEM	78.69	98.65	716.51	617.30
Norm-POEM	7.28	10.15	227.88	142.50
CRF	4.94	3.43	89.24	72.34

Table 7.4: Time in seconds averaged across validation runs. CRF is implemented by scikit-learn [87].

We observe the same trends for Norm-POEM when different properties of π_0 are varied (e.g. stochasticity and quality), as reported for POEM in Section 6.7.5 and Section 6.7.6.

7.7 Real-World Experiment

A large media company, based in New York, wanted to introduce a new facet containing news articles in their search results. They wanted to personalize the placement of this facet on the results page so as to engage their users with relevant, high-quality content. Recall the $x \mapsto y \mapsto \delta$ schematic of Section 2.1. In this application, x encodes contextual factors like the user, the ranking of search results, the contents of the news facet, etc. The action y denotes what position the newsbox is inserted into (see Figure 7.2). The feedback δ is a particular whole page metric called Mean Reciprocal Rank (MRR) that takes values in $[0, 1]$. The goal is to design a policy $\pi : x \mapsto y$ that achieves high MRR, or equivalently, a low $1 - \text{MRR}$.



About 738,000,000 results (0.71 seconds)

Dog - Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/**Dog** Wikipedia
The domestic **dog** is a domesticated canid which has been selectively bred over millennia for various behaviours, sensory capabilities, and physical attributes.
Origin of the domestic dog · Breed · Man's best friend (phrase) · Tamaskan Dog

Dog Breeds | Browse 151 dog breeds | Petfinder
https://www.petfinder.com/**dog**-breeds/ Petfinder
Dog breeds: Browse our list of 152 **dog** breeds to find the perfect **dog** breed for you, and then find adoptable **dogs** and **dog** shelters close to you.
Dog Breeds · Adopt an Alaskan Malamute · Adopt an Akita · Afghan Hound

In the news

Pokémon Go has gone to the dogs
USA TODAY - 2 hours ago
People across the country are dragging their **dogs** on monster-catching marathons.

Cats and dogs are living together because the new Ghostbusters is actually good
Ars Technica - 7 hours ago

Animal Shelter Wants Pokemon Go Players to Walk Rescue Dogs
Us Weekly - 17 hours ago

More news for dogs

Complete Guide to Caring for Dogs | Dog Breed Information, D..
www.vetstreet.com/**dogs**/
Your **dog's** online owners manual, featuring articles about breed information, **dog** selection, training, grooming and care for **dogs** and puppies.

Figure 7.2: Illustrating the Newsbox placement problem. We wish to insert the news facet at a "good" location on a search results page. So, we want to tune the placement engine to use contextual factors (e.g., the user, news contents, etc.) to place the newsbox at one of the candidate positions so as to improve user engagement with the results.

The search team engineered a rule-based π^{RULE} that scored all candidate positions to decide where to insert the newsbox, and developed randomized variants of π^{RULE}:

$$\pi_0(\text{Position } p) \propto \exp\left(\alpha \cdot \pi^{RULE}(\text{Position } p)\right),$$

using a specific temperature hyper-parameter $\alpha > 0$. By simply deploying π_0, they collected log data that was suitable for BLBF with Norm-POEM. A scalable but approximate version of Norm-POEM was trained on this data, and the resulting policies were evaluated using Equation (6.1). Figure 7.3 shows 100 repetitions of this experiment and reports the average $1 - \text{MRR}$ performance of the learned policies (POEM) as well as the empirical standard deviation. The performance of the old rule-based policy is plotted (RULE) as a baseline. On average, the policy learned using Norm-POEM performs significantly better than the rule-based policy. The Norm-POEM results appear to have a much larger variance than the rule-based policy. The figure on the right shows the performance of Norm-POEM relative to π^{RULE} for each of the 100 repetitions. There are positive improvements on 98 out of 100 datasets. The search team conjectured that the logging policy π_0 is very different from the one learned by Norm-POEM, and the variance of off-policy estimates should be greatly reduced when the new Norm-POEM policies are deployed.

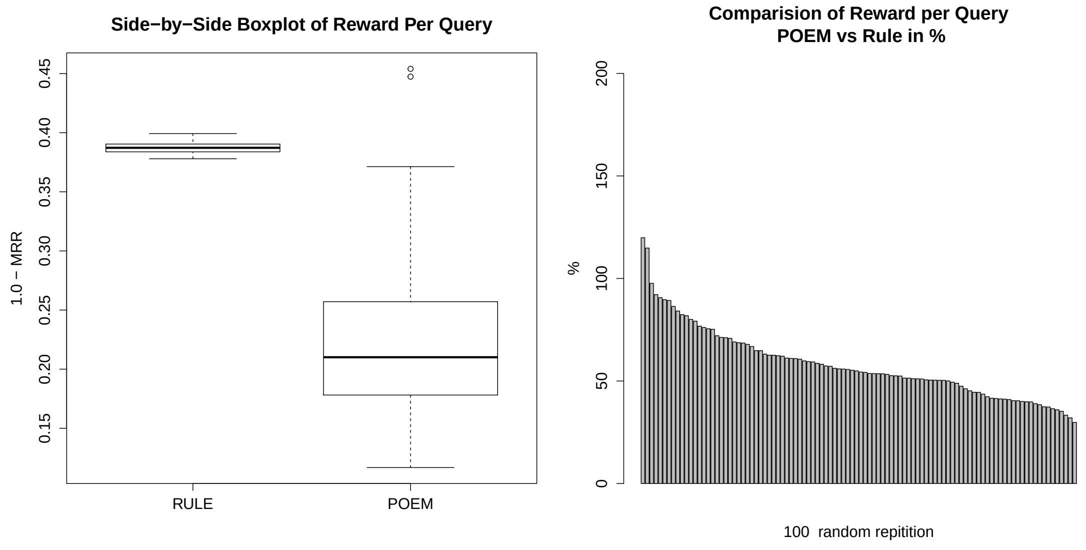


Figure 7.3: The average 1 − MRR of the placement policy learned by Norm-POEM and the rule-based π^{RULE} on 100 random datasets logged by π_0 are plotted in a side-by-side boxplot.

The key take away of these experiments is that it is possible to do controlled randomization without a massive impact on the current system, and the resulting logged datasetis valuable because it enables counterfactual learning.

7.8 Conclusions

We identify the problem of propensity overfitting when using the conventional unbiased risk estimator for ERM in batch learning from bandit feedback. To remedy this problem, we propose the use of a multiplicative control variate that leads to the self-normalized risk estimator. This estimator provably avoids the anomalies of the conventional estimator. Deriving a new learning algorithm called Norm-POEM based on the CRM principle using the new estimator, we show that this algorithm has significantly better generalization performance.

CHAPTER 8

RESOURCES

This chapter collects additional reading material, datasets, and software for counterfactual evaluation and learning. These resources were either referenced in the thesis, used in experiments, or developed during my research.

8.1 Datasets

Building and deploying interactive systems in real-world settings can be challenging. The datasets below allow us to conduct research in counterfactual evaluation and learning techniques without requiring access to real-world deployments. Of course the success of any technique we discover using these datasets is ultimately governed by their performance in a real-world interactive system.

Several datasets were collected for training supervised learning algorithms in interactive scenarios. These datasets can be used for semi-synthetic simulations following the methodology sketched in Section 1.2 — see Chapters 4, 5 and 6 for concrete examples. Some of these datasets are:

- LETOR datasets [91] These datasets contain query-document relevances and features describing query-document pairs. These datasets are useful for setting up learning-to-rank and off-policy simulations; see Chapters 4 and 5.
- LibSVM repository for multi-label datasets Multi-class and multi-label classification are more traditional supervised learning problems. It is still possible to experiment with these datasets using the Supervised $\mapsto$ Bandit

methodology[66, 8]; see Chapters 6 and 7.

Some datasets were collected directly in an interactive setting. To ensure that there are no unobserved confounders, they come with test sets that were collected using uniform randomization.

- Yahoo! R3 [79] Users volunteered ratings for songs. In the test set, they were also assigned songs uniformly at random to rate.
- Yahoo! R6 [72] The Yahoo! front page module uniformly randomized news recommendations for 10 days. User clicks were recorded as feedback.
- Coat Shopping [101] We created this dataset by first creating an online catalog of coats. Amazon Mechanical Turkers browsed through this catalog and provided ratings for coats they would like to buy. The Turkers were also assigned uniformly randomly chosen items to be rated.

We also released the Criteo dataset [67] which has records of user interactions with display ads on the web. The interactions are modeled well as a contextual combinatorial bandit feedback (see Chapter 5) and the logging system employed non-uniform randomization with carefully logged propensities. Hence this dataset enables direct off-policy causal estimation on a large scale. We also have logged features describing user contexts, displayed banners of ads and features for all candidate ads considered for display by the logging system. Hence, this dataset contains sufficient information for off-policy learning [67].

8.2 Software

There are a few BLBF algorithms and implementations that are available online.

- Vowpal Wabbit This implements several algorithms for BLBF at scale by streaming data through an online learner.
- POEM Our implementation of POEM [113] and Norm-POEM [116] is also online. They are batch algorithms for structured prediction and are bottlenecked by the speed of CRFs [64] for comparable supervised structured prediction tasks. A more scalable stand-alone implementation accompanies the Criteo dataset.
- The latest version of SVM-rank [52, 57] incorporates the propensity weighting we introduced in Chapter 4.
- Our implementation of propensity-weighted matrix factorization (introduced in Chapter 3) is also available online This algorithm implements collaborative filtering at scale to recommend items to users by training on MNAR ratings.

Finally, we demonstrated several off-policy estimators on a toy recommendation example [117] This demonstration contains implementations of several off policy estimators (including our contributions in Chapter 5) and BLBF learning algorithms.

8.3 Additional Reading Material

Two tutorials cover off policy evaluation and learning techniques with a focus on interactive systems.

- Tutorial on Counterfactual Evaluation and Learning for Search, Recommendation and Ad Placement [117].
- Tutorial on Offline Evaluation and Optimization for Interactive Systems [69].

Also, Bottou et al [12] provide a gentle introduction to confounding effects that arise when optimizing learning models in interactive systems.

CHAPTER 9

CONCLUSION

We related several applications involving interactive systems to causal estimation and inference. We introduced a counterfactual model for information retrieval (Chapter 4), viewed recommendations by collaborative filtering systems as interventions (Chapter 3) and extensively explored off-policy problems in a stylized model called Batch Learning from Bandit Feedback — BLBF (Chapters 5, 6 and 7).

Departing from the standard machine learning view of each of these applications and instead formalizing a counterfactual model proved useful in each case. Using techniques for observational studies, we were able to learn recommender systems (Chapter 3) and ranking models (Chapter 4) without requiring active randomization of a deployed system. By studying the statistical properties of off-policy estimation problems, we designed new off-policy estimators (Chapter 5) and new learning algorithms (Chapters 6 and 7) for Batch Learning from Bandit Feedback (BLBF).

In each application, we found that the techniques motivated by a counterfactual view of the problem naturally complemented algorithms developed from a standard machine learning view. This view allowed us to incorporate causal reasoning in several popular machine learning algorithms that are widely used in practice today. In Chapter 5 we combined the PI estimator for the off-policy slate recommendation problem with classic pointwise learning-to-rank algorithms to derive a ranking algorithm. In Chapter 4 we saw that traditional pairwise learning-to-rank algorithms could be naturally coupled with propensity modeling. In Chapter 3 we showed how inverse propensity weighting perfectly complements standard weighted matrix factorization approaches to the rating prediction problem. Chapters 6 and 7 used linear structured predictors from supervised learning to efficiently implement the Counterfactual Risk Minimization (CRM) principle.

This thesis contributed techniques that advance the state-of-the-art for evaluating and training interactive systems. The current practice for training and evaluating these systems either does not re-use the logged data collected from historical systems or requires manual annotations to employ conventional supervised learning. Online A/B testing (for evaluation) and explore-exploit bandit algorithms (for learning) essentially ignore the vast amounts of user feedback we collected from earlier systems. The techniques explored in this thesis enable "offline" A/B testing, and warm-starting bandit algorithms so that they interact more reliably. Evaluation and training using offline machine learning algorithms require supervised judgments. The techniques we studied allow us to train variants of these algorithms using logged user feedback directly.

9.1 Future Work

Answering evaluation ("How good is a new system?") and training ("Find the best new system") questions directly using logged data is an exercise in counterfactual reasoning and we saw two broad approaches to answer such questions. The first ("Model the world") models user behavior and directly answers "How would the user react to a new action?"; The second ("Model the bias") models the confounding factors in collected data to draw unbiased and reliable counterfactual estimates in aggregate. Modeling users ("Model the world") is a powerful approach to answering counterfactual questions about their behavior.

However, user models are typically misspecified and can sometimes be overkill for getting reliable estimates for evaluating and learning good interaction policies. Model-free ("Model the bias") methods are founded on Monte Carlo estimation and directly address the policy evaluation problem. However, by being agnostic to user behavior and dealing with population-level aggregates, they are often limited in the kinds of off-policy inferences they can reliably perform. By being flexible about combining and applying these approaches in practice, we can achieve tractable solutions and gain new insight. For instance, model-based methods ("model the world") typically use a machine-learned model trained on some data as a user or environment model. "Model the bias" reasoning tells us that we must not simply think of this machine-learned model as a black-box, but we should also account for confounding factors in its training data.

There is much work that can be done to develop these approaches further. These avenues for future research can be grouped into four inter-related *research thrusts* — *theoretical*, *algorithmic*, *learning methods*, and *deployment*.

Learning Methods We have developed counterfactual learning methods for specific applications and a recipe to derive other methods using the Counterfactual Risk Minimization principle. Preliminary results indicate that recent advances in *deep learning* are compatible with this recipe. Can we develop a repertoire of counterfactual learning methods (e.g., ensembles, trees, etc.)?

Theory We still lack a complete understanding of what properties are important for counterfactual estimation. Beyond statistical properties like bias and variance of estimators, optimization properties can become crucial when they are employed in counterfactual learning. How do we best pick the bias-variance-optimization trade-offs when *designing* counterfactual estimators? Moreover, in several applications, we need to estimate propensities. How should we estimate them, and how should we subsequently use them? For example, in econometrics, *matching* propensities were found to be more robust to estimation errors than IPS and its variants.

We saw several examples of combinatorial structured actions or treatments $\mathbf{y}$ throughout the thesis; practical systems also typically have multi-dimensional feedback δ as measures of their performance. Can we exploit the structure of $\mathbf{y}$ and δ in novel ways to design better estimators?

Algorithms The counterfactual learning algorithms we have developed so far rely on efficient implementations of closely related supervised ML algorithms. Can we derive efficient training algorithms without this dependence? Also, can we develop general-purpose optimization techniques (like stochastic gradient descent) that are compatible with broad classes of counterfactual estimators? Finally, can we develop software that makes counterfactual techniques equally easy to apply in practice as supervised learning algorithms?

Focussing more narrowly on the use of data-dependent regularizers, POEM (Chapter 6) showed that empirical variance regularization is crucial for counterfactual learning in BLBF settings. Can we find similar *risk-sensitive* regularizers for the more general problem of off-policy reinforcement learning?

Deployment Success in developing counterfactual methods should be measured by their impact on industrial practice – when practitioners rely less on trial and error with online A/B tests and instead use these techniques. What, and how much, should be logged during system operation to enable subsequent counterfactual analysis?

The concluding message of this thesis is: by imbuing current machine learning practice with causal reasoning, we elevate their principled application beyond prediction to intervene reliably in interactive systems.

APPENDIX A
SUPPLEMENTARY MATERIAL FOR CHAPTER 3

A.1 Proof of Proposition 1

Proposition (Tail Bound for IPS Estimator)**.** *For any given Δ, let P be the independent Bernoulli probabilities of observing each entry of Δ. For any $\hat{\Delta}$, with probability $1-\eta$, the IPS estimator $\hat{R}_{IPS}(\hat{\Delta} \mid P)$ does not deviate from the true $R(\hat{\Delta})$ by more than:*

$$\left|\hat{R}_{IPS}(\hat{\Delta} \mid P) - R(\hat{\Delta})\right| \leq \frac{1}{\mathcal{X} \cdot \mathcal{Y}} \sqrt{\frac{\log \frac{2}{\eta}}{2} \sum_{x,y} \rho_{x,y}^2},$$

where $\rho_{x,y} = \frac{\delta_{x,y}(\Delta,\hat{\Delta})}{P_{x,y}}$ if $P_{x,y} < 1$, and $\rho_{x,y} = 0$ otherwise.

