Bias-Aware Multi-Objective Optimization for Online Rankings: Demand-Side Approach to Supply-Side Efficiency*

Malika Korganbekova[1] Alibek Korganbekov[2] Aliya Korganbekova[3]

[1] *The University of Chicago, Booth School of Business*
[2] *Boston University, Questrom School of Business*
[3] *Harvard University, Harvard Business School*

November 5, 2025

CLICK HERE FOR MOST RECENT VERSION

Abstract

Retail shipping is responsible for nearly 25% of global greenhouse gas emissions and costs retailers billions annually. Firms have long treated shipping optimization as a supply chain problem, investing in warehouses, routes, and infrastructure. We show that a largely overlooked lever—product ranking algorithms at the consumer search stage—can deliver substantial efficiency and sustainability gains without capital investment. We redesign ranking as a cost–conversion optimization, embedding shipping costs directly into the ranking algorithm. Because such re-ranking significantly reshapes which products are shown, it challenges both offline evaluation and online learning; we therefore extend the Adaptive Doubly Robust methods with embedding-based neighborhood representations to correct for the competitive context biases. We ran a field experiment with 2.8 million consumers on a major e-commerce platform to test the algorithm. This demand-side intervention reduced shipping costs by 4.2%, cut delivery distances by 34 miles per order, and lowered emissions by 3.8%, while preserving conversions and customer satisfaction. The emissions savings were equivalent to the annual electricity use of 50,000 U.S. households or the emissions of 58,000 gasoline cars. When the ranking policy underperformed, it exposed structural rigidities such as brand-driven demand or inflexible warehouse networks. These results show that ranking algorithms can both shape demand and diagnose supply chain strategy, helping platforms align profitability with sustainability at scale.

Keywords: Retail shipping; Greenhouse gas emissions; Ranking Algorithms; Adaptive Doubly Robust methods; Thompson Sampling, Field Experiments

*M. Korganbekova: malika.korganbekova@chicagobooth.edu, A. Korganbekov: alibekk@bu.edu, A. Korganbekova: akorganbekova@hbs.edu. We thank the data science and analytics team at the company that is the source of our data for their help and numerous insights. We are grateful to Eric Anderson, Anja Lambrecht, Sanjog Misra, and Andrey Simonov for fruitful discussions of this project. We thank the conference participants at NBER Summer Institute, CODE MIT, Northwestern University, IIOC, and Marketing Science for the discussions, comments, and suggestions. All remaining errors are our own.

1 Introduction

The retail sector is responsible for roughly 25% of global greenhouse gas emissions, and as much as 98% of those emissions originate from its supply chain shipping operations.[1] Beyond its environmental impact, shipping also represents a large share of retailers' costs—about 14% of revenues for Amazon ($89.5 billion) and Wayfair ($2.2 billion).[2] Because shipping is simultaneously a cost driver and a major source of emissions, retailers have long pursued shipping optimization. Traditionally, it was treated as an operations problem addressed through fulfillment system redesign, infrastructure investment, and route optimization—approaches that are capital-intensive and slow to implement and adapt (Madeka et al., 2022). More recently, some retailers started to give monetary nudges for consumers to choose slower delivery windows or pickup systems, which allow shipment consolidation and reduce costly last-mile deliveries (e.g., Amazon).[3] However, these interventions occur only *after* consumers have completed their product search and made their purchase decisions.

In contrast, we propose and A/B test a complementary *demand-side approach*: present consumers with options that are cheaper to ship but are still aligned with their preferences *at the consumers' search stage, before the purchase decision is made.* This timing matters because consumers are more flexible before checkout, giving retailers a better opportunity to guide choices toward lower-cost, lower-emission options. The natural way to achieve this is to embed shipping cost signals directly into the product-ranking algorithms.

In this paper, we design a new ranking algorithm that directly accounts for shipping costs. We collaborate with a large online retailer and run a large-scale field experiment to show the effectiveness of this new ranking algorithm.

Existing retail ranking systems often optimize for demand-side revenue proxies such as click-through or conversion rates, while overlooking supply-side costs like shipping. Firms sometimes justify this as a "customer-first" strategy—prioritizing user experience over operational efficiency (e.g., Amazon).[4] In practice, however, the misalignment often stems from organizational frictions (Hortaçsu et al., 2024): large platforms structure responsibilities so that marketing teams own conversion optimization, while supply chain teams focus on cost minimization. In the absence of coordination, rankings end up favoring conversions not out of deliberate customer-centric design, but because objectives are siloed rather than jointly optimized.