Proof. Hoeffding's inequality states that for independent bounded random variables $Z_1, ..., Z_N$ that take values in intervals of sizes $\rho_1, ..., \rho_N$ with probability 1 and for any $\epsilon > 0$

$$\Pr\left(\left|\sum_k Z_k - \mathbb{E}\left[\sum_k Z_k\right]\right| \geq \epsilon\right) \leq 2\exp\left(\frac{-2\epsilon^2}{\sum_k \rho_k^2}\right).$$

Let $N = \mathcal{X} \cdot \mathcal{Y}$, and let $1 \leq k \leq N$ iterate over every pair (x, y). Defining $\Pr\left(Z_k = \frac{\delta_{x,y}(\Delta,\hat{\Delta})}{P_{x,y}}\right) = P_{x,y}$ and $\Pr(Z_k = 0) = 1 - P_{x,y}$ relates Hoeffding's inequality to the IPS estimator and its expectation, which equals $R(\hat{\Delta})$ as shown in Equation (3.11). This yields

$$\Pr\left(\left|\hat{R}_{IPS}(\hat{\Delta} \mid P) - R(\hat{\Delta})\right| \geq \epsilon\right) \leq 2\exp\left(\frac{-2\epsilon^2 \mathcal{X}^2 \cdot \mathcal{Y}^2}{\sum_{x,y} \rho_{x,y}^2}\right),$$

where $\rho_{x,y}$ is defined as in the statement of the proposition. Solving for ϵ completes the proof. □

A.2 Proof of Theorem 1

Theorem (Propensity-Scored ERM Generalization Error Bound)**.** *For any finite hypothesis space of predictions $\mathcal{H} = \{\hat{\Delta}_1, \ldots \hat{\Delta}_{|\mathcal{H}|}\}$ and loss $0 \leq \delta_{x,y}(\Delta, \hat{\Delta}) \leq M$, the true risk $R(\hat{\Delta}^{ERM})$ of the empirical risk minimizer $\hat{\Delta}^{ERM}$ from $\mathcal{H}$ using the IPS estimator, given training observations O from Δ with independent Bernoulli propensities P, is bounded with probability $1 - \eta$ by:*

$$R(\hat{\Delta}^{ERM}) \quad \leq \quad \hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid P) + \frac{M}{\mathcal{X} \cdot \mathcal{Y}} \sqrt{\frac{\log(2|\mathcal{H}|/\eta)}{2}} \sqrt{\sum_{x,y} \frac{1}{P_{x,y}^2}}.$$

Proof. Making a uniform convergence argument via Hoeffding's inequality and union bound yields:

$$\begin{aligned}
&\Pr\left(\left|R(\hat{\Delta}^{ERM}) - \hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid P)\right| \leq \epsilon\right) \geq 1 - \eta \\
&\Leftarrow \Pr\left(\max_{\hat{\Delta}_i} \left|R(\hat{\Delta}_i) - \hat{R}_{IPS}(\hat{\Delta}_i \mid P)\right| \leq \epsilon\right) \geq 1 - \eta \\
&\Leftrightarrow \Pr\left(\bigvee_{\hat{\Delta}_i} \left|R(\hat{\Delta}_i) - \hat{R}_{IPS}(\hat{\Delta}_i \mid P)\right| \geq \epsilon\right) < \eta \\
&\Leftarrow \sum_{i=1}^{|\mathcal{H}|} \Pr\left(\left|R(\hat{\Delta}_i) - \hat{R}_{IPS}(\hat{\Delta}_i \mid P)\right| \geq \epsilon\right) < \eta \\
&\Leftarrow |\mathcal{H}| \cdot 2\exp\left(\frac{-2\epsilon^2}{\frac{M^2}{\mathcal{X}^2 \cdot \mathcal{Y}^2} \sum_{x,y} \frac{1}{P_{x,y}^2}}\right) < \eta
\end{aligned}$$

Solving the last line for ϵ yields the desired result. □

A.3 Proof of Lemma 2

Lemma (Bias of IPS Estimator under Inaccurate Propensities)**.** *Let P be the marginal probabilities of observing an entry of the rating matrix Δ, and let $\hat{P}$ be the* *estimated propensities such that* $\hat{P}_{x,y} > 0$ *for all* x, y*. The bias of the IPS estimator Equation* (3.10) *using* $\hat{P}$ *is:*

$$\text{bias}\left(\hat{R}_{IPS}(\hat{\Delta} \mid \hat{P})\right) = \mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta} \mid \hat{P})\right] - R(\hat{\Delta}) \quad = \quad \sum_{x,y} \frac{\delta_{x,y}(\Delta, \hat{\Delta})}{\mathcal{X} \cdot \mathcal{Y}} \left[1 - \frac{P_{x,y}}{\hat{P}_{x,y}}\right]. \tag{A.1}$$

Proof. Expanding both terms in the definition of bias yields

$$R(\hat{\Delta}) \quad = \quad \frac{1}{\mathcal{X} \cdot \mathcal{Y}} \sum_{x,y} \delta_{x,y}(\Delta, \hat{\Delta}), \tag{A.2}$$

$$\mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta} \mid \hat{P})\right] \quad = \quad \frac{1}{\mathcal{X} \cdot \mathcal{Y}} \sum_{x,y} \frac{P_{x,y}}{\hat{P}_{x,y}} \delta_{x,y}(\Delta, \hat{\Delta}). \tag{A.3}$$

Rest follows after subtracting line Equation (A.2) from Equation (A.3). □

A.4 Proof of Theorem 3

Theorem. *For any finite hypothesis space of predictions* $\mathcal{H} = \{\hat{\Delta}_1, \ldots \hat{\Delta}_{|\mathcal{H}|}\}$*, the transductive prediction error of the empirical risk minimizer* $\hat{\Delta}^{ERM}$*, using the IPS estimator with estimated propensities* $\hat{P}$ *(*$\hat{P}_{x,y} > 0$*) and given training observations* O *from* Δ *with independent Bernoulli propensities* P*, is bounded by:*

$$\begin{aligned} R(\hat{\Delta}^{ERM}) \quad \leq \quad & \hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P}) + \frac{M}{\mathcal{X} \cdot \mathcal{Y}} \sum_{x,y} \left|1 - \frac{P_{x,y}}{\hat{P}_{x,y}}\right| \\ & + \frac{M}{\mathcal{X} \cdot \mathcal{Y}} \sqrt{\frac{\log(2|\mathcal{H}|/\eta)}{2}} \sqrt{\sum_{x,y} \frac{1}{\hat{P}^2_{x,y}}}. \end{aligned} \tag{A.4}$$

Proof. First, notice that we can write

$$\begin{aligned} R(\hat{\Delta}^{ERM}) &= R(\hat{\Delta}^{ERM}) - \mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P})\right] + \mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P})\right] \\ &= \text{bias}\left(\hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P})\right) + \mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P})\right] \\ &\leq \frac{M}{\mathcal{X} \cdot \mathcal{Y}} \sum_{x,y} \left|1 - \frac{P_{x,y}}{\hat{P}_{x,y}}\right| + \mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P})\right] \end{aligned}$$

which follows from Lemma 2.

We are left to bound the following

$$\begin{aligned} &\Pr\left(\left|\hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P}) - \mathbb{E}_O\left[\hat{R}_{IPS}(\hat{\Delta}^{ERM} \mid \hat{P})\right]\right| \leq \epsilon\right) \\ &\geq 1 - \eta \\ &\Leftarrow |\mathcal{H}| \cdot 2\exp\left(\frac{-2\epsilon^2}{\frac{M^2}{\mathcal{X}^2 \cdot \mathcal{Y}^2} \sum_{x,y} \frac{1}{\hat{P}_{x,y}^2}}\right) < \eta. \end{aligned}$$

The intermediate steps here are analogous to the steps in the proof of Theorem 1 in Appendix A.2. Rearranging the terms gives the stated results. □

A.5 Propensity Estimation via Logistic Regression

In contrast to other discriminative models, logistic regression offers some attractive properties for propensity estimation. For the logistic propensity model, observe that at optimality of the Maximum Likelihood estimate, the following two equations hold:

$$\forall y : \sum_x O_{x,y} = \sum_x \hat{P}_{x,y} \tag{A.5}$$

$$\forall x : \sum_y O_{x,y} = \sum_y \hat{P}_{x,y}. \tag{A.6}$$

In other words, the logistic propensity model is able to learn well-calibrated marginal probabilities.

Proof. The log-likelihood function of the entire model after simplification is:

$$\begin{aligned}\mathcal{L}(O \mid \mathbf{f}, \phi) = &\sum_{(x,y):O_{x,y}=1} \left[\alpha^T \mathbf{f}_{x,y} + \beta_x + \gamma_y\right] \\ &- \sum_{x,y} \log\left[1 + \exp\left(\alpha^T \mathbf{f}_{x,y} + \beta_x + \gamma_y\right)\right].\end{aligned} \tag{A.7}$$

The gradient for bias term β_x (analogously for γ_y) for item y is given as

$$\frac{\partial \mathcal{L}}{\partial \beta_x} = \sum_y O_{x,y} - \sum_y \hat{P}_{x,y}. \tag{A.8}$$

Solving the gradient for zero yields $\sum_y O_{x,y} - \sum_y \hat{P}_{x,y}$. Doing this for each β_x and γ_y yields the stated result. □

APPENDIX B
SUPPLEMENTARY MATERIAL FOR CHAPTER 5

B.1 Proofs of Theorem 4 and Theorem 5

Claim 1. $\mathbf{\Gamma}_{\pi_0,x} = \mathbb{E}_{\pi_0}[\mathbf{1}_{\mathbf{y}}\mathbf{1}_{\mathbf{y}}^T \mid x]$.

Proof. Consider the matrix $\mathbf{1}_{\mathbf{y}}\mathbf{1}_{\mathbf{y}}^T$. Its element in the row indexed (j, y) and column indexed (k, y') equals

$$\mathbf{1}\{\mathbf{y}_j = y, \mathbf{y}_k = y'\} = \begin{cases} \mathbf{1}\{\mathbf{y}_j = y\} & \text{if } j = k \text{ and } a = a', \\ \mathbf{1}\{\mathbf{y}_j = y, \mathbf{y}_k = y'\} & \text{if } j \neq k, \\ 0 & \text{otherwise.} \end{cases}$$

The claim follows by taking a conditional expectation with respect to π_0. □

Proof of Theorem 4. Fix one x. Recall from Section 5.5.1 that

$$V(x, \mathbf{y}) = \mathbf{1}_{\mathbf{y}}^T \phi_x.$$

Let $N = |\operatorname{supp} \pi_0(\cdot \mid x)|$ be the size of the support of $\pi_0(\cdot \mid x)$ and let $\mathbf{M} \in \{0, 1\}^{N \times m\ell}$ denote the binary matrix with rows $\mathbf{1}_{\mathbf{y}}^T$ for each $\mathbf{y} \in \operatorname{supp} \mu(\cdot \mid x)$. Thus $\mathbf{M}\phi_x$ is the vector enumerating $V(x, \mathbf{y})$ over $\mathbf{y}$ for which $\pi_0(\mathbf{y} \mid x) > 0$. Let $\operatorname{Null}(\mathbf{M})$ denote the null space of $\mathbf{M}$ and $\mathbf{\Pi}$ be the projection on $\operatorname{Null}(\mathbf{M})$. Let $\phi_x^\star = (\mathbf{I} - \mathbf{\Pi})\phi_x$. Then clearly, $\mathbf{M}\phi_x = \mathbf{M}\phi_x^\star$, and hence, for any $\mathbf{y} \in \operatorname{supp} \pi_0(\cdot \mid x)$,

$$V(x, \mathbf{y}) = \mathbf{1}_{\mathbf{y}}^T \phi_x^\star \quad . \tag{B.1}$$

We will now show that $\boldsymbol{\phi}^\star_x = \boldsymbol{\Gamma}^\dagger_{\pi_0,x}\boldsymbol{\theta}_{\pi_0,x}$, which will complete the proof. Recall from Section 5.5.1 that

$$\boldsymbol{\theta}_{\pi_0,x} = \boldsymbol{\Gamma}_{\pi_0,x}\boldsymbol{\phi}_x. \tag{B.2}$$

Next note that $\boldsymbol{\Gamma}_{\pi_0,x}$ is symmetric positive semidefinite by Claim 1, so

$$\mathrm{Null}(\boldsymbol{\Gamma}_{\pi_0,x}) = \{\mathbf{v} : \mathbf{v}^T\boldsymbol{\Gamma}_{\pi_0,x}\mathbf{v} = 0\} = \{\mathbf{v} : \mathbf{1}_{\mathbf{y}}^T\mathbf{v} = 0 \text{ for all } \mathbf{y} \in \operatorname{supp}\pi_0(\cdot \mid x)\} = \mathrm{Null}(\mathbf{M})$$

where the first step follows by positive semi definiteness of $\boldsymbol{\Gamma}_{\pi_0,x}$, the second step is from the expansion of $\boldsymbol{\Gamma}_{\pi_0,x}$ as in Claim 1, and the final step from the definition of $\mathbf{M}$. Since $\mathrm{Null}(\boldsymbol{\Gamma}_{\pi_0,x}) = \mathrm{Null}(\mathbf{M})$, we have from Equation (B.2) that $\boldsymbol{\theta}_x = \boldsymbol{\Gamma}_{\pi_0,x}\boldsymbol{\phi}^\star_x$. Importantly, this also implies $\boldsymbol{\phi}^\star_x \perp \mathrm{Null}(\boldsymbol{\Gamma}_{\pi_0,x})$. From the definition of pseudoinverse,

$$\boldsymbol{\Gamma}^\dagger_{\pi_0,x}\boldsymbol{\theta}_x = \boldsymbol{\phi}^\star_x.$$

This proves Theorem 4, since for any $\mathbf{y}$ with $\pi_0(\mathbf{y} \mid x) > 0$, we argued that $V(x,\mathbf{y}) = \mathbf{1}_{\mathbf{y}}^T\boldsymbol{\phi}^\star_x = \mathbf{1}_{\mathbf{y}}^T\boldsymbol{\Gamma}^\dagger_{\pi_0,x}\boldsymbol{\theta}_x$. □

Proof of Theorem 5. Note that it suffices to analyze the expectation of a single term (due to linearity of expectation) in the estimator, that is

$$\sum_{\mathbf{y}\in Y(x_i)} \pi(\mathbf{y} \mid x_i)\mathbf{1}_{\mathbf{y}}^T\boldsymbol{\Gamma}^\dagger_{\pi_0,x_i}\hat{\boldsymbol{\theta}}_i.$$

First note that $\mathbb{E}_{(\mathbf{y}_i,\delta_i)\sim\pi_0(\cdot,\cdot|x_i)}\left[\hat{\boldsymbol{\theta}}_i\right] = \boldsymbol{\theta}_{x_i}$, because

$$\mathbb{E}_{(\mathbf{y}_i,\delta_i)\sim\pi_0(\cdot,\cdot|x_i)}\left[\hat{\theta}_i(j,y)\right] = \mathbb{E}_{(\mathbf{y}_i,\delta_i)\sim\pi_0(\cdot,\cdot|x_i)}\left[\delta_i\mathbf{1}\{\mathbf{y}_j = y\}\right] = \theta_{x_i}(j,y).$$

The remainder follows by Theorem 4:

$$\begin{aligned}\mathbb{E}\left[\sum_{\mathbf{y}\in Y(x_i)} \pi(\mathbf{y}\mid x_i)\mathbf{1}_{\mathbf{y}}^T\boldsymbol{\Gamma}_{\pi_0,x_i}^{\dagger}\hat{\boldsymbol{\theta}}_i\right] &= \mathbb{E}_{x_i\sim\Pr(X)}\left[\sum_{\mathbf{y}\in Y(x_i)} \pi(\mathbf{y}\mid x_i)\mathbf{1}_{\mathbf{y}}^T\boldsymbol{\Gamma}_{\pi_0,x_i}^{\dagger}\mathbb{E}_{(\mathbf{y}_i,\delta_i)\sim\pi_0(\cdot,\cdot\mid x_i)}[\hat{\boldsymbol{\theta}}_i]\right]\\ &= \mathbb{E}_{x_i\sim\Pr(X)}\left[\sum_{\mathbf{y}\in Y(x_i)} \pi(\mathbf{y}\mid x_i)\mathbf{1}_{\mathbf{y}}^T\boldsymbol{\Gamma}_{\pi_0,x_i}^{\dagger}\boldsymbol{\theta}_{x_i}\right]\\ &= \mathbb{E}_{x_i\sim\Pr(X)}\left[\sum_{\mathbf{y}\in Y(x_i)} \pi(\mathbf{y}\mid x_i)V(x_i,\mathbf{y})\right] = V(\pi). \qquad \square\end{aligned}$$

B.2 Proof of Theorem 6

Proof. The proof applies Bernstein's inequality to the centered sum

$$\sum_{i=1}^{n}\left[\mathbf{q}_{\pi,x_i}^T\boldsymbol{\Gamma}_{\pi_0,x_i}^{\dagger}\hat{\boldsymbol{\theta}}_i - V(\pi)\right] .$$

The fact that this quantity is centered is directly from Theorem 5. We must compute both the second moment and the range to apply Bernstein's inequality. By independence of the n samples, we can focus on just one term, so we will drop the subscript i. First, bound the variance:

$$\begin{aligned}\boldsymbol{Var}\left[\mathbf{q}_{\pi,x}^T\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\hat{\boldsymbol{\theta}}\right] &\le \mathbb{E}_{\pi_0}\left[\left(\mathbf{q}_{\pi,x}^T\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\hat{\boldsymbol{\theta}}\right)^2\right]\\ &= \mathbb{E}_{\pi_0}\left[\left(\mathbf{q}_{\pi,x}^T\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\delta\mathbf{1}_{\mathbf{y}}\right)^2\right]\\ &\le \mathbb{E}_{\pi_0}\left[\left(\mathbf{q}_{\pi,x}^T\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\mathbf{1}_{\mathbf{y}}\right)^2\right]\\ &= \mathbb{E}_{x\sim\Pr(X)}\left[\mathbf{q}_{\pi,x}^T\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\mathbb{E}_{\mathbf{y}\sim\pi_0(\cdot\mid x)}\left[\mathbf{1}_{\mathbf{y}}\mathbf{1}_{\mathbf{y}}^T\right]\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\mathbf{q}_{\pi,x}\right]\\ &= \mathbb{E}_{x\sim\Pr(X)}\left[\mathbf{q}_{\pi,x}^T\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\boldsymbol{\Gamma}_{\pi_0,x}\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\mathbf{q}_{\pi,x}\right]\\ &= \mathbb{E}_{x\sim\Pr(X)}\left[\mathbf{q}_{\pi,x}^T\boldsymbol{\Gamma}_{\pi_0,x}^{\dagger}\mathbf{q}_{\pi,x}\right]\\ &= \sigma^2.\end{aligned}$$

Thus the per-term variance is at most σ^2. We now bound the range, again focusing on one term,

$$
\begin{aligned}
\left|\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \hat{\boldsymbol{\theta}} - V(\pi)\right| &\leq \left|\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \hat{\boldsymbol{\theta}}\right| + 1 \\
&= \left|\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \delta \mathbf{1}_{\mathbf{y}}\right| + 1 \\
&\leq \left|\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}}\right| + 1 \\
&\leq \rho + 1.
\end{aligned}
$$

The first line here is the triangle inequality, coupled with the fact that since rewards are bounded in $[-1, 1]$, so is $V(\pi)$. The second line is from the definition of $\hat{\boldsymbol{\theta}}$, while the third follows because $\delta \in [-1, 1]$. The final line follows from the definition of ρ.