To address these organizational frictions and align marketing with supply chain goals, we

[1] See How retailers can reduce carbon emissions.

[2] See Amazon's annual report and Wayfair's 2024 annual reports.

[3] According to MIT Review, last-mile delivery expenses account for up to 53% of total shipping costs. See this article for additional details.

[4] See Amazon SEO: 7 ways to improve your product's search rankings and Amazon's 'Day 1' philosophy.

develop a multi-objective ranking policy that jointly considers conversion likelihood and shipping costs. Let $\hat{\mu}_j$ denote product j's predicted conversion likelihood and c_j its estimated shipping cost. The objective is to design a scoring function $f(\hat{\mu}_j, c_j)$ that maximizes conversions while penalizing products with high shipping costs—but only when those costs do not meaningfully improve conversions. In particular, we avoid penalizing products with higher costs when they provide compensating benefits such as fast or free delivery and flexible returns, while down-weighting products whose high shipping costs add little to customer experience or product quality.

A key challenge in building this multi-objective cost-aware ranking system is the reliability of conversion estimates ($\hat{\mu}_j$). These estimates are not only a function of a product's true appeal but also of its historical position in the rankings (position bias), how often it was shown (exposure or popularity bias), and the quality of the products surrounding it (competitive context or neighborhood bias). Firms usually try to correct for these biases through randomized ranking experiments or off-policy evaluation methods such as Inverse Propensity Weighting (Horvitz and Thompson, 1952) and Doubly Robust estimators (Imbens and Rubin, 2015). However, they are not designed to handle the much larger problem that arises when rankings are reshuffled substantially: competitive context bias. Randomized experiments typically perturb rankings only slightly,[5] so they fail to reveal how products perform when placed in very different competitive contexts (i.e., composition of surrounding products for each focal product changes significantly). Similarly, off-policy evaluation requires sufficient overlap between historical and counterfactual rankings (Khan et al., 2024), but under our new ranking policy fewer than 10% of products on the first pages overlap with the old rankings. In this setting, past conversion data provides little guidance, and competitive context bias becomes the main challenge.

To address competitive context bias, we model conversions as a function of both the focal product characteristics and its competitive neighborhood. A naïve approach would record the exact identifiers of neighboring products for each impression, but this is inefficient: when the ranking policy changes, the set of neighboring products changes as well, eliminating overlap with historical data. Instead, we embed each product into a low-dimensional vector using the text and image models that we trained, augmented with observable features such as price, ratings, and reviews. For each impression, we summarize the neighborhood by aggregating the embeddings of neighboring products. The conversion model conditions on both the focal product's characteristics and the neighborhood representation.

This representation-based approach restores overlap where product identifiers cannot. For example, suppose that product A was next to by products $\{B, C, D\}$ in the historical rankings

[5] Organizational constraints make it hard to justify ranking changes due to high opportunity costs.

and next to products $\{E, F, G\}$ under the new ranking policy. Even though the products are themselves different, the two neighborhoods may yield similar representations (e.g., "visually similar, fast-shipping, mid-price, highly rated"), allowing the model to generalize. We incorporate these features into an Adaptive Doubly Robust framework (Luedtke and van der Laan, 2016), estimating both outcome and propensity models with the focal and neighborhood representations. This yields de-biased, competitive context-aware conversion estimates μ_j that we then use in the final scoring function $f(\mu_j, c_j)$.

We collaborated with a large U.S. online retail platform to conduct an A/B test assessing the practical effectiveness of our de-biased ranking algorithm. The platform sells big-ticket items across hundreds of categories. The platform relies on two fulfillment models: products can either be shipped directly from the seller's warehouse to the consumer (dropship), or from the platform's centralized warehouses. In the latter case, sellers position their products in advance in the centralized facilities, and the platform assumes responsibility for fulfillment, similar to the model used by Amazon.[6]

Our field experiment lasted eight days and involved approximately 2.8 million consumers. On the product ranking pages, at the search stage, consumers were randomly assigned to either a control or a treatment group. In the control group, consumers were shown the platform's standard, conversion-maximizing rankings. In the treatment group, consumers saw rankings generated by our proposed policy. The new ranking algorithm prioritized products that were less costly to ship but still exhibited a high likelihood of conversion. Consequently, consumers in the treatment group were more often presented with products located in warehouses closer to them. Furthermore, because shipping from centralized warehouses tends to be less expensive, the treatment condition placed greater emphasis on products fulfilled through the platform's centralized warehouse facilities.