Now, we may apply Bernstein's inequality, which says that for any $\eta \in (0, 1)$, with probability at least $1 - \eta$,

$$
\left|\sum_{i=1}^{n} \left[\mathbf{q}_{\pi,x_i}^T \mathbf{\Gamma}_{\pi_0,x_i}^{\dagger} \hat{\boldsymbol{\theta}}_i - V(\pi)\right]\right| \leq \sqrt{2n\sigma^2 \ln(2/\eta)} + \frac{2(\rho+1)\ln(2/\eta)}{3}.
$$

The theorem follows by dividing by n. □

B.3 Pseudoinverse Estimator when Logging Policy and Target Policy Coincide

In this section we show that when the target policy coincides with logging (i.e., $\pi = \pi_0$), we have $\sigma^2 = \rho = 1$, i.e., the bound of Theorem 6 is independent of the number of actions and slots. Indeed, in Claim 3 we will see that the estimator actually simplifies to taking an empirical average of rewards which are bounded in $[-1, 1]$. Before proving Claim 3 we prove one supporting claim:

Claim 2. *For any policy π_0 and x, we have $\mathbf{q}_{\pi_0,x}^T \mathbf{\Gamma}_{\pi_0,x}^\dagger \mathbf{1}_\mathbf{y} = 1$ for all $\mathbf{y} \in \operatorname{supp} \pi_0(\cdot \mid x)$.*

Proof. To simplify the exposition, write $\mathbf{q}$ and $\mathbf{\Gamma}$ instead of a more verbose $\mathbf{q}_{\pi_0,x}$ and $\mathbf{\Gamma}_{\pi_0,x}$. The bulk of the proof is in deriving an explicit expression for $\mathbf{\Gamma}^\dagger$. We begin by expressing $\mathbf{\Gamma}$ in a suitable basis. Since $\mathbf{\Gamma}$ is the matrix of second moments and $\mathbf{q}$ is the vector of first moments of $\mathbf{1}_\mathbf{y}$, $\mathbf{\Gamma}$ can be written as

$$\mathbf{\Gamma} = \mathbf{V} + \mathbf{q}\mathbf{q}^T$$

where $\mathbf{V}$ is the covariance matrix of $\mathbf{1}_\mathbf{y}$, i.e., $\mathbf{V} := \mathbb{E}_{\mathbf{y}\sim\pi_0(\cdot|x)}\left[(\mathbf{1}_\mathbf{y} - \mathbf{q})(\mathbf{1}_\mathbf{y} - \mathbf{q})^T\right]$. Assume that the rank of $\mathbf{V}$ is r and consider the eigenvalue decomposition of $\mathbf{V}$

$$\mathbf{V} = \sum_{i=1}^{r} \lambda_i \mathbf{u}_i \mathbf{u}_i^T = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T,$$

where $\lambda_i > 0$ and vectors $\mathbf{u}_i$ are orthonormal; we have grouped the eigenvalues into the diagonal matrix $\mathbf{\Lambda} := \operatorname{diag}(\lambda_1, \dots \lambda_r)$ and eigenvectors into the matrix $\mathbf{U} := (\mathbf{u}_1\ \mathbf{u}_2\ \dots\ \mathbf{u}_r)$.

We next argue that $\mathbf{q} \notin \operatorname{Range}(\mathbf{V})$. To see this, note that the all-ones-vector $\mathbf{1}$ is in the null space of $\mathbf{V}$ because, for any valid slate $\mathbf{y}$, we have $\mathbf{1}_\mathbf{y}^T\mathbf{1} = \ell$ and thus also for the convex combination $\mathbf{q}$ we have $\mathbf{q}^T\mathbf{1} = \ell$, which means that

$$\mathbf{1}^T\mathbf{V}\mathbf{1} = \mathbb{E}_{\mathbf{y}\sim\pi_0(\cdot|x)}\left[\mathbf{1}^T(\mathbf{1}_\mathbf{y} - \mathbf{q})(\mathbf{1}_\mathbf{y} - \mathbf{q})^T\mathbf{1}\right] = 0.$$

Now, since $\mathbf{1} \perp \operatorname{Range}(\mathbf{V})$ and $\mathbf{q}^T\mathbf{1} = \ell$, we have that $\mathbf{q} \notin \operatorname{Range}(\mathbf{V})$. In particular, we can write $\mathbf{q}$ in the form

$$\mathbf{q} = \sum_{i=1}^{r} \beta_i \mathbf{u}_i + \alpha\mathbf{n} = \begin{pmatrix}\mathbf{U} & \mathbf{n}\end{pmatrix}\begin{pmatrix}\beta \\ \alpha\end{pmatrix} \tag{B.3}$$

where $\alpha \neq 0$ and $\mathbf{n} \in \operatorname{Null}(\mathbf{V})$ is a unit vector. Note that $\mathbf{n} \perp \mathbf{u}_i$ since $\mathbf{u}_i \perp \operatorname{Null}(\mathbf{V})$. Thus, the second moment matrix $\mathbf{\Gamma}$ can be written as

$$\mathbf{\Gamma} = \mathbf{V} + \mathbf{q}\mathbf{q}^T = \begin{pmatrix}\mathbf{U} & \mathbf{n}\end{pmatrix}\begin{pmatrix}\mathbf{\Lambda} + \boldsymbol{\beta}\boldsymbol{\beta}^T & \alpha\boldsymbol{\beta} \\ \alpha\boldsymbol{\beta}^T & \alpha^2\end{pmatrix}\begin{pmatrix}\mathbf{U} & \mathbf{n}\end{pmatrix}^T . \tag{B.4}$$

Let $\mathbf{Q} \in \mathbb{R}^{(r+1)\times(r+1)}$ denote the middle matrix in Equation (B.4):

$$\mathbf{Q} := \begin{pmatrix}\mathbf{\Lambda} + \boldsymbol{\beta}\boldsymbol{\beta}^T & \alpha\boldsymbol{\beta} \\ \alpha\boldsymbol{\beta}^T & \alpha^2\end{pmatrix}. \tag{B.5}$$

This matrix is a representation of $\mathbf{\Gamma}$ with respect to the basis $\{\mathbf{u}_1, \ldots \mathbf{u}_r, \mathbf{n}\}$. Since $\mathbf{q} \notin \text{Range}(\mathbf{V})$, the rank of $\mathbf{\Gamma}$ and that of $\mathbf{Q}$ is $r+1$. Thus, $\mathbf{Q}$ is invertible and

$$\mathbf{\Gamma}^\dagger = \begin{pmatrix}\mathbf{U} & \mathbf{n}\end{pmatrix}\mathbf{Q}^{-1}\begin{pmatrix}\mathbf{U} & \mathbf{n}\end{pmatrix}^T . \tag{B.6}$$

To obtain $\mathbf{Q}^{-1}$, we use the following identity [88]:

$$\begin{pmatrix}\mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22}\end{pmatrix}^{-1} = \begin{pmatrix}\mathbf{M}^{-1} & -\mathbf{M}^{-1}\mathbf{A}_{12}\mathbf{A}_{22}^{-1} \\ -\mathbf{A}_{22}^{-1}\mathbf{A}_{21}\mathbf{M}^{-1} & \mathbf{A}_{22}^{-1}\mathbf{A}_{21}\mathbf{M}^{-1}\mathbf{A}_{12}\mathbf{A}_{22}^{-1} + \mathbf{A}_{22}^{-1}\end{pmatrix}, \tag{B.7}$$

where $\mathbf{M} := \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}$ is the Schur complement of $\mathbf{A}_{22}$. The identity of Equation (B.7) holds whenever $\mathbf{A}_{22}$ and its Schur complement $\mathbf{M}$ are both invertible. In Equation (B.5), we have $\mathbf{A}_{22} = \alpha^2 > 0$ and

$$\mathbf{M} = (\mathbf{\Lambda} + \boldsymbol{\beta}\boldsymbol{\beta}^T) - (\alpha\boldsymbol{\beta})\alpha^{-2}(\alpha\boldsymbol{\beta}^T) = \mathbf{\Lambda},$$

so Equation (B.7) can be applied to obtain $\mathbf{Q}^{-1}$:

$$\mathbf{Q}^{-1} = \begin{pmatrix} \mathbf{\Lambda} + \boldsymbol{\beta}\boldsymbol{\beta}^T & \alpha\boldsymbol{\beta} \\ \alpha\boldsymbol{\beta}^T & \alpha^2 \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{\Lambda}^{-1} & -\mathbf{\Lambda}^{-1}(\alpha\boldsymbol{\beta})\alpha^{-2} \\ -\alpha^{-2}(\alpha\boldsymbol{\beta}^T)\mathbf{\Lambda}^{-1} & \alpha^{-2}(\alpha\boldsymbol{\beta}^T)\mathbf{\Lambda}^{-1}(\alpha\boldsymbol{\beta})\alpha^{-2} + \alpha^{-2} \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{\Lambda}^{-1} & -\alpha^{-1}\mathbf{\Lambda}^{-1}\boldsymbol{\beta} \\ -\alpha^{-1}\boldsymbol{\beta}^T\mathbf{\Lambda}^{-1} & \alpha^{-2}(1 + \boldsymbol{\beta}^T\mathbf{\Lambda}^{-1}\boldsymbol{\beta}) \end{pmatrix}. \tag{B.8}$$

Next, we will evaluate $\mathbf{\Gamma}^{\dagger}\mathbf{q}$, using the factorizations in Equations (B.6) and (B.3), and substituting Equation (B.8) for $\mathbf{Q}^{-1}$:

$$\begin{aligned}
\mathbf{\Gamma}^{\dagger}\mathbf{q} &= \begin{pmatrix} \mathbf{U} & \mathbf{n} \end{pmatrix} \mathbf{Q}^{-1} \begin{pmatrix} \mathbf{U} & \mathbf{n} \end{pmatrix}^T \begin{pmatrix} \mathbf{U} & \mathbf{n} \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta} \\ \alpha \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{U} & \mathbf{n} \end{pmatrix} \mathbf{Q}^{-1} \begin{pmatrix} \boldsymbol{\beta} \\ \alpha \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{U} & \mathbf{n} \end{pmatrix} \begin{pmatrix} \mathbf{\Lambda}^{-1}\boldsymbol{\beta} - \mathbf{\Lambda}^{-1}\boldsymbol{\beta} \\ -\alpha^{-1}\boldsymbol{\beta}^T\mathbf{\Lambda}^{-1}\boldsymbol{\beta} + \alpha^{-1}(1 + \boldsymbol{\beta}^T\mathbf{\Lambda}^{-1}\boldsymbol{\beta}) \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{U} & \mathbf{n} \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ \alpha^{-1} \end{pmatrix} \\
&= \alpha^{-1}\mathbf{n}.
\end{aligned}$$

To finish the proof, we consider any $\mathbf{y} \in \operatorname{supp}\pi_0(\cdot \mid x)$ and consider the decomposition of $\mathbf{1_y}$ in the basis $\{\mathbf{u}_1, \ldots \mathbf{u}_r, \mathbf{n}\}$. First, $(\mathbf{1_y} - \mathbf{q}) \perp \operatorname{Null}(\mathbf{V})$ since

$$\operatorname{Null}(\mathbf{V}) = \{\mathbf{v} : \mathbb{E}_{\mathbf{y}\sim\pi_0(\cdot|x)}\left[\left((\mathbf{1_y} - \mathbf{q})^T\mathbf{v}\right)^2\right] = 0\} = \{\mathbf{v} : (\mathbf{1_y} - \mathbf{q})^T\mathbf{v} = 0 \text{ for all } \mathbf{y} \in \operatorname{supp}\pi_0(\cdot \mid x)\}.$$

Thus, $(\mathbf{1_y} - \mathbf{q}) \in \operatorname{Range}(\mathbf{V})$. Therefore, we obtain

$$\mathbf{q}^T\mathbf{\Gamma}^{\dagger}_{\pi_0,x}\mathbf{1_y} = \alpha^{-1}\mathbf{n}^T\mathbf{1_y} = \alpha^{-1}\mathbf{n}^T(\mathbf{1_y} - \mathbf{q}) + \alpha^{-1}\mathbf{n}^T\mathbf{q} = 0 + \alpha^{-1}\alpha = 1,$$

where the third equality follows because $(\mathbf{1}_\mathbf{y} - \mathbf{q}) \perp \mathbf{n}$ and the decomposition in Equation (B.3) shows that $\mathbf{n}^T\mathbf{q} = \alpha$. □

Claim 3. *If $\pi = \pi_0$ then $\sigma^2 = \rho = 1$ and $\hat{V}_{\mathrm{PI}}(\pi) = \hat{V}_{\mathrm{PI}}(\pi_0) = \frac{1}{n}\sum_{i=1}^n \delta_i$.*

Proof. From Claim 2

$$\mathbf{q}_{\pi_0,x}^T \mathbf{\Gamma}_{\pi_0,x}^\dagger \mathbf{q}_{\pi_0,x} = \mathbb{E}_{\mathbf{y}\sim\pi_0(\cdot|x)}[\mathbf{q}_{\pi_0,x}^T \mathbf{\Gamma}_{\pi_0,x}^\dagger \mathbf{1}_\mathbf{y}] = 1.$$

Taking expectation over x then yields $\sigma^2 = 1$. Equally, $\rho = 1$ follows immediately from plugging Claim 2 into the definition of ρ. The final statement of Claim 3 follows by applying Claim 2 to a single term of $\hat{V}_{\mathrm{PI}}(\pi_0)$:

$$\mathbf{q}_{\pi_0,x_i}^T \mathbf{\Gamma}_{\pi_0,x_i}^\dagger \delta_i \mathbf{1}_{\mathbf{y}_i} = \delta_i.$$ □

B.4 Proof of Proposition 2

For a given logging policy π_0 and context x, let

$$\bar{\rho}_{\pi_0,x} := \sup_{\mathbf{y}\in\operatorname{supp}\pi_0(\cdot|x)} \mathbf{1}_\mathbf{y}^T \mathbf{\Gamma}_{\pi_0,x}^\dagger \mathbf{1}_\mathbf{y}.$$

This quantity can be viewed as a norm of $\mathbf{\Gamma}_{\pi_0,x}^\dagger$ with respect to the set of slates chosen by π_0 with non-zero probability. It can be used to bound σ^2 and ρ, and thus to bound an error of $\hat{V}_{\mathrm{PI}}$:

Proposition 4. *For any logging policy π_0 and target policy π that is absolutely continuous with respect to π_0, we have*

$$\sigma^2 \leq \rho \leq \sup_x \bar{\rho}_{\pi_0,x}.$$

Proof. Recall that

$$\sigma^2 = \mathbb{E}_{x\sim\Pr(X)}\left[\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{q}_{\pi,x}\right], \quad \rho = \sup_x \sup_{\mathbf{y}\in\operatorname{supp}\pi_0(\cdot|x)} \left|\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}}\right|.$$

To see that $\sigma^2 \le \rho$ note that

$$\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{q}_{\pi,x} = \mathbb{E}_{\mathbf{y}\sim\pi(\cdot|x)}\left[\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}}\right] \le \rho$$

where the last inequality follows by the absolute continuity of π with respect to π_0. It remains to show that $\rho \le \sup_x \bar{\rho}_{\pi_0,x}$.