Compared to the control condition, our proposed ranking algorithm reduces average shipping costs by 2–3%, shortens shipping distances by approximately 34 miles per order, and decreases delivery-related emissions by 3.8%, all without negatively affecting conversions or customer satisfaction. We measure customer satisfaction using product returns, ratings, reviews, and incident rates. We find no significant differences in returns, ratings, or reviews, but incident rates decline by 10% in the treatment condition, consistent with the reduction in long-distance shipments. On the platform side, profits increase by 0.2%. Although this effect is modest in magnitude, it is statistically significant and arises because lower shipping costs are achieved without meaningful changes in revenue.

If this equilibrium is stable, the annual environmental emissions savings would be comparable to the yearly electricity consumption of 50,000 U.S. households or to the emissions produced by 58,000 gasoline-powered cars. These results are substantial, as such savings can be achieved solely

[6] See Amazon Fulfilled by Amazon (FBA) information.

by adjusting product rankings. To better understand the conditions under which the experiment is effective, we further investigate sources of heterogeneity.

We find that the success of our proposed ranking algorithm is driven by both consumer demand flexibility and supplier network flexibility. The experiment was particularly effective in product categories where consumers were more willing to substitute across options. In contrast, in categories where brand identity plays a central role and the products surfaced by the treatment were not necessarily the dominant brand items, consumers were less likely to substitute, leading to a 0.3% decrease in orders. These results suggest that managers should exercise caution: in categories where brand loyalty is strong, consumers are less likely to substitute products. Therefore, the improvements in the supply chain and curated assortments are necessary.

Moreover, most of the impact came from categories with sufficient warehouse flexibility on the seller side. Categories in which more than 50% of distinct products were stocked in centralized warehouses generated the largest gains in both profit and order volume. By contrast, in the areas with limited warehouse coverage, the lack of supply chain flexibility diminished the effectiveness of the proposed approach. For managers, this implies that the re-ranking policy can be highly effective in certain categories and regions, but in others it requires complementary investments in supply chain flexibility.

Another important source of heterogeneity relates to consumers' perceptions of quality. Although the proposed algorithm performed well on average, the gains were not uniform across regions. In the Pacific (U.S. West Coast) region, shipping cost minimization coincided with lower conversion rates despite improvements in displayed product quality, delivery times, and shipping distances. This outcome stemmed from the fact that cost minimization in this region led to the resurfacing of more dropship sellers rather than centralized warehouse products. The West Coast is a distinctive case because of its large geographic distances; in the absence of many centralized warehouses, platforms rely heavily on the sellers' own warehouse networks. However, dropship sellers rarely receive the fast-shipping badge that signals the ability to deliver within one to two days, even when they are capable of doing so. While neither actual product quality nor delivery performance declined, the absence of this badge reduced conversions, underscoring the importance of *perceived*—rather than actual—quality in shaping demand, consistent with Akerlof's theory of signaling (Akerlof, 1978). These findings suggest that platforms must carefully consider how badges are allocated and ensure that accurate quality signals are transmitted to consumers.

We further benchmark the performance of our algorithm against an alternative that employed the same scoring function but used the historical biased conversion estimates $\hat{\mu}_j$, instead of the de-biased ones μ_j. Several weeks prior to our main experiment, the platform conducted a separate

field test in which the treatment condition used our scoring rule $f(\hat{\mu}_j, c_j)$, but relied on historical conversion scores without applying the competitive-context de-biasing procedure that underlies our approach. The control group in that experiment was again the baseline conversion-maximizing ranking algorithm.

The biased scoring rule reduced shipping distances by 8% and shipping costs by 2.2%, but it also led to significant declines in revenue (-0.3%) and profit (-0.1%). Because the only difference between this treatment and ours was the absence of de-biasing, we attribute the weaker performance to the algorithm's failure to surface the correct products under a substantially altered competitive context. These findings underscore the importance of correcting for bias when re-ranking products. Bias correction allows the algorithm to identify and promote products whose quality was previously underestimated—often because they had been disadvantaged by unfavorable placement in the old rankings. By adjusting for these competitive context (neighborhood) effects, our algorithm ranks such products more accurately, increasing their visibility without sacrificing quality.