First, by positive semi-definiteness of $\mathbf{\Gamma}_{\pi_0,x}^{\dagger}$ and from the definition of $\bar{\rho}_{\pi_0,x}$, we have that for any slates $\mathbf{y}, \mathbf{y}' \in \operatorname{supp}\pi_0(\cdot \mid x)$ and any $z \in \{-1, 1\}$

$$z\mathbf{1}_{\mathbf{y}'}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}} \le \frac{\mathbf{1}_{\mathbf{y}}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}} + \mathbf{1}_{\mathbf{y}'}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}'}}{2} \le \max\{\mathbf{1}_{\mathbf{y}}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}}, \mathbf{1}_{\mathbf{y}'}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}'}\} \le \bar{\rho}_{\pi_0,x}.$$

Therefore, for any π absolutely continuous with respect to π_0 and any $\mathbf{y} \in \operatorname{supp}\pi_0(\cdot \mid x)$, we have

$$\left|\mathbf{q}_{\pi,x}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}}\right| = \max_{z\in\{-1,1\}} \mathbb{E}_{\mathbf{y}'\sim\pi(\cdot|x)}\left[z\mathbf{1}_{\mathbf{y}'}^T \mathbf{\Gamma}_{\pi_0,x}^{\dagger} \mathbf{1}_{\mathbf{y}}\right] \le \bar{\rho}_{\pi_0,x}.$$

Taking a supremum over x and $\mathbf{y} \in \operatorname{supp}\pi_0(\cdot \mid x)$, we obtain $\rho \le \sup_x \bar{\rho}_{\pi_0,x}$. □

We next derive bounds on $\bar{\rho}_{\pi_0,x}$ for uniformly-random policies in the ranking and cartesian product examples. Then we prove a translation theorem, which allows translating of the bound for uniform distribution into a bound for κ-uniform distributions. Finally, we put these results together to prove Proposition 2.

B.4.1 Bounds for Uniform Distributions

Let $\mathbf{1}_j \in \mathbb{R}^{\ell m}$ be the vector that is all-ones on the actions in the j-th position and zeros elsewhere. Similarly, let $\mathbf{1}_y \in \mathbb{R}^{\ell m}$ be the vector that is all-ones on the action

y in all positions and zeros elsewhere. Finally, let $\mathbf{1} \in \mathbb{R}^{\ell m}$ be the all-ones vector. We also use $\mathbf{I}_j = \operatorname{diag}(\mathbf{1}_j)$ to denote the diagonal matrix with all-ones on the actions in the j-th position and zeros elsewhere.

Proposition 5. *Consider the product slate space where $Y(x) = Y_1(x) \times \cdots \times Y_\ell(x)$ with $|Y_j(x)| = m_j$. Let ν be the uniform exploration policy, i.e., $\nu(\mathbf{y} \mid x) = 1/|Y(x)|$. Then $\bar{\rho}_{\nu,x} = \sum_j m_j - \ell + 1$ and*

$$\mathbf{\Gamma}^{\dagger}_{\nu,x} = \sum_{j=1}^{\ell} \left(m_j \mathbf{I}_j - \mathbf{1}_j \mathbf{1}_j^T \right) + \left(\sum_{j=1}^{\ell} \frac{1}{m_j} \right)^{-2} \sum_{j,k} \frac{\mathbf{1}_j \mathbf{1}_k}{m_j m_k}.$$

For any policy π, any $\mathbf{y} \in Y(x)$, and any $\delta \in [-1, 1]$ we then have

$$\mathbf{q}^T_{\pi,x} \mathbf{\Gamma}^{\dagger}_{\nu,x} \delta \mathbf{1}_{\mathbf{y}} = \delta \cdot \left[\sum_{j=1}^{\ell} \frac{\pi(\mathbf{y}_j \mid x)}{1/m_j} - \ell + 1 \right]. \tag{B.9}$$

Proof. Throughout the proof we will write $\mathbf{\Gamma}$ instead of the more verbose $\mathbf{\Gamma}_{\nu,x}$ and similarly $\bar{\rho}$ instead of $\bar{\rho}_{\nu,x}$. We will construct an explicit eigendecomposition of $\mathbf{\Gamma}$, which will immediately yield $\mathbf{\Gamma}^{\dagger}$. The remaining statements will follow by a direct calculation. From the definition of $\mathbf{\Gamma}$, we obtain

$$\mathbf{\Gamma} = \sum_{j=1}^{\ell} \frac{\mathbf{I}_j}{m_j} + \sum_{j,k} \frac{\mathbf{1}_j \mathbf{1}_k^T}{m_j m_k} - \sum_{j} \frac{\mathbf{1}_j \mathbf{1}_j^T}{m_j^2}. \tag{B.10}$$

Let $\mathbf{v} = \sum_j \mathbf{1}_j / m_j$ so that the second term on the right-hand side of Equation (B.10) corresponds to $\mathbf{v}\mathbf{v}^T$. Thus, we can write

$$\mathbf{\Gamma} = \|\mathbf{v}\|_2^2 \cdot \frac{\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|_2^2} + \sum_{j=1}^{\ell} \frac{1}{m_j} \left(\mathbf{I}_j - \frac{\mathbf{1}_j \mathbf{1}_j^T}{m_j} \right). \tag{B.11}$$

We argue that this constitutes an eigendecomposition. Let $\mathbf{P}_j := \mathbf{I}_j - \mathbf{1}_j \mathbf{1}_j^T / m_j$ denote the terms appearing in the sum on the right-hand side of Equation (B.11).

Note that $\mathbf{P}_j$'s are projection matrices, i.e., their eigenvalues are in $\{0, 1\}$. Moreover, their ranges are orthogonal to each other, because Range($\mathbf{P}_j$) is a subset of the span of the coordinates corresponding to the slot j. Finally, note that $\mathbf{v}$ is orthogonal to all of the ranges, because

$$\mathbf{v}^T\mathbf{P}_j\mathbf{v} = \mathbf{v}^T\mathbf{I}_j\mathbf{v} - (\mathbf{v}^T\mathbf{1}_j)^2/m_j = 1/m_j - 1/m_j = 0.$$

This shows that Equation (B.11) is an eigendecomposition of $\boldsymbol{\Gamma}$, so

$$\begin{aligned}
\boldsymbol{\Gamma}^\dagger &= \|\mathbf{v}\|_2^{-2} \cdot \frac{\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|_2^2} + \sum_{j=1}^{\ell} m_j \left(\mathbf{I}_j - \frac{\mathbf{1}_j\mathbf{1}_j^T}{m_j}\right) \qquad \text{(B.12)}\\
&= \|\mathbf{v}\|_2^{-4} \cdot \mathbf{v}\mathbf{v}^T + \sum_{j=1}^{\ell} \left(m_j\mathbf{I}_j - \mathbf{1}_j\mathbf{1}_j^T\right)\\
&= \left(\sum_{j=1}^{\ell} \frac{1}{m_j}\right)^{-2} \sum_{j,k} \frac{\mathbf{1}_j\mathbf{1}_k^T}{m_j m_k} + \sum_{j=1}^{\ell} \left(m_j\mathbf{I}_j - \mathbf{1}_j\mathbf{1}_j^T\right),
\end{aligned}$$

where the last equality follows from the definition of $\mathbf{v}$. It remains to derive $\bar{\rho}$ and Equation (B.9). Both will follow by analyzing the expression $\mathbf{1}_{\mathbf{y}'}^T\boldsymbol{\Gamma}^\dagger\mathbf{1}_{\mathbf{y}}$ for $\mathbf{y}, \mathbf{y}' \in Y(x)$. To begin, note that $\mathbf{1}_j^T\mathbf{1}_{\mathbf{y}} = 1$ since any valid slate chooses exactly one action in each position. Thus,

$$\begin{aligned}
\mathbf{1}_{\mathbf{y}'}^T\boldsymbol{\Gamma}^\dagger\mathbf{1}_{\mathbf{y}} &= \left(\sum_{j=1}^{\ell} \frac{1}{m_j}\right)^{-2} \sum_{j,k} \frac{(\mathbf{1}_{\mathbf{y}'}^T\mathbf{1}_j)(\mathbf{1}_k^T\mathbf{1}_{\mathbf{y}})}{m_j m_k} + \sum_{j=1}^{\ell}(m_j\mathbf{1}_{\mathbf{y}'}^T\mathbf{I}_j\mathbf{1}_{\mathbf{y}} - (\mathbf{1}_{\mathbf{y}'}^T\mathbf{1}_j)(\mathbf{1}_j^T\mathbf{1}_{\mathbf{y}}))\\
&= \left(\sum_{j=1}^{\ell} \frac{1}{m_j}\right)^{-2} \sum_{j,k} \frac{1}{m_j m_k} + \sum_{j=1}^{\ell}(m_j\mathbf{1}\{\mathbf{y}'_j = \mathbf{y}_j\} - 1)\\
&= \left(\sum_{j=1}^{\ell} \frac{1}{m_j}\right)^{-2} \left(\sum_{j=1}^{\ell} \frac{1}{m_j}\right)^{2} + \sum_{j=1}^{\ell} \frac{\mathbf{1}\{\mathbf{y}'_j = \mathbf{y}_j\}}{1/m_j} - \ell\\
&= 1 + \sum_{j=1}^{\ell} \frac{\mathbf{1}\{\mathbf{y}'_j = \mathbf{y}_j\}}{1/m_j} - \ell.
\end{aligned}$$

Now the value of $\bar{\rho}$ follows by setting $\mathbf{y}' = \mathbf{y}$, and Equation (B.9) follows by taking an expectation over $\mathbf{y}' \sim \pi(\cdot \mid x)$. □

Proposition 6. *Consider the ranking setting where for each x there is a set $Y(x)$ such that $Y_j(x) = Y(x)$ and where all slates $\mathbf{y} \in Y(x)^\ell$ without repetitions are legal. Let ν denote the uniform exploration policy. If $\ell < m$, then $\bar{\rho}_{\nu,x} = m\ell - \ell + 1$ and*

$$\mathbf{\Gamma}^\dagger_{\nu,x} = \left(\frac{1}{\ell^2} - \frac{m-1}{m(m-\ell)}\right) \cdot \mathbf{1}\mathbf{1}^T + (m-1)\mathbf{I} - \frac{m-1}{m}\sum_j \mathbf{1}_j\mathbf{1}_j^T + \frac{m-1}{m-\ell}\sum_y \mathbf{1}_y\mathbf{1}_y^T,$$

and for $\ell = m$, we have $\bar{\rho}_{\nu,x} = m^2 - 2m + 2$ and

$$\mathbf{\Gamma}^\dagger_{\nu,x} = \frac{1}{m} \cdot \mathbf{1}\mathbf{1}^T + (m-1)\mathbf{I} - \frac{m-1}{m}\sum_j \mathbf{1}_j\mathbf{1}_j^T - \frac{m-1}{m}\sum_y \mathbf{1}_y\mathbf{1}_y^T.$$

For $\ell = m$, we have for any policy π, any $\mathbf{y} \in Y(x)$, and any $\delta \in [-1, 1]$ that

$$\mathbf{q}^T_{\pi,x}\mathbf{\Gamma}^\dagger_{\nu,x}\delta\mathbf{1}_{\mathbf{y}} = \delta \cdot \left[\sum_{j=1}^{\ell} \frac{\pi(\mathbf{y}_j \mid x)}{1/(m-1)} - m + 2\right]. \tag{B.13}$$

Proof. Throughout the proof we will write $\mathbf{\Gamma}$ instead of the more verbose $\mathbf{\Gamma}_{\nu,x}$. Note that for ranking and the uniform distribution we have

$$\Gamma(j, y; k, y') = \begin{cases} \frac{1}{m} & \text{if } j = k \text{ and } y = y' \\ \frac{1}{m(m-1)} & \text{if } j \neq k \text{ and } y \neq y' \\ 0 & \text{otherwise.} \end{cases}$$

Thus, for any $\mathbf{z}$

$$\begin{aligned}\mathbf{z}^T\mathbf{\Gamma}\mathbf{z} &= \sum_{j,y}\frac{z_{j,y}^2}{m} + \frac{1}{m(m-1)}\sum_{j\neq k, y\neq y'} z_{j,y}z_{k,y'} \\ &= \frac{1}{m}\|\mathbf{z}\|_2^2 + \frac{1}{m(m-1)}\left((\mathbf{z}^T\mathbf{1})^2 - \sum_j(\mathbf{z}^T\mathbf{1}_j)^2 - \sum_y(\mathbf{z}^T\mathbf{1}_y)^2 + \|\mathbf{z}\|_2^2\right) \\ &= \frac{1}{m(m-1)}\left((\mathbf{z}^T\mathbf{1})^2 - \sum_j(\mathbf{z}^T\mathbf{1}_j)^2 - \sum_y(\mathbf{z}^T\mathbf{1}_y)^2 + m\|\mathbf{z}\|_2^2\right). \qquad \text{(B.14)}\end{aligned}$$

Let $\mathbf{1}_{\mathcal{J}} \in \mathbb{R}^\ell$ and $\mathbf{1}_{\mathcal{Y}} \in \mathbb{R}^m$ be all-ones vectors in the respective spaces and $\mathbf{I}_{\mathcal{J}} \in \mathbb{R}^{\ell\times\ell}$ and $\mathbf{I}_{\mathcal{Y}} \in \mathbb{R}^{m\times m}$ be identity matrices in the respective spaces. We can rewrite the quadratic form described by $\mathbf{\Gamma}$ as

$$\begin{aligned}m(m-1)\mathbf{\Gamma} &= \mathbf{1}\mathbf{1}^T - \sum_j \mathbf{1}_j\mathbf{1}_j^T - \sum_y \mathbf{1}_y\mathbf{1}_y^T + m\mathbf{I} \\ &= (\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T)\otimes(\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T) - \mathbf{I}_{\mathcal{J}}\otimes(\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T) - (\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T)\otimes\mathbf{I}_{\mathcal{Y}} + m\cdot\mathbf{I}_{\mathcal{J}}\otimes\mathbf{I}_{\mathcal{Y}} \\ &= \ell m\cdot\frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\otimes\frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m} - m\cdot\mathbf{I}_{\mathcal{J}}\otimes\frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m} - \ell\cdot\frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\otimes\mathbf{I}_{\mathcal{Y}} + m\cdot\mathbf{I}_{\mathcal{J}}\otimes\mathbf{I}_{\mathcal{Y}} \\ &= \ell(m-1)\cdot\frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\otimes\frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m} - m\cdot\mathbf{I}_{\mathcal{J}}\otimes\left(\frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m} - \mathbf{I}_{\mathcal{Y}}\right) - \ell\cdot\frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\otimes\left(\mathbf{I}_{\mathcal{Y}} - \frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m}\right) \\ &= \ell(m-1)\cdot\frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\otimes\frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m} \\ &\quad + m\cdot\left(\mathbf{I}_{\mathcal{J}} - \frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\right)\otimes\left(\mathbf{I}_{\mathcal{Y}} - \frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m}\right) + (m-\ell)\cdot\frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\otimes\left(\mathbf{I}_{\mathcal{Y}} - \frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m}\right).\end{aligned} \qquad \text{(B.15)}$$

Next, we would like to argue that Equation (B.15) is an eigendecomposition. For this, we just need to show that each of the three Kronecker products in Equation (B.15) equals a projection matrix in $\mathbb{R}^{\ell m}$, and that the ranges of the projection matrices are orthogonal. The first property follows, because if $\mathbf{P}_1$ and $\mathbf{P}_2$ are projection matrices then so is $\mathbf{P}_1 \otimes \mathbf{P}_2$. The second property follows, because for $\mathbf{P}_1, \mathbf{P}'_1$ (square of the same dimension) and $\mathbf{P}_2, \mathbf{P}'_2$ (square of the same dimension) such that either ranges of $\mathbf{P}_1$ and $\mathbf{P}'_1$ are orthogonal or ranges of $\mathbf{P}_2$ and $\mathbf{P}'_2$ are orthogonal, we obtain that the ranges of $\mathbf{P}_1 \otimes \mathbf{P}_2$ and $\mathbf{P}'_1 \otimes \mathbf{P}'_2$ are orthogonal. To derive the pseudo-inverse, we distinguish two cases.

Case $\ell < m$: We directly invert the eigenvalues in Equation (B.15) to obtain

$$\begin{aligned}
\mathbf{\Gamma}^\dagger &= \frac{m}{\ell} \cdot \frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell} \otimes \frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m} + (m-1) \cdot \left(\mathbf{I}_{\mathcal{J}} - \frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\right) \otimes \left(\mathbf{I}_{\mathcal{Y}} - \frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m}\right) \\
&\quad + \frac{m-1}{1-\ell/m} \cdot \frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell} \otimes \left(\mathbf{I}_{\mathcal{Y}} - \frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m}\right) \\
&= \frac{1}{\ell^2} \cdot \mathbf{1}\mathbf{1}^T + (m-1) \cdot \left(\mathbf{I}_{\mathcal{J}} + \frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{m-\ell}\right) \otimes \left(\mathbf{I}_{\mathcal{Y}} - \frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m}\right) \\
&= \left(\frac{1}{\ell^2} - \frac{m-1}{m(m-\ell)}\right) \cdot \mathbf{1}\mathbf{1}^T + (m-1)\mathbf{I} - \frac{m-1}{m}\sum_j \mathbf{1}_j\mathbf{1}_j^T + \frac{m-1}{m-\ell}\sum_y \mathbf{1}_y\mathbf{1}_y^T.
\end{aligned}$$

Recall that Equation (B.15) involves $m(m-1)\mathbf{\Gamma}$. To obtain $\bar{\rho}$, we again evaluate $\mathbf{1}_{\mathbf{y}'}^T\mathbf{\Gamma}^\dagger\mathbf{1}_{\mathbf{y}}$ for any $\mathbf{y} \in Y(x)$. We write $A_{\mathbf{y}}$ for the set of actions appearing in $\mathbf{y}$:

$$\begin{aligned}
\mathbf{1}_{\mathbf{y}'}^T\mathbf{\Gamma}^\dagger\mathbf{1}_{\mathbf{y}} &= \left(\frac{1}{\ell^2} - \frac{m-1}{m(m-\ell)}\right) \cdot (\mathbf{1}_{\mathbf{y}'}^T\mathbf{1})(\mathbf{1}^T\mathbf{1}_{\mathbf{y}}) + (m-1)\mathbf{1}_{\mathbf{y}'}^T\mathbf{1}_{\mathbf{y}} - \frac{m-1}{m}\sum_j (\mathbf{1}_{\mathbf{y}'}^T\mathbf{1}_j)(\mathbf{1}_j^T\mathbf{1}_{\mathbf{y}}) \\
&\quad + \frac{m-1}{m-\ell}\sum_y (\mathbf{1}_{\mathbf{y}'}^T\mathbf{1}_y)(\mathbf{1}_y^T\mathbf{1}_{\mathbf{y}}) \\
&= \left(\frac{1}{\ell^2} - \frac{m-1}{m(m-\ell)}\right) \cdot \ell^2 + \sum_j \frac{\mathbf{1}\{\mathbf{y}'_j = \mathbf{y}_j\}}{1/(m-1)} - \frac{m-1}{m} \cdot \ell \\
&\quad + \frac{m-1}{m-\ell}\sum_y \mathbf{1}\{y \in A_{\mathbf{y}'}\}\mathbf{1}\{y \in A_{\mathbf{y}}\} \qquad \text{(B.16)} \\
&= 1 - \frac{(m-1)(\ell^2 + m\ell - \ell^2)}{m(m-\ell)} + \sum_j \frac{\mathbf{1}\{\mathbf{y}'_j = \mathbf{y}_j\}}{1/(m-1)} + \frac{m-1}{m-\ell} \cdot |A_{\mathbf{y}'} \cap A_{\mathbf{y}}| \\
&= 1 - \frac{m-1}{m-\ell} \cdot \ell + \sum_j \frac{\mathbf{1}\{\mathbf{y}'_j = \mathbf{y}_j\}}{1/(m-1)} + \frac{m-1}{m-\ell} \cdot |A_{\mathbf{y}} \cap A_{\mathbf{y}'}|,
\end{aligned}$$

where Equation (B.16) follows because $\mathbf{1}^T\mathbf{1}_{\mathbf{y}} = \ell$ and $\mathbf{1}_j^T\mathbf{1}_{\mathbf{y}} = 1$ for any valid slate $\mathbf{y}$. By setting $\mathbf{y}' = \mathbf{y}$, we obtain $\bar{\rho} = 1 + \ell(m-1) = m\ell - \ell + 1$.

Case $\ell = m$: Again, we directly invert the eigenvalues in Equation (B.15):

$$\begin{aligned}
\boldsymbol{\Gamma}^\dagger &= \frac{1}{\ell^2}\cdot \mathbf{1}\mathbf{1}^T + (m-1)\cdot\left(\mathbf{I}_{\mathcal{J}} - \frac{\mathbf{1}_{\mathcal{J}}\mathbf{1}_{\mathcal{J}}^T}{\ell}\right)\otimes\left(\mathbf{I}_{\mathcal{Y}} - \frac{\mathbf{1}_{\mathcal{Y}}\mathbf{1}_{\mathcal{Y}}^T}{m}\right)\\
&= \frac{1}{m}\cdot\mathbf{1}\mathbf{1}^T + (m-1)\mathbf{I} - \frac{m-1}{m}\sum_j \mathbf{1}_j\mathbf{1}_j^T - \frac{m-1}{m}\sum_y \mathbf{1}_y\mathbf{1}_y^T.
\end{aligned}$$

We finish the theorem by evaluating $\mathbf{1}_{\mathbf{y}'}^T\boldsymbol{\Gamma}^\dagger\mathbf{1}_{\mathbf{y}}$:

$$\begin{aligned}
\mathbf{1}_{\mathbf{y}'}^T\boldsymbol{\Gamma}^\dagger\mathbf{1}_{\mathbf{y}} &= \frac{1}{m}\cdot(\mathbf{1}_{\mathbf{y}'}^T\mathbf{1})(\mathbf{1}^T\mathbf{1}_{\mathbf{y}}) + (m-1)\mathbf{1}_{\mathbf{y}'}^T\mathbf{1}_{\mathbf{y}} - \frac{m-1}{m}\sum_j(\mathbf{1}_{\mathbf{y}'}^T\mathbf{1}_j)(\mathbf{1}_j^T\mathbf{1}_{\mathbf{y}})\\
&\quad - \frac{m-1}{m}\sum_y(\mathbf{1}_{\mathbf{y}'}^T\mathbf{1}_a)(\mathbf{1}_a^T\mathbf{1}_{\mathbf{y}})\\
&= \frac{1}{m}\cdot m^2 + \sum_j\frac{\mathbf{1}\{\mathbf{y}'_j = \mathbf{y}_j\}}{1/(m-1)} - \frac{m-1}{m}\cdot m - \frac{m-1}{m}\cdot m\\
&= \sum_j\frac{\mathbf{1}\{\mathbf{y}'_j = \mathbf{y}_j\}}{1/(m-1)} - m + 2.
\end{aligned}$$

We obtain $\bar{\rho} = m^2 - 2m + 2$ by setting $\mathbf{y}' = \mathbf{y}$ and Equation (B.13) by taking an expectation over $\mathbf{y}' \sim \pi(\cdot \mid x)$. □

B.4.2 Translation Theorem and Proofs for Kappa-Uniform Distributions

In this section we derive bounds on $\bar{\rho}_{\pi_0,x}$ when π_0 is not uniform, but only κ-uniform. The main result is a translation theorem relating $\bar{\rho}_{\mu,x}$ to $\bar{\rho}_{\nu,x}$ for arbitrary μ and ν. This lets us translate the bounds for uniform distributions in Appendix B.4.1 into bounds for κ-uniform distributions.

Theorem 10. *Let μ be a κ-uniform policy and let ν denote the uniform stochastic policy. Then*

$$\kappa\bar{\rho}_{\mu,x} \leq \bar{\rho}_{\nu,x}.$$

Proof of Proposition 2. The proposition follows by Proposition 4 with the $\bar{\rho}_{\pi_0,x}$ bounded by Theorem 10, using the definition of κ-uniform distributions and the values of $\bar{\rho}_{\nu,x}$ obtained in Propositions 5 and 6. □

B.5 The P-values for Plots in Figure 5.1 and Figure 5.2

number of samples (n)	PI vs IPS		PI vs DM	
	TTS	UTILITYRATE	TTS	UTILITYRATE
200	2.5×10^{-1}	4.7×10^{-3}	—	—
600	3.8×10^{-2}	1.6×10^{-3}	—	—
2 000	1.3×10^{-5}	2.0×10^{-2}	—	—
6 000	3.7×10^{-5}	2.0×10^{-2}	—	1.8×10^{-2}
20 000	1.2×10^{-5}	1.9×10^{-2}	1.5×10^{-3}	4.3×10^{-7}
60 000	4.5×10^{-6}	$< 10^{-8}$	8.1×10^{-4}	$< 10^{-8}$

Table B.1: The p-values of a t-test between PI and IPS, and PI and DM on search engine data (Figure 5.1). Results where DM performs better than PI are omitted.

number of samples (n)	$\ell = 5, m = 20, \alpha = 0$		$\ell = 10, m = 20, \alpha = 0$		$\ell = 5, m = 20, \alpha = 10$	
	PI vs wIPS	PI vs DM	PI vs wIPS	PI vs DM	PI vs wIPS	PI vs DM
200	$< 10^{-8}$	—	$< 10^{-8}$	—	$< 10^{-8}$	—
600	$< 10^{-8}$	—	$< 10^{-8}$	—	1.0×10^{-8}	—
2 000	$< 10^{-8}$	—	$< 10^{-8}$	—	$< 10^{-8}$	—
6 000	$< 10^{-8}$	—	$< 10^{-8}$	—	$< 10^{-8}$	—
20 000	$< 10^{-8}$	7.3×10^{-2}	$< 10^{-8}$	—	$< 10^{-8}$	—
60 000	$< 10^{-8}$	5.6×10^{-3}	$< 10^{-8}$	—	$< 10^{-8}$	—
200 000	$< 10^{-8}$	6.0×10^{-5}	$< 10^{-8}$	6.1×10^{-2}	$< 10^{-8}$	4.4×10^{-4}
600 000	$< 10^{-8}$	$< 10^{-8}$	$< 10^{-8}$	7.3×10^{-4}	$< 10^{-8}$	7.5×10^{-5}

Table B.2: The p-values of a t-test between PI and IPS, and PI and DM on semi-synthetic data (Figure 5.2). Results where DM performs beats PI are omitted.

B.6 Off-Policy Evaluation on Semi-Synthetic Data

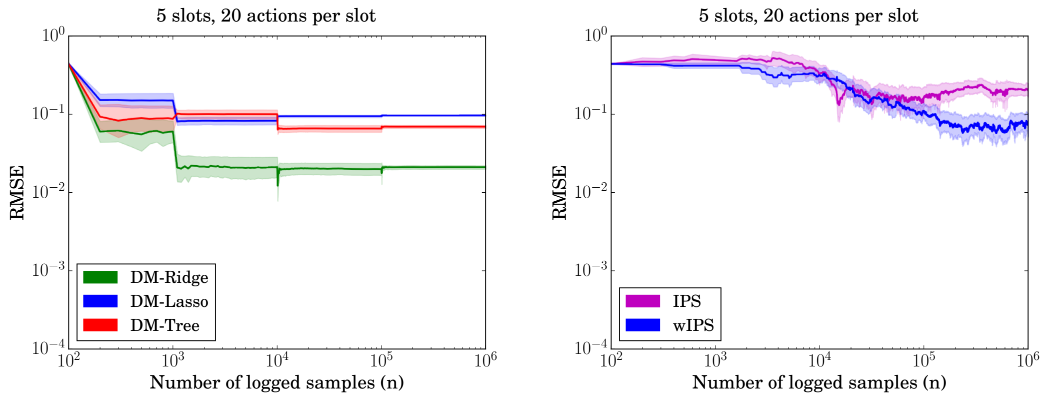


Figure B.1: RMSE of value estimators with increasing amounts of logged data from a uniform logging policy. Policies rank $\ell = 5$ out of $m = 20$ candidates. Left: DM methods, Right: IPS estimators.

In Figure B.1, we compare the performance of several variants of estimators in each family of baseline approaches (DM and IPS) plotted in Figure 5.2. For the DM family of approaches, variants differ in their choice of regression predictor $\hat{\delta}(x, \mathbf{y})$ that maps $\mathbf{f}(x, \mathbf{y})$ to $V(x, \mathbf{y})$. $\mathbf{f}(x, \mathbf{y})$ is defined as the concatenation of displayed document features (in order) $\mathbf{f}(x, y)$ for all these variants. Regression hyper-parameters are selected via five-fold cross-validation with each fold containing disjoint queries.

1. DM-tree: $\hat{\delta}(x, \mathbf{y})$ is predicted by a regression tree; maximum tree depth is the only hyper-parameter that is tuned.
2. DM-ridge: $\hat{\delta}(x, \mathbf{y})$ is implemented using ridge regression; ℓ_2-regularization is cross-validated.
3. DM-lasso: $\hat{\delta}(x, \mathbf{y})$ is implemented using lasso regression; ℓ_1-regularization is cross-validated.

For the IPS family, we compare standard inverse propensity scoring (IPS) against the weighted variant (wIPS). As theory predicts, RMSE of IPS is worse than that of wIPS since wIPS achieves a more favorable bias-variance trade-off.

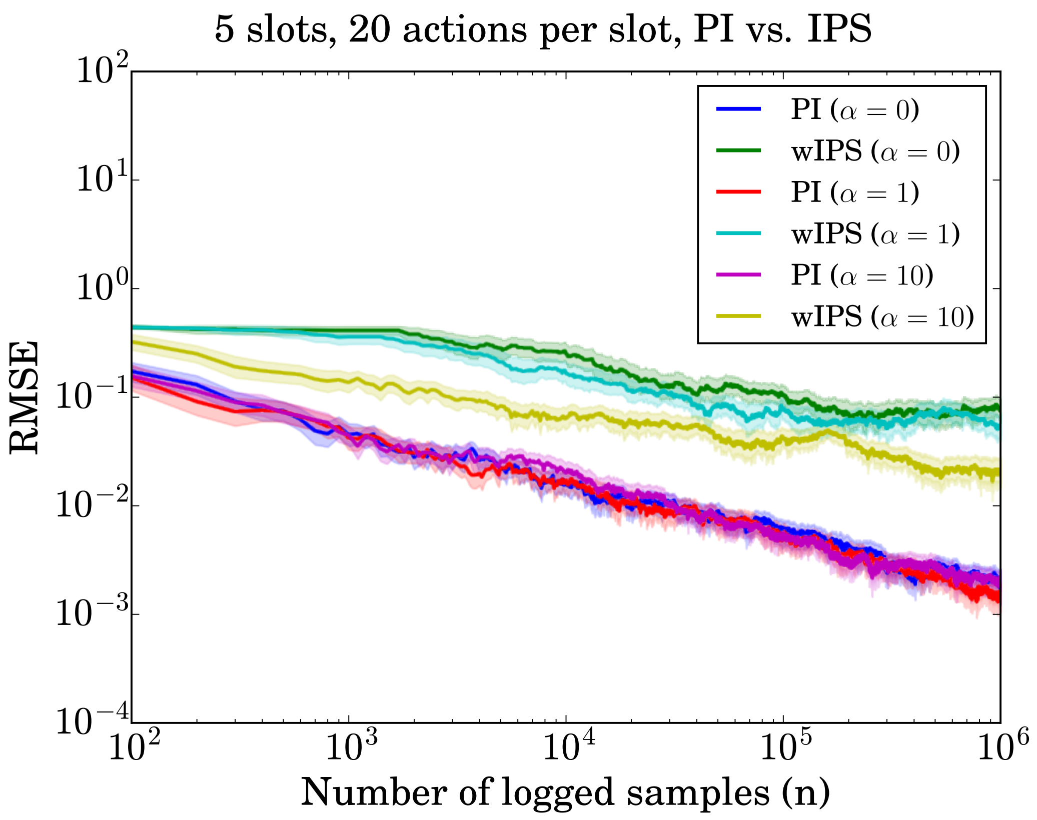


Figure B.2: RMSE curves for pseudoinverse estimator and wIPS. Policies rank $\ell = 10$ out of $m = 20$ candidates. Overlap between logging and target policy is controlled via $\alpha \in \{0, 1, 10\}$.

Figure B.2 shows a relative comparison of the pseudoinverse estimators and IPS estimators as the discrepancy between the logging policy and the target policy is varied. The logging policy is as described in Section 5.6.1, parametrized by $\alpha \geq 0$. $\alpha = 0$ yields a uniform random logging policy and $\alpha \to \infty$ corresponds to a deterministic policy. As α is varied in $\{0, 1, 10\}$, PI remains stable while wIPS improves — this improvement is because the target policy and $pred_{\text{title}}$ (the deterministic extreme of π_0) overlap and the inverse propensity scores are better scaled and induce lower variance.