Overall, we demonstrate that the problem of minimizing shipping costs and emissions can be understood not only from an operations or supply-chain perspective, but also as a marketing problem that can be addressed through marketing-based solutions. Specifically, we show that re-ranking products to reduce shipping costs, while preserving conversions, generates favorable outcomes on both shipping cost and emissions dimensions without compromising consumer outcomes.

This paper provides several important generalizable insights. First, many retail firms view supply chain optimization as separate from the marketing part of the organization. Moreover, despite the fact that fulfillment costs are large and both economically and environmentally significant, they remain absent from most platform ranking objectives. Our findings demonstrate that substantial gains can be realized by reducing these organizational frictions and leveraging consumer-facing marketing tools—such as ranking algorithms—to address supply chain challenges. Second, the issues of bias in historical performance estimates and the failure of overlap conditions when rankings change substantially are not unique to our setting; they can arise in many contexts across different firms. The methodological solution we propose—using a lower-dimensional representation to correct competitive-context bias and ensure overlap—can therefore be applied broadly in other environments.

The rest of the paper proceeds as follows: Section 2 situates our work within the literature on shipping logistics and sustainability; bias correction in adaptive experimentation; and multi-objective optimization in recommender systems. Section 3 outlines the baseline Bayesian ranking framework, which we build on, and provides some illustrative examples. Section 4 describes the proposed multi-objective ranking score. Section 5 develops our Neighborhood-corrected ADR methodology. Section 6 presents the field experiment design, results, and comparison with conversion-based and

non–de-biased algorithms. Section 7 concludes.

2 Related Literature

This paper contributes to several strands of literature: (i) shipping logistics and sustainability, (ii) bias correction in adaptive learning and offline evaluation, (iii) competitive context effects, and (iv) multi-objective optimization (MOO) in ranking systems.

Shipping logistics and sustainability. As noted earlier, shipping efficiency has traditionally been treated as an operations problem. The operations research literature emphasizes supply-side strategies such as shipment consolidation (Xu et al., 2024; Wei et al., 2021; Sonntag et al., 2023; Dror et al., 2007), cost-sharing mechanisms, and fulfillment network optimization (Cheng et al., 2025). In economics, the literature examines low-carbon delivery pathways (Kang et al., 2021), collaborative fulfillment (Juan et al., 2014), and the carbon footprint of retail logistics (Tran and Lam, 2022; Cristea et al., 2013). Marketing research mostly studies consumer-facing information disclosure interventions such as eco-labeling and sustainability claims (Brecko and Kim, 2023; Bollinger et al., 2023; Proserpio et al., 2025).

In contrast, our approach holds both the supply chain and the information disclosure fixed. Instead, we test whether it is possible to achieve shipping and sustainability gains by redesigning the ranking algorithms. This can improve logistical efficiency without altering the fulfillment network or consumer-facing information, offering a scalable way to align operational goals with sustainability objectives.

Bias correction in adaptive learning and offline evaluation. This paper also relates to the literature on bias correction in adaptive learning and offline evaluation. Exposure bias—where items with early success receive disproportionate future exposure—has received considerable attention in the recommendation literature. In adaptive learning settings such as Thompson Sampling, Luedtke and van der Laan (2016); Hadad et al. (2021); Dimakopoulou et al. (2021) identify this bias and propose Adaptive Doubly Robust methods to correct it. We extend their methodology to also address competitive context bias in adaptive learning settings.

Our approach also builds on the offline policy evaluation literature for targeting, bandits, and ranking. In marketing, Hitsch et al. (2024) and Simester et al. (2020) evaluate targeting policies using data from a randomized experiment. In machine learning, Dudík et al. (2014) and Jiang and Li (2016) develop doubly robust (DR) estimators for contextual bandits and sequential decision making, combining model-based and propensity-based adjustments to control bias and variance. In ranking, Kiyohara et al. (2022) and Saito (2020) adapt DR methods to offline evaluation, often

under specific click models or metric definitions. Rather than offline evaluation under static logged data, we address an *adaptive* ranking environment with dynamic neighborhoods.

Competitive context (neighborhood) biases. Competitive context effects have been studied in operations management and marketing under labels such as assortment optimization, shelf spillovers, and substitution–complementarity effects (e.g., Cisternas et al., 2020; Rooderkerk et al., 2013; Hübner and Kühn, 2024). These effects typically arise from *static* display or category management decisions and are often negative—for example, close substitutes cannibalize each other when consumers choose only one product within a shelf or category.

To the best of our knowledge, there is limited work on neighborhood bias in recommender systems. In computer science, a closely related paper is Li et al. (2024). Both our works tackle the same neighborhood bias, but Li et al. (2024) focus on recommender systems, while we focus on ranking algorithms. Moreover, they redefine the uni-objective loss function. In contrast, we de-bias estimates within the standard loss function, add a tractable neighborhood representation, and then integrate it into a new multi-objective cost–conversion ranking policy.

There is also a strand of literature that focuses on network effects, interference, and SUTVA violations in randomized experiments (e.g., Eckles et al., 2017; Holtz et al., 2025; Goli et al., 2024). In those settings, interference arises from pre-specified networks (e.g., social links, supplier–consumer relationships).

Our setting differs in two respects from these streams. First, in adaptive ranking systems, the neighborhoods are *dynamic*: the set of products shown together changes as the ranking algorithm updates. There is no pre-specified network because the context changes as the algorithm updates rankings. Second, spillovers are not necessarily negative. Even under unit demand, consumers may click on or add multiple products to their cart, actions that increase those products' algorithmic attractiveness and improve their chances of being shown—and eventually purchased—by other consumers in the future. Thus, while neighborhood effects arise in other domains, our context is distinguished by its dynamic nature and the possibility of both positive and negative spillovers.

Multi-objective optimization in ranking. Work on MOO in recommender systems typically addresses trade-offs among objectives such as relevance, engagement, and revenue through algorithmic frameworks. There is a large literature in Computer Science. In Marketing, the prominent paper is Wang et al. (2025), who propose a hierarchical approach for multi-objective recommendation on multi-row homepages on Uber, predicting each objective separately and solving constrained optimization problems for within- and across-row ranking. They do offline evaluation on the holdout of the data, in which rankings were randomized. Similarly, we also tune the model on the randomly ranked part. However, we also address neighborhood bias in both offline evaluation

and online training by using a low-dimensional, embedding-based representation of co-displayed items. This approach enables real-time correction for neighborhood covariate shift, complementing standard position and exposure adjustments. Relatedly, Greminger (2022) provides an econometric analysis of the trade-offs among conversion, revenue, and consumer welfare using position-effect heterogeneity and structural modeling, offering counterfactual comparisons of ranking rules rather than a production MOO system.

By integrating bias-corrected estimation into multi-objective ranking and applying it to a sustainability-relevant cost–conversion trade-off, we address a form of dynamic, mixed-sign neighborhood bias not covered in prior work. The resulting system is, to our knowledge, the first large-scale deployment of a ranking algorithm that jointly accounts for both conversions and shipping costs, demonstrating operational and environmental gains without any loss in consumer outcomes.

3 Set-up

We begin by formalizing the baseline Bayesian ranking environment and the multi-armed bandit algorithms that platforms commonly use to display products.[7] Understanding these baseline algorithms is essential, as our interventions build directly on them. This baseline will serve as the foundation for later sections where we discuss the challenges of competitive context bias and motivate our corrected estimator.

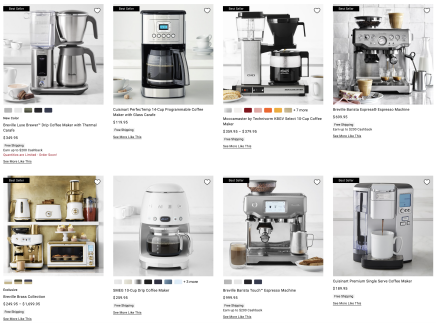
Suppose there is an online platform that needs to decide on the ordering in which products will be presented to the consumers. To achieve that, the platform creates product ranking pages. We assume that these pages are not personalized in the sense that the consumers' historical search patterns do not impact the future rankings they see. Figure 1 shows example ranking pages from Williams Sonoma and Amazon websites.