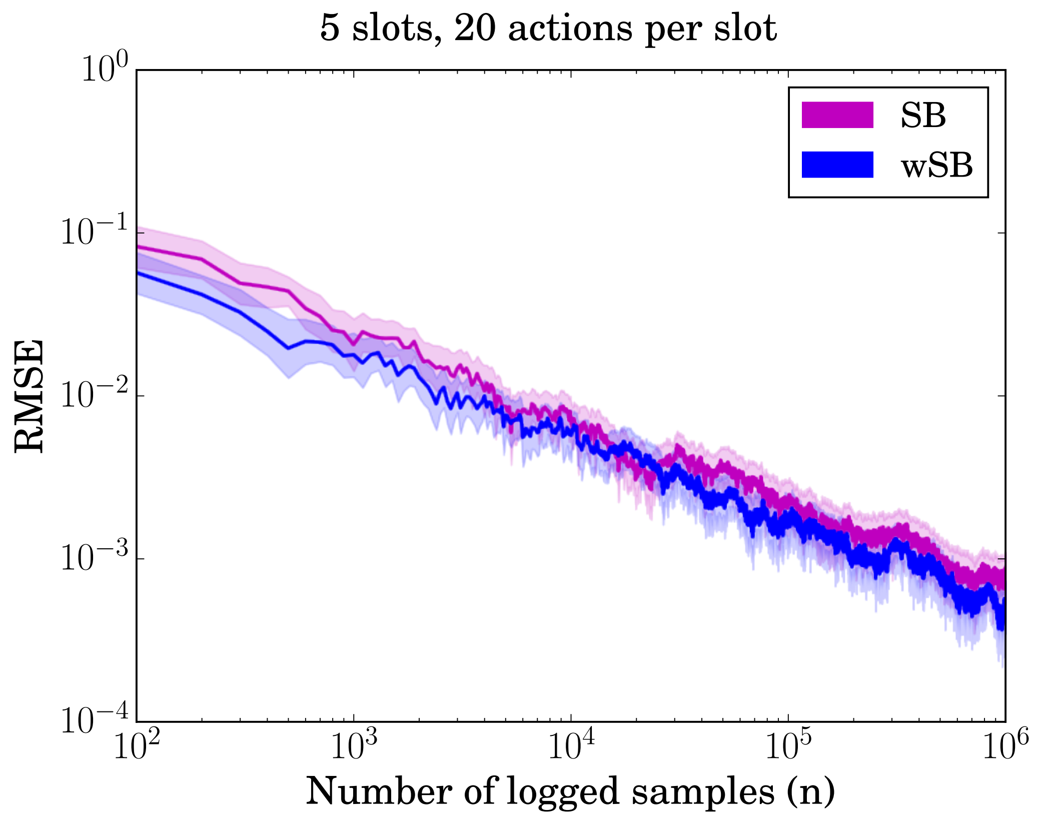


Figure B.3: RMSE curves for semi-bandit approaches using the IPS estimator per slot (SB) and the wIPS estimator per slot (wSB). The logging policy selects slates uniformly at random. Policies rank $\ell = 5$ out of $m = 20$ candidates.

Finally, we also compare to the hypothetical semi-bandit approach, which uses more information than assumed by PI, IPS and DM. Semi-bandits assume that intrinsic values $\phi_{x_i}(j, y_{ij})$ are observed for $j \le \ell$. Given these values, as defined in Example 3, i.e., $\phi_{x_i}(j, y_{ij}) = \left(2^{rel(x_i,y_{ij})} - 1\right) / \log_2(j+1)\mathrm{DCG}^\star(x_i)$, the estimator $\hat{V}_{\mathrm{wSB}}$ sums wIPS estimates across slots:

$$\hat{V}_{\mathrm{wSB}}(\pi) := \sum_{j=1}^{\ell}\Big[\sum_{i=1}^{n}\phi_{x_i}(j, y_{ij}) \cdot \frac{\pi(y_{ij} \mid x_i)}{\pi_0(y_{ij} \mid x_i)}\Big/\Big(\sum_{i=1}^{n}\frac{\pi(y_{ij} \mid x_i)}{\pi_0(y_{ij} \mid x_i)}\Big)\Big].$$

It is only asymptotically unbiased, but it outperforms the unbiased variant based on standard IPS for each slot, as seen in Figure B.3.

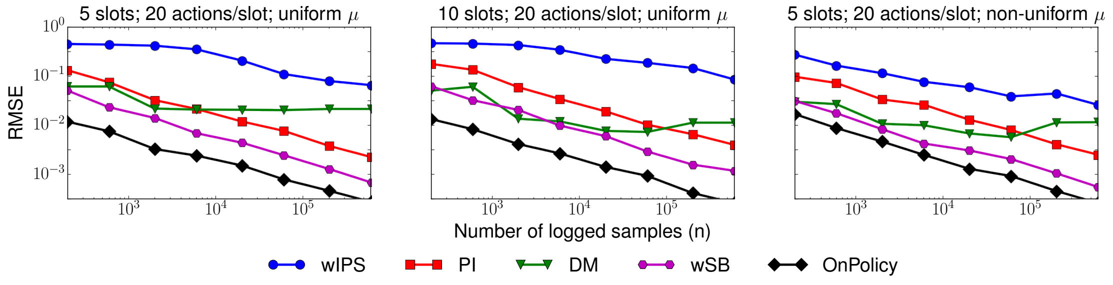


Figure B.4: RMSE curves under uniform logging ($\alpha = 0$) and non-uniform logging ($\alpha = 10$) policies. This recreates Figure 5.2, but also including the hypothetical semi-bandit (SB) approaches.

As Figure B.4 shows, the wSB approach requires somewhere between 4x and 10x less data than PI. So, in those cases when additional per-action feedback which relates to the page-level reward according to the semi-bandit model is available, this method is clearly preferred over PI. When the available feedback does not obviously satisfy the semi-bandit model, however, this approach will exhibit bias like the direct method. For instance, no obvious feedback of this nature was available in the search engine data from Section 5.6.3 and hence we could not evaluate the semi-bandits baseline in that setting.

B.7 Comparison of Pointwise Learning-to-Rank Approaches for Off-Policy Optimization

Companion tables for Section 5.6.2 are provided in Table B.3, Table B.4 and Table B.5. Supervised pointwise learning-to-rank (L2R) algorithms typically regress to some monotone function of annotated relevance judgements $rel(x, y)$. Direct regression to $rel(x, y)$ gives the SUP-Rel approach, while regressing to $2^{rel(x,y)} - 1$ (which is well motivated by the fact that these methods are eventually trying to optimize NDCG) gives the SUP-Gain approach. Our PI-OPT approach is outlined in Section 5.6.2.

We studied the behavior of PI-OPT for three different model classes: decision tree regression, lasso and ridge regression. Since the MQ2008 dataset was already divided into 5 folds, for each fold we used the validation fold to tune hyper-parameters. After re-training on the train and validate folds, we report the test fold NDCG. This procedure is repeated for 10 independent runs of $n = 10^5$ samples collected from the uniform random logging policy. Recall that SUP-Rel and SUP-Gain use approximately 12K annotated pairs. The average of the test set NDCG per fold, and the macro-average over folds is reported.

We find that PI-OPT is able to compete and even outperform the best among SUP-Rel and SUP-Gain. We find that the number of samples needed to achieve parity is quite modest. Moreover, the variability across runs is negligible at $n = 10^5$ (standard error in NDCG across 10 runs for each fold < 0.002).

Finally, in Figure B.5, we depict the performance of PI-OPT as a function of increasing amount of logged samples.

Fold	Logger	SUP-Rel	SUP-Gain	PI-OPT
1	0.273	0.455	0.461	0.473
2	0.285	0.426	0.427	0.421
3	0.289	0.415	0.420	0.426
4	0.273	0.470	0.469	0.469
5	0.259	0.480	0.489	0.492
Avg	0.276	0.449	0.453	0.456

Table B.3: Results of off-policy optimization using a decision tree regression model class.

Fold	Logger	SUP-Rel	SUP-Gain	PI-OPT
1	0.273	0.466	0.459	0.467
2	0.285	0.427	0.427	0.413
3	0.289	0.425	0.423	0.413
4	0.273	0.468	0.462	0.484
5	0.259	0.492	0.486	0.517
Avg	0.276	0.456	0.451	0.459

Table B.4: Results of off-policy optimization when using lasso regression.

Fold	Logger	SUP-Rel	SUP-Gain	PI-OPT
1	0.273	0.456	0.455	0.451
2	0.285	0.418	0.416	0.418
3	0.289	0.418	0.417	0.413
4	0.273	0.460	0.457	0.454
5	0.259	0.487	0.486	0.476
Avg	0.276	0.448	0.446	0.442

Table B.5: Results of off-policy optimization when using ridge regression.

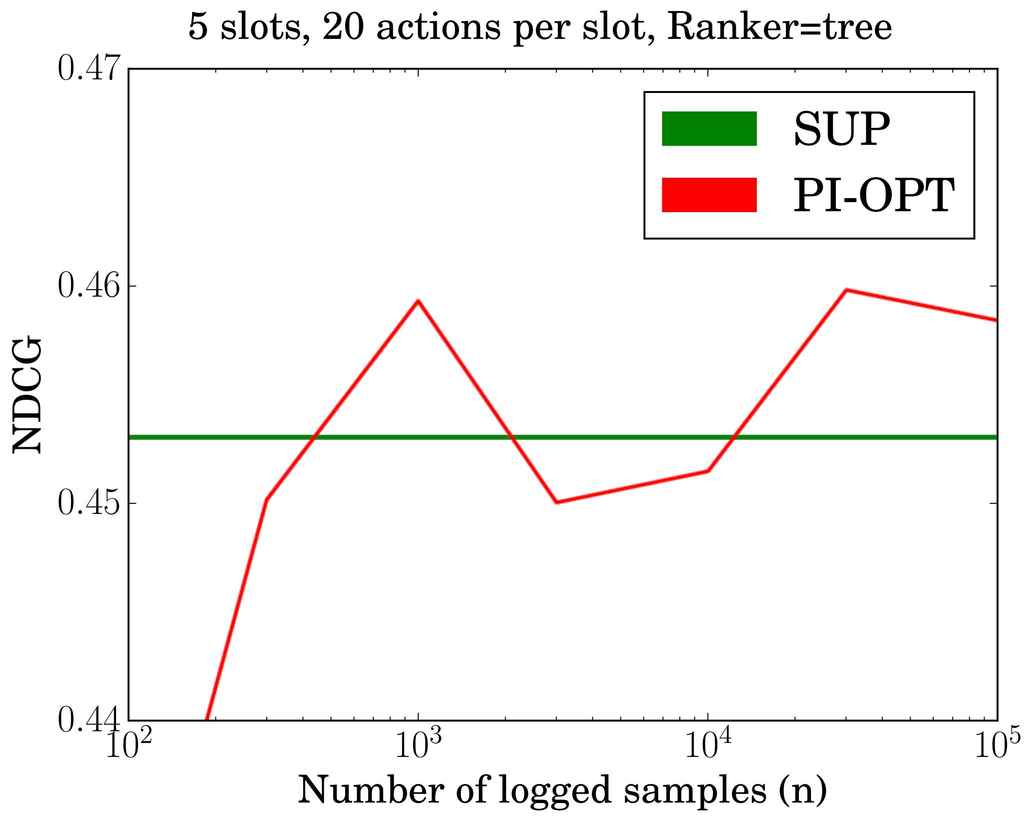


Figure B.5: One trial of off-policy optimization with the decision tree model class. Test set NDCG is plotted as a function of dataset size sampled from a uniform random logging policy.

APPENDIX C

SUPPLEMENTARY MATERIAL FOR CHAPTER 6

C.1 Proof of Theorem 8

Theorem. *With probability at least* $1-\eta$ *in the random vector* $(x_1, y_1)\dots(x_n, y_n) \overset{i.i.d.}{\sim} \pi_0$*, with observed losses* $\delta_1, \dots \delta_n$*, for* $n \geq 16$ *and a stochastic hypothesis space* $\mathcal{H}$ *with capacity* $\mathcal{N}_\infty(\frac{1}{n}, \mathcal{F}_\mathcal{H}, 2n)$*,*

$$\forall \pi \in \mathcal{H} : R(\pi) \leq \hat{R}^M(\pi) + \sqrt{18\frac{\hat{\boldsymbol{Var}}(z_\pi)Q_\mathcal{H}(n,\eta)}{n}} + M\frac{15Q_\mathcal{H}(n,\eta)}{n-1},$$

$$\textit{where, } Q_\mathcal{H}(n,\eta) \equiv \log(\frac{10 \cdot \mathcal{N}_\infty(\frac{1}{n}, \mathcal{F}_\mathcal{H}, 2n)}{\eta}), \quad 0 < \eta < 1.$$

Proof. The proof follows from a direct application of empirical Bernstein bounds derived by Maurer and Pontil (see Theorem 6 [82]) applied to the deterministic function class $\mathcal{F}_\mathcal{H}$. We sketch the main argument using symmetrization and Rademacher variables here.

Define the random variable $s_\pi = 1 + \frac{z_\pi}{M}$ with mean $\mathbb{E}_{\pi_0}[s_\pi]$ and variance $\boldsymbol{Var}(s_\pi)$. Observe that $\mathbb{E}_{\pi_0}[s_\pi] = 1 + \frac{R^M(\pi)}{M}$ from Lemma 7. Let ${s_\pi}^i = 1 + \frac{{z_\pi}^i}{M}$. The sample $\mathcal{D}$ essentially contains n i.i.d. observations of s_π. Let $\hat{\bar{s}}_\pi$ and $\hat{\boldsymbol{Var}}(s_\pi)$ denote the empirical mean and variance of $\{{s_\pi}^i\}_{i=1}^n$ respectively. Observe that $\hat{\boldsymbol{Var}}(s_\pi) = \frac{\hat{\boldsymbol{Var}}(z_\pi)}{M^2}$. Abusing notation slightly, we will use boldface $\boldsymbol{s_\pi}$ to refer to the sample $\{{s_\pi}^i\}_{i=1}^n$.

We begin with Bennet's inequality. For $s, \{s^i\}_{i=1}^n$ i.i.d. bounded random variables in $[0,1]$ having mean $\mathbb{E}[s]$ and variance $\boldsymbol{Var}(s)$, with probability at least

$1 - \eta$ in $\{s^i\}_{i=1}^n \equiv \boldsymbol{s}$,

$$\mathbb{E}[s] - \hat{\hat{s}} \leq \sqrt{\frac{2\boldsymbol{Var}(s) \log 1/\eta}{n}} + \frac{\log 1/\eta}{3n}. \tag{C.1}$$

Intuitively, Bennet's inequality tells us that the estimate $\hat{\hat{s}}$ has lower accuracy if $\boldsymbol{Var}(s)$ is high, which exactly captures our intuition about the variance introduced by importance sampling when estimating the risk of a hypothesis "far" from π_0. However, the diameter of this confidence interval depends on the unobservable $\boldsymbol{Var}(s)$.

We now recite the empirical Bernstein bound (Theorem 11 [82]) that gives a variance-sensitive bound with an observable confidence interval.

Under the same conditions as Bennet's inequality Equation (C.1), let $n \geq 2, \hat{\boldsymbol{Var}}(s)$ represent the empirical variance of $\{s^i\}_{i=1}^n$. With probability at least $1 - \eta$,

$$\mathbb{E}[s] - \hat{\hat{s}} \leq \sqrt{\frac{2\hat{\boldsymbol{Var}}(s) \log 2/\eta}{n}} + \frac{7 \log 2/\eta}{3(n-1)}. \tag{C.2}$$

This follows from confidence bounds on the sample standard deviation $\sqrt{\hat{\boldsymbol{Var}}(s)}$ compared to the true standard deviation $\mathbb{E}_{\boldsymbol{s}}\left[\hat{\boldsymbol{Var}}(s)\right]$. Based on this bound, Maurer and Pontil [82] define two Lipschitz continuous functions, $\Phi, \Psi : [0,1]^n \times \mathbb{R}_+ \to \mathbb{R}$.

$$\Phi(\boldsymbol{s}, t) = \hat{\hat{s}} + \sqrt{\frac{2\hat{\boldsymbol{Var}}(s)t}{n}} + \frac{7t}{3(n-1)}$$

$$\Psi(\boldsymbol{s}, t) = \hat{\hat{s}} + \sqrt{\frac{18\hat{\boldsymbol{Var}}(s)t}{n}} + \frac{11t}{n-1}.$$

These functions are Lipschitz continuous,

$$\Phi(\boldsymbol{s}, t) - \Phi(\boldsymbol{s'}, t) \leq (1 + 2\sqrt{\frac{t}{n}})\|\boldsymbol{s} - \boldsymbol{s'}\|_\infty$$
$$\Psi(\boldsymbol{s}, t) - \Psi(\boldsymbol{s'}, t) \leq (1 + 6\sqrt{\frac{t}{n}})\|\boldsymbol{s} - \boldsymbol{s'}\|_\infty. \tag{C.3}$$

The inequalities follow directly from $\sqrt{\hat{\boldsymbol{Var}}(s)} - \sqrt{\hat{\boldsymbol{Var}}(s')} \leq \sqrt{2}\|\boldsymbol{s} - \boldsymbol{s'}\|_\infty$.

For the symmetrization argument, consider two sets of n samples $\mathcal{D}$ and $\mathcal{D}'$ drawn from π_0 according to the conditions of Theorem 8 and used to estimate risk of a hypothesis π. This gives rise to two sets of n i.i.d. random variables $\boldsymbol{s}_\pi$ and $\boldsymbol{s}'_\pi$. Also define the Rademacher variables $\sigma_1, \ldots \sigma_n \overset{i.i.d}{\sim} \mathcal{U}\{-1, 1\}$. Define $(\boldsymbol{\sigma}, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi)$ as the vector with i^{th} co-ordinate set to ${s_\pi}^i$ or ${s'_\pi}^i$ as specified by σ_i.

$$(\boldsymbol{\sigma}, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi)_i = \begin{cases} {s_\pi}^i & \text{if } \sigma_i = 1 \\ {s'_\pi}^i & \text{if } \sigma_i = -1. \end{cases}$$

For a fixed $\pi \in \mathcal{H}$ and a fixed double sample $\boldsymbol{s}_\pi, \boldsymbol{s}'_\pi$ as described above,

$$\Pr_\sigma [\Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) \geq \Psi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t)] \leq 5e^{-t}. \tag{C.4}$$

This follows (see Lemma 14 [82]) by decomposing the event $[\Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) \geq \Psi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t)]$ as $[\Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) \geq A] \wedge [A \geq \Psi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t)]$ where A uses the true mean and variance of s_π. The probability of the first event can be bounded using Bennet's inequality from Equation (C.1), while the second event can be bounded using the empirical Bernstein bound from Equation (C.2) and the confidence bounds on the sample standard deviation $\sqrt{\hat{\boldsymbol{Var}}(s)}$.

Set $t = \log \frac{2}{\eta}$ and consider $t \geq \log 4$ (i.e. $\eta \leq \frac{1}{2}$). Equation (C.2) implies, for any $\pi \in \mathcal{H}$,

$$\Pr(\Phi(\boldsymbol{s}_\pi, t) \geq \mathbb{E}[s_\pi]) \geq \frac{1}{2}. \tag{C.5}$$

Hence, for any $\rho > 0$,

$$\begin{aligned}
&\Pr_{\mathcal{D}}(\exists \pi \in \mathcal{H} : \mathbb{E}[s_\pi] > \Psi(\boldsymbol{s}_\pi, t) + \rho) = \mathbb{E}_{\mathcal{D}}\left[\sup_{\pi \in \mathcal{H}} \mathbf{1}\{\mathbb{E}[s_\pi] > \Psi(\boldsymbol{s}_\pi, t) + \rho\}\right] \\
&\leq \mathbb{E}_{\mathcal{D}}\left[\sup_{\pi \in \mathcal{H}} \mathbf{1}\{\mathbb{E}[s_\pi] > \Psi(\boldsymbol{s}_\pi, t) + \rho\}\right] 2\Pr(\Phi(\boldsymbol{s}'_\pi, t) \geq \mathbb{E}[s'_\pi]) && \text{Eq. (C.5)} \\
&= 2\mathbb{E}_{\mathcal{D}}\left[\sup_{\pi \in \mathcal{H}} \mathbb{E}_{\mathcal{D}'}\left[\mathbf{1}\{\mathbb{E}[s_\pi] > \Psi(\boldsymbol{s}_\pi, t) + \rho \wedge \Phi(\boldsymbol{s}'_\pi, t) \geq \mathbb{E}[s_\pi]\}\right]\right] && \because \mathbb{E}[s_\pi] = \mathbb{E}[s'_\pi] \\
&\leq 2\mathbb{E}_{\mathcal{D}}\mathbb{E}_{\mathcal{D}'}\left[\sup_{\pi \in \mathcal{H}} \mathbf{1}\{\mathbb{E}[s_\pi] > \Psi(\boldsymbol{s}_\pi, t) + \rho \wedge \Phi(\boldsymbol{s}'_\pi, t) \geq \mathbb{E}[s_\pi]\}\right] \\
&\leq 2\mathbb{E}_{\mathcal{D}}\mathbb{E}_{|mathcalD'}\left[\sup_{\pi \in \mathcal{H}} \mathbf{1}\{\Phi(\boldsymbol{s}'_\pi, t) > \Psi(\boldsymbol{s}_\pi, t) + \rho\}\right] \\
&= 2\mathbb{E}_{\sigma}\mathbb{E}_{\mathcal{D}}\mathbb{E}_{\mathcal{D}'}\left[\sup_{\pi \in \mathcal{H}} \mathbf{1}\{\Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) > \Psi((-\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) + \rho\}\right] && \because \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi \text{ are iid} \\
&\leq 2\sup_{\mathcal{D},\mathcal{D}'} \mathbb{E}_{\sigma}\left[\sup_{\pi \in \mathcal{H}} \mathbf{1}\{\Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) > \Psi((-\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) + \rho\}\right] \\
&= 2\sup_{\mathcal{D},\mathcal{D}'} \Pr_{\sigma}(\exists \pi \in \mathcal{H} : \Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) > \Psi((-\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) + \rho).
\end{aligned}$$

For a fixed $\mathcal{D}, \mathcal{D}'$, consider the ϵ−cover of $\mathcal{F}_{\mathcal{H}}$, ${\mathcal{F}_{\mathcal{H}}}^0$. Denote the set of stochastic policies that correspond to each $f_\pi \in {\mathcal{F}_{\mathcal{H}}}^0$ by $\mathcal{H}^0$. We know that $\left|\mathcal{H}^0\right| \leq \mathcal{N}_\infty(\epsilon, \mathcal{F}_{\mathcal{H}}, 2n)$ (by definition of the covering number, and since there is a one-to-one mapping from π to f_π) and $\forall \pi \in \mathcal{H}, \exists \pi' \in \mathcal{H}^0$ such that $\|\boldsymbol{s}_\pi - \boldsymbol{s}_{\pi'}\|_\infty \leq \epsilon$ and $\|\boldsymbol{s}'_\pi - \boldsymbol{s}'_{\pi'}\|_\infty \leq \epsilon$ (by definition of ϵ−cover). Instantiate $\rho = \epsilon(2 + 8\sqrt{\frac{t}{n}})$ and suppose $\exists \pi \in \mathcal{H}$ such that $\Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) > \Psi((-\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) + \rho$. Since Φ and Ψ are Lipschitz continuous, as demonstrated in Equation (C.3), hence there must exist a $\pi' \in \mathcal{H}^0$ such that $\Phi((\sigma, \boldsymbol{s}_{\pi'}, \boldsymbol{s}'_{\pi'}), t) > \Psi((-\sigma, \boldsymbol{s}_{\pi'}, \boldsymbol{s}'_{\pi'}), t)$. Hence,

$$\begin{aligned}
&\Pr_{\sigma}(\exists \pi \in \mathcal{H} : \Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) > \Psi((-\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) + \epsilon(2 + 8\sqrt{\frac{t}{n}})) \\
&\leq \Pr_{\sigma}(\exists \pi \in \mathcal{H}^0 : \Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) > \Psi((-\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t)) \\
&\leq \sum_{\pi \in \mathcal{H}^0} \Pr_{\sigma}(\Phi((\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t) > \Psi((-\sigma, \boldsymbol{s}_\pi, \boldsymbol{s}'_\pi), t)) \\
&\leq 5e^{-t}\mathcal{N}_\infty(\epsilon, \mathcal{F}_{\mathcal{H}}, 2n) && \text{Equation (C.4)}\,.
\end{aligned}$$

In short,

$$\Pr_{\mathcal{D}}(\exists\pi \in \mathcal{H} : \mathbb{E}[s_\pi] > \Psi(\boldsymbol{s}_\pi, t) + \epsilon(2 + 8\sqrt{\frac{t}{n}})) \leq 10e^{-t}\mathcal{N}_\infty(\epsilon, \mathcal{F}_{\mathcal{H}}, 2n).$$

Setting $10e^{-t}\mathcal{N}_\infty(\epsilon, \mathcal{F}_{\mathcal{H}}, 2n) = \eta$ we get $t_\eta = \log\frac{10\mathcal{N}_\infty(\epsilon,\mathcal{F}_{\mathcal{H}},2n)}{\eta} > 1$. Moreover, $\frac{2(t_\eta+1)}{n} \leq \frac{2(t_\eta+1)}{n-1} \leq \frac{4t_\eta}{n-1}$ and for $n \geq 16$, $8\sqrt{\frac{t_\eta}{n}} \leq 2t_\eta$. Substituting $\epsilon = \frac{1}{n}$ and simplifying,

$$\Pr_{\mathcal{D}}(\exists\pi \in \mathcal{H} : \mathbb{E}[s_\pi] > \hat{\bar{s}}_\pi + \sqrt{\frac{18\hat{\boldsymbol{Var}}(s_\pi)t_\eta}{n}} + \frac{15t_\eta}{n-1}) \leq \eta.$$

Finally, $\mathbb{E}[s_\pi] = 1 + \frac{R^M(\pi)}{M}$, $\hat{\bar{s}}_\pi = 1 + \frac{\hat{R}^M\pi}{M}$ and $\hat{\boldsymbol{Var}}(s_\pi) = \frac{\hat{\boldsymbol{Var}}(z_\pi)}{M^2}$. Since $\delta(\cdot,\cdot) \leq 0$, hence $R(\pi) \leq R^M(\pi)$. Putting it all together,

$$\Pr_{\mathcal{D}}(\exists\pi \in \mathcal{H} : R(\pi) > \hat{R}^M(\pi) + \sqrt{\frac{18\hat{\boldsymbol{Var}}(z_\pi)t_\eta}{n}} + \frac{15Mt_\eta}{n-1}) \leq \eta.$$

□

C.2 Proof of Proposition 3

Proposition. *For any* $\mathbf{w}_0$ *such that* $\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0}) > 0$,

$$\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}})} \leq A_{\mathbf{w}_0}\sum_{i=1}^{n} {z_{\mathbf{w}}}^i + B_{\mathbf{w}_0}\sum_{i=1}^{n}\{{z_{\mathbf{w}}}^i\}^2 + C_{\mathbf{w}_0}$$

$$A_{\mathbf{w}_0} \equiv \frac{-\hat{\bar{z}}_{\mathbf{w}_0}}{(n-1)\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}},$$

$$B_{\mathbf{w}_0} \equiv \frac{1}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}},$$

$$C_{\mathbf{w}_0} \equiv \frac{n\{\hat{\bar{z}}_{\mathbf{w}_0}\}^2}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}} + \frac{\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}{2}.$$

Proof. Consider a first order Taylor approximation of $\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}})}$ around $\mathbf{w}_0$. Observe that $\sqrt{\cdot}$ is concave.

$$\begin{aligned}
\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}})} &\leq \sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})} + \nabla_\zeta \left(\sqrt{\zeta}\right)|_{\zeta=\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})} (\hat{\boldsymbol{Var}}(z_{\mathbf{w}}) - \hat{\boldsymbol{Var}}(z_{\mathbf{w}_0}))\\
&= \sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})} + \frac{\hat{\boldsymbol{Var}}(z_{\mathbf{w}}) - \hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}{2\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}\\
&= \frac{\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}{2} + \frac{1}{2\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}\hat{\boldsymbol{Var}}(z_{\mathbf{w}})\\
&= \frac{\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}{2} + \frac{\sum_{i=1}^{n}\{z_{\mathbf{w}}{}^{i}\}^2}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}} + \frac{-n\{\hat{\bar{z}}_{\mathbf{w}}\}^2}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(z_{\mathbf{w}_0})}}.
\end{aligned}$$

Again Taylor approximate $-\{\hat{\bar{z}}_{\mathbf{w}}\}^2$, noting that $-\{\cdot\}^2$ is concave.

$$\begin{aligned}
-\{\hat{\bar{z}}_{\mathbf{w}}\}^2 &\leq -\{\hat{\bar{z}}_{\mathbf{w}_0}\}^2 + \nabla_\zeta \left(-\zeta^2\right)|_{\zeta=\hat{\bar{z}}_{\mathbf{w}_0}} (\hat{\bar{z}}_{\mathbf{w}} - \hat{\bar{z}}_{\mathbf{w}_0})\\
&= -\{\hat{\bar{z}}_{\mathbf{w}_0}\}^2 + 2\{\hat{\bar{z}}_{\mathbf{w}_0}\}^2 - 2\hat{\bar{z}}_{\mathbf{w}_0}\hat{\bar{z}}_{\mathbf{w}}\\
&= \{\hat{\bar{z}}_{\mathbf{w}_0}\}^2 - \frac{2\hat{\bar{z}}_{\mathbf{w}_0}}{n}\sum_{i=1}^{n} z_{\mathbf{w}}{}^{i}.
\end{aligned}$$

Substituting above and re-arranging terms, we derive the proposition. □

APPENDIX D
OLD RESEARCH NOTES

I wrote a silly story in February 2014 to motivate, what we later called, the BLBF problem setting and sketched some ideas for possible solutions. The story is reproduced below. I implemented a Gaussian Process | Bayesian Optimization style solution based on the sketched ideas, and it worked remarkably well on toy problems. However, scalability was hopeless, and by April 2014 I implemented POEM, which went on to dominate this approach. The reason behind POEM's rhyme (i.e., formal proofs for its performance) followed in June 2014. I received an email from Thorsten Joachims on May 27, 2014. Subject:⟨Link [82]⟩. Body: "I remembered this paper, in which you will find Theorem 4 particularly interesting." That one-liner has led to this 150+ page thesis!

D.1 Learning with Bandit Supervision

In an apocalyptic future, *Google*® lies in shambles after numerous lawsuits and counter-lawsuits. In the aftermath of these lawsuits, antitrust activists force *Google* to reveal (appropriately anonymized of course) the logs they had maintained on every user on the planet. The courts ruled, "data recording user behavior may not be used as a competitive advantage, but any annotations and transformations that *Google* did in-house remain its sole property". The skeletal remains of *Google* now guard their models, parameters, and curated data with even greater secrecy.

In this landscape, several salvagers dredge through *Google*'s logs, hoping to unlock the secrets of Life, the Universe, and Everything™ and, of course, the secret to building a good search engine. Our protagonist in this tale is one such salvager: the fledgling company *voogle*, started by graduate students frustrated by endless rounds of conference submissions and rejections.

The future may be apocalyptic, but the research industry is flourishing: conferences abound, with a gazillion manuscript submissions and re-submissions. The peer-reviewing process is, hence, stretched to the limit. Rather than bidding for papers, reviewers log on to GMT (*Google*'s response to $Microsoft$® CMT™), and search for submissions using keywords. GMT assigns them a manuscript, and they can respond with a review and a confidence score. Rumor has it: these confidence scores govern the promotion prospects of GMT's developers.

voogle want to build the next generation of conference management toolkits. The core task: Reviewer inputs a query x, and the system should respond with the best manuscript y for x. The erstwhile graduate students slogged for a week and identified the best manuscripts for a handful of queries. Imagine their frustration when they realized that this annotated data would barely do as a test set! Creating a labeled training set was out of the question. In despair, they turned to the GMT logs, hoping to find something they can use.

Indeed they did! Scattered through the logs were several instances of a user issuing a query x, the manuscript that GMT in its infinite wisdom assigned, y, and the *Calibrated Confidence Score*™ $\delta(x, y)$. These scores were the de-biased and normalized confidence ratings of a population of reviewers done with *Google*'s patented transmogrifiers – clearly their data-scrubbing team overlooked these scores when releasing the logs.

Chief Scientist Sir Models-A-Lot scanned these $(x, y, \delta(x, y))$ triples, remarking, "Wow, these are the machinations of a remarkably intelligent contextualized bandit. Look how it explores and exploits across sessions! If only we knew the exact algorithm *Google* used, and the parameters for the online learning model, we could simply replicate the model and deploy it." He looked across hopefully to his colleague, Dr. Data-Morph, but she shook her head. "*Google* will never give us their models."

They discussed their position and their options going forward. "We have a model, parametrized by w, that takes in a reviewer's query x and spits out a manuscript y. The users of our system will generate queries $x \in \mathcal{X}$, and we can safely assume they are independent identical draws from some unknown but fixed distribution $\Pr(\mathcal{X})$. Our system lacks the technological sophistication of GMT's *Calibrated Confidence Score*. And it is not an online learning system: we build, and we deploy. If something catches fire, we rinse and repeat. Without confidence scores, we rely on a binary feedback: if reviewers are happy with their assignment, they will actually review the darn thing, and if they are not, we will not hear a thing. The learning task is to find parameters w^* that maximizes the expected happiness of our users. The challenge is to learn w^* from the data that we have: triples of $(x, y, \delta(x, y))$."

Dr. Data-Morph offered the following suggestion: "We have several learning algorithms for our model which, when given a training set (x, y) containing the best manuscript for each query, can find w^* that generalizes from these to unseen instances. Can we not transmogrify the triples data to this form? Collect all instances of GMT exploring different alternatives y for the same query x, and create a single instance (x, y^*) which has the best-observed score $\delta(x, y^*)$."

"No no no, No no, No", nodded Sir Models-A-Lot. "First, a technicality: this transformation ensures every query x occurs exactly once in the training data. The actual $\Pr(\mathcal{X})$ may look nothing like a uniform distribution, and it is dangerous to argue about expected loss on training and test data when the expectations are with respect to different distributions."

"Second, we are relying too much on GMT having explored well for each query x. What if, for a whole bunch of queries, the observed y have pitifully low $\delta(x, y)$, and the best option actually lies elsewhere. We would be creating a willfully misleading supervised dataset."

"The triples dataset clearly indicates we are modeling the wrong thing. Let us construct a model that models $\delta(x, y)$ directly: it is just a regression problem. Given a query x and a manuscript y, it shall predict $\delta(x, y)$, and our scavenged data is just what Goldilocks ordered for a supervised algorithm. For generating our system's output for a specific query x when deployed, we shall enumerate every possible y, predict $\delta(x, y)$ and pick the highest."

A tiny voice piped up from across the room, and both researchers jumped back in surprise. It was the intern, No-op Noob. "Speak up, Mr. Noob," encouraged Sir Models-A-Lot.

"Nothing, sir. If Dr. Data-Morph's transformation of the training data seemed arbitrary, so too is the procedure to predict y using the regression model. We may have guarantees for the expected regression loss, but I do not know if that translates to our goal: expected happiness of reviewers."

"Also, it may be wrong to assume that (x, y) pairs seen in the triples dataset are independent identical draws from some $\Pr(\mathcal{X} \times \mathcal{Y})$. Clearly, GMT has been using clever policies to generate Y, and we do not know if they are merely i.i.d.

draws from some $\Pr(Y \mid X)$."

The voice of John Langford boomed through the room, "There are ways to control for that," but it fell on deaf ears.

"How about a more principled, less feasible way to create a supervised training set from the triples dataset?" continued No-op Noob.

"For each query x in the triples dataset, from the various $\delta(x, y)$ measurements sampled, we shall transduce $\delta(x, y)$ for the unsampled y's along with confidence scores for our transduction (not to be confused with the *Calibrated Confidence Score*™ $\delta(x, y)$). With this step, we now have a distribution over y for every query x, $\Pr(Y = y \mid x)$ telling us the probability that y is the best manuscript for x. We materialize a weighted training set and learn Dr. Data-Morph's models the good old way. And, … and … profit?"

BIBLIOGRAPHY

[1] Martin Anthony and Peter L. Bartlett. *Neural network learning: Theoretical foundations.* Cambridge University Press, 2009.

[2] Hossein Arsham, A. Feuerverger, Don L. Mcleish, Joseph Kreimer, and Reuven Y. Rubinstein. Sensitivity analysis and the "what if" problem in simulation analysis. *Mathematical and Computer Modelling*, 1989.

[3] Javed A. Aslam, Virgil Pavlu, and Emine Yilmaz. A statistical method for system evaluation using incomplete judgments. In *SIGIR*, 2006.

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

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

[6] Robert Bell, Yehuda Koren, and Chris Volinsky. Chasing $1,000,000: How we won the netflix progress prize. *Statistical Computing and Graphics*, 2007.

[7] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2009.

[8] Alina Beygelzimer and John Langford. The offset tree for learning with partial labels. In *KDD*, 2009.

[9] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 2009.

[10] Avrim Blum, Adam Kalai, and John Langford. Beating the hold-out: Bounds for K-fold and progressive cross-validation. In *COLT*, 1999.

[11] Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. A neural click model for web search. In *WWW*, 2016.

[12] 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.

[13] Phelim Boyle, Mark Broadie, and Paul Glasserman. Monte carlo methods for security pricing. *Journal of Economic Dynamics and Control*, 1997.

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

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

[16] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 1995.

[17] Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games.* Cambridge University Press, 2006.

[18] Olivier Chapelle and Yi Chang. Yahoo! learning to rank challenge overview. In *ICML*, 2011.

[19] Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. Large-scale validation and analysis of interleaved search evaluation. *Transactions on Information Systems*, 2012.

[20] Olivier Chapelle and Mingrui Wu. Gradient descent optimization of smoothed information retrieval metrics. *Information Retrieval*, 2010.

[21] Olivier Chapelle and Ya Zhang. A dynamic bayesian network click model for web search ranking. In *WWW*, 2009.

[22] Wei Chu, Lihong Li, Lev Reyzin, and Robert E. Schapire. Contextual bandits with linear payoff functions. In *AISTATS*, 2011.

[23] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. *Click models for web search.* Morgan & Claypool, 2015.

[24] Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *NIPS*, 2010.

[25] Corinna Cortes, Mehryar Mohri, Michael Riley, and Afshin Rostamizadeh. Sample selection bias correction theory. In *ALT*, 2008.

[26] Corinna Cortes and Vladimir N. Vapnik. Support–vector networks. *Machine Learning*, 1995.

[27] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. An experimental comparison of click position-bias models. In *WSDM*, 2008.

[28] Varsha Dani, Thomas P. Hayes, and Sham M. Kakade. The price of bandit information for online optimization. In *NIPS*, 2008.

[29] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011.

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

[31] Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. In *ICML*, 2011.

[32] Felix Elwert. A brief review of counterfactual causality., 2013.

[33] Michael Evans and Tim Swartz. Methods for approximating integrals in statistics with special emphasis on bayesian integration problems. *Statistical Science*, 1995.

[34] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. In *NIPS*, 2010.

[35] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White. Evaluating implicit measures to improve web search. *Transactions on Information Systems*, 2005.

[36] Nicolas Galichet, Michèle Sebag, and Olivier Teytaud. Exploration vs exploitation vs safety: Risk-aware multi-armed bandits. In *ACML*, 2013.

[37] Javier Garcia and Fernando Fernandez. Safe exploration of state and action spaces in reinforcement learning. *Journal of Artificial Intelligence Research*, 2012.

[38] Rainer Gemulla, Peter J. Haas, Erik Nijkamp, , and Yannis Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *KDD*, 2011.

[39] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 1992.

[40] Jose M. Hernandez-lobato, Neil Houlsby, and Zoubin Ghahramani. Probabilistic matrix factorization with non-random missing data. In *ICML*, 2014.

[41] Tim C. Hesterberg. *Advances in importance sampling.* Stanford University, 1988.

[42] Tim C. Hesterberg. Weighted average importance sampling and defensive mixture distributions. *Technometrics*, 1995.

[43] Keisuke Hirano, Guido W. Imbens, and Geert Ridder. Efficient estimation of average treatment effects using the estimated propensity score. *Econometrica*, 2003.

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

[45] Katja Hofmann, Anne Schuth, Shimon Whiteson, and Maarten de Rijke. Reusing historical interaction data for faster online learning to rank for IR. In *WSDM*, 2013.

[46] Paul W. Holland. Statistics and causal inference. *Journal of the American Statistical Association*, 1986.

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

[48] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, 2008.

[49] Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. In *NIPS*, 2006.

[50] Guido W. Imbens and Donald B. Rubin. *Causal inference for statistics, social and biomedical sciences.* Cambridge University Press, 2015.

[51] Edward L. Ionides. Truncated importance sampling. *Journal of Computational and Graphical Statistics*, 2008.

[52] Thorsten Joachims. Optimizing search engines using clickthrough data. In *KDD*, 2002.

[53] Thorsten Joachims. Evaluating retrieval performance using clickthrough data. In *Text Mining*. Physica/Springer Verlag, 2003.

[54] Thorsten Joachims. A support vector method for multivariate performance measures. In *ICML*, 2005.

[55] Thorsten Joachims. Training linear SVMs in linear time. In *KDD*, 2006.

[56] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *Transactions on Information Systems*, 2007.

[57] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. Unbiased learning-to-rank with biased feedback. In *WSDM*, 2017.

[58] Herman Kahn and Andy W. Marshall. Methods for reducing sample size in monte-carlo computations. *Operations Research*, 1953.

[59] Satyen Kale, Lev Reyzin, and Robert E. Schapire. Non-stochastic bandit slate problems. In *NIPS*, 2010.

[60] 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.

[61] Augustine Kong. A note on importance sampling using standardized weights. Technical report, University of Chicago, 1992.

[62] Yehuda Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *KDD*, 2008.

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

[64] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[65] John Langford, Alexander Strehl, and Jennifer Wortman. Exploration scavenging. In *ICML*, 2008.

[66] John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *NIPS*, 2008.

[67] Damien Lefortier, Adith Swaminathan, Xiaotao Gu, Thorsten Joachims, and Maarten de Rijke. Large-scale validation of counterfactual learning methods: A test-bed. *arXiv:1612.00367*, 2016.

[68] Adrian S. Lewis and Michael L. Overton. Nonsmooth optimization via quasi-newton methods. *Mathematical Programming*, 2013.

[69] Lihong Li. Offline evaluation and optimization for interactive systems. In *WSDM*, 2015.

[70] Lihong Li, Shunbao Chen, Jim Kleban, and Ankur Gupta. Counterfactual estimation and optimization of click metrics in search engines: A case study. In *WWW*, 2015.

[71] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *WWW*, 2010.

[72] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *WSDM*, 2011.

[73] Lihong Li, Rémi Munos, and Csaba Szepesvári. Toward minimax off-policy value estimation. In *AISTATS*, 2015.

[74] Dawen Liang, Laurent Charlin, James McInerney, and David M. Blei. Modeling user exposure in recommendation. In *WWW*, 2016.

[75] Daryl Lim, Julian McAuley, and Gert Lanckriet. Top-n recommendation with missing implicit feedback. In *RecSys*, 2015.

[76] Chih-Jen Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Computation*, 2007.

[77] Roderick J. A. Little and Donald B. Rubin. *Statistical analysis with missing data.* John Wiley, 2002.

[78] Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 2009.

[79] Benjamin M. Marlin and Richard S. Zemel. Collaborative prediction and ranking with non-random missing data. In *RecSys*, 2009.

[80] Benjamin M. Marlin, Richard S. Zemel, Sam Roweis, and Malcolm Slaney. Collaborative filtering and the missing at random assumption. In *UAI*, 2007.

[81] Jérémie Mary, Philippe Preux, and Olivier Nicol. Improving offline evaluation of contextual bandit algorithms via bootstrapping techniques. In *ICML*, 2014.

[82] Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample-variance penalization. In *COLT*, 2009.

[83] Daniel F. McCaffrey, Greg Ridgeway, and Andrew R. Morral. Propensity score estimation with boosted regression for evaluating causal effects in observational studies. *Psychological Methods*, 2004.

[84] Arnaud De Myttenaere, Bénédicte Le Grand, Boris Golden, and Fabrice Rossi. Reducing offline evaluation bias in recommendation systems. In *Benelearn*, 2014.

[85] Jerzy Neyman. On the application of probability theory to agricultural experiments: Essay on principles. *Statistical Science*, 1923.

[86] Art B. Owen. *Monte Carlo theory, methods and examples.* Draft, 2013.

[87] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011.

[88] Kaare B. Petersen, Michael S. Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 2008.

[89] Bruno Pradel, Nicolas Usunier, and Patrick Gallinari. Ranking with non-random missing ratings: Influence of popularity and positivity on evaluation metrics. In *RecSys*, 2012.

[90] Lijing Qin, Shouyuan Chen, and Xiaoyan Zhu. Contextual combinatorial bandit and its application on diversified online recommendation. In *ICDM*, 2014.

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

[92] Karthik Raman, Thorsten Joachims, Pannagadatta K. Shivaswamy, and Tobias Schnabel. Stable coactive learning via perturbation. In *ICML*, 2013.

[93] Matthew Richardson, Ewa Dominowska, and Robert Ragno. Predicting clicks: Estimating the click-through rate for new ads. In *WWW*, 2007.

[94] Paul R. Rosenbaum. *Observational studies.* Springer New York, 2002.

[95] Paul R. Rosenbaum and Donald B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 1983.

[96] Donald B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Educational Psychology*, 1974.

[97] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo method.* Wiley, 2008.

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

[99] Tetsuya Sakai. Alternatives to bpref. In *SIGIR*, 2007.

[100] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 1990.

[101] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *ICML*, 2016.

[102] John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. Trust region policy optimization. In *ICML*, 2015.

[103] Anne Schuth, Harrie Oosterhuis, Shimon Whiteson, and Maarten de Rijke. Multileave gradient descent for fast online learning to rank. In *WSDM*, 2016.

[104] Shaun R. Seaman and Ian R. White. Review of inverse probability weighting for dealing with missing data. *Statistical Methods in Medical Research*, 2013.

[105] Pannagadatta K. Shivaswamy and Thorsten Joachims. Multi-armed bandit problems with history. In *AISTATS*, 2012.

[106] Karen Sparck Jones and C. J. van Rijsbergen. Report on the need for and provision of an ideal information retrieval test collection. Technical report, University of Cambridge, 1975.

[107] Harald Steck. Training and testing of recommender systems on data missing not at random. In *KDD*, 2010.

[108] Harald Steck. Item popularity and recommendation accuracy. In *RecSys*, 2011.

[109] Harald Steck. Evaluation of recommendations: Rating-prediction and ranking. In *RecSys*, 2013.

[110] Alexander L. Strehl, John Langford, Lihong Li, and Sham M. Kakade. Learning from logged implicit exploration data. In *NIPS*, 2010.

[111] Masashi Sugiyama and Motoaki Kawanabe. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation.* MIT Press, 2012.

[112] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction.* The MIT Press, 1998.

[113] Adith Swaminathan and Thorsten Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *Journal of Machine Learning Research*, 2015.

[114] Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization. In *WWW Workshop on Offline and Online Evaluation of Web-based Services*, 2015.

[115] Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *ICML*, 2015.

[116] Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. In *NIPS*, 2015.

[117] Adith Swaminathan and Thorsten Joachims. Counterfactual evaluation and learning for search, recommendation and ad placement. In *SIGIR Tutorials*, 2016.

[118] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miroslav Dudík, John Langford, Damien Jose, and Imed Zitouni. Off-policy evaluation for slate recommendation. In *ICML Workshop on Computational Frameworks for Personalization*, 2016.

[119] Philip S. Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High-confidence off-policy evaluation. In *AAAI*, 2015.

[120] Steven K. Thompson. *Sampling.* John Wiley & Sons, 2012.

[121] Hale F. Trotter and John W. Tukey. Conditional monte carlo for normal samples. In *Symposium on Monte Carlo Methods*, 1956.

[122] Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML*, 2004.

[123] Gerrit J. J. van den Burg and Patrick J. F. Groenen. GenSVM: A generalized multiclass support vector machine. Technical report, Erasmus University Rotterdam, 2014.

[124] Vladimir N. Vapnik. *Statistical learning theory.* Wiley, 1998.

[125] Vladimir N. Vapnik and Alexey Y. Chervonenkis. Theory of uniform convergence of frequencies of events to their probabilities and problems of search for optimal solutions from empirical data. *Automation and Remote Control*, 1971.

[126] Abraham Wald. A method of estimating plane vulnerability based on damage of survivors. Technical report, Columbia University, 1943.

[127] Lidan Wang, Jimmy J. Lin, and Donald Metzler. A cascade ranking model for efficient ranked retrieval. In *SIGIR*, 2011.

[128] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. Learning to rank with selection bias in personal search. In *SIGIR*, 2016.

[129] Yue Wang, Dawei Yin, Luo Jie, Pengyuan Wang, Makoto Yamada, Yi Chang, and Qiaozhu Mei. Beyond ranking: Optimizing whole-page presentation. In *WSDM*, 2016.

[130] Markus Weimer, Alexandros Karatzoglou, Quoc V. Le, and Alexander J. Smola. COFI RANK: Maximum margin matrix factorization for collaborative ranking. In *NIPS*, 2007.

[131] Jeffrey M. Wooldridge. Inverse probability weighted estimation for general missing data problems. *Journal of Econometrics*, 2007.

[132] Emine Yilmaz, Evangelos Kanoulas, and Javed A. Aslam. A simple and efficient sampling method for estimating AP and NDCG. In *SIGIR*, 2008.

[133] Hsiang-Fu Yu, Cho-Jui Hsieh, Si Si, and Inderjit Dhillon. Scalable coordinate descent approaches to parallel matrix factorization for recommender systems. In *ICDM*, 2012.

[134] Jin Yu, S.V.N. Vishwanathan, Simon Günter, and Nicol N. Schraudolph. A quasi-Newton approach to nonsmooth convex optimization problems in machine learning. *Journal of Machine Learning Research*, 2010.

[135] Yisong Yue and Thorsten Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML*, 2009.

[136] Yisong Yue, Rajan Patel, and Hein Roehrig. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *WWW*, 2010.

[137] Bianca Zadrozny, John Langford, and Naoki Abe. Cost-sensitive learning by cost-proportionate example weighting. In *ICDM*, 2003.