The platform's objective when ranking the products is to maximize the conversion rates. In each period t, the platform displays a set of M_t products indexed by $j = 1, \ldots, M_t$. Let $y_{jt} \in \{0, 1\}$ indicate whether product j is purchased in period t. Each product j has a vector of features $X_{jt} \in \mathbb{R}^k$ describing observed, possibly time-varying attributes (e.g., position indicators, contextual covariates). In addition, each product has its own latent attractiveness parameter $\theta_j \in \mathbb{R}$ that captures product-specific fixed effects.

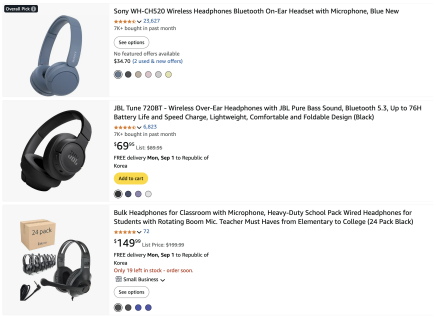
[7] Bayesian methods are commonly used to represent and update uncertainty, balancing exploration with exploitation. For instance, Microsoft's Bing uses the AdPredictor Bayesian probit model for CTR prediction (Graepel et al., 2010); Wayfair implements a Bayesian product-ranking framework with posterior sampling and bias correction (Wayfair, 2022); and marketplaces such as Amazon, eBay, and Walmart employ Bayesian Personalized Ranking or related approaches (Rendle et al., 2009; eBay Inc., 2016).

Figure 1: Ranking pages

(a) Williams Sonoma



(b) Amazon



Purchase model. The probability that product j is purchased in period t is modeled as

$$\mu_{jt} = \Pr(y_{jt} = 1 \mid X_{jt}, \theta_j, \gamma) = \sigma\Big(\theta_j + \gamma^\top X_{jt}\Big), \qquad \sigma(u) = \frac{1}{1+e^{-u}}, \tag{1}$$

where $\gamma \in \mathbb{R}^k$ are global coefficients shared across products, and θ_j is a product-level parameter, and $\sigma(\cdot)$ is a logistic function.

Prior and posterior. Suppose the platform uses a Bayesian approach and places independent priors on the products' quality parameters: $\theta_j \sim \psi_1(\theta),\ \gamma \sim \psi_2(\gamma)$. Given the dataset on the conversion outcomes $(y_{j\cdot})$ and products' characteristics $(X_{j\cdot})$ observed up to time t

$$\mathcal{D}_t \ = \ \{(y_{js}, X_{js}) : s = 1, \ldots, t,\ j = 1, \ldots, M_s\},$$

the likelihood is Bernoulli:

$$\mathcal{L}(\{\theta_j\}, \gamma; \mathcal{D}_t) = \prod_{s=1}^{t} \prod_{j=1}^{M_s} \Big[\sigma(\theta_j + \gamma^\top X_{js})\Big]^{y_{js}} \Big[1 - \sigma(\theta_j + \gamma^\top X_{js})\Big]^{1-y_{js}}$$

The posterior can be written as:

$$p(\{\theta_j\}, \gamma \mid \mathcal{D}_t) \ \propto \ \psi_2(\gamma) \prod_j \psi_1(\theta_j)\, \mathcal{L}(\{\theta_j\}, \gamma; \mathcal{D}_t)$$

Posterior predictive and ranking. For any candidate product j at period $t{+}1$ with features $X_{j,t+1}$, the posterior predictive purchase probability is

$$\hat{p}_{j,t+1} = \int \sigma(\theta_j + \gamma^\top X_{j,t+1})\, p(\{\theta_j\}, \gamma \mid \mathcal{D}_t)\, d\theta_j\, d\gamma$$

One option is for the platform to rank items by $\hat{p}_{j,t+1}$ to maximize expected conversions. However,

many platforms nowadays use multi-armed bandit algorithms instead of purely greedy rankings. The reason is that the platforms may want to give a chance to the products that are not necessarily best but may have potential. Multi-Armed Bandit algorithms balance *exploitation* (favoring products with high predicted conversion rates) and *exploration* (allocating exposure to products with greater uncertainty). One of the popular options is to use **Thompson Sampling**. Under Thompson sampling, the platform draws a parameter sample from the posterior distribution

$$\tilde{\theta}_j, \tilde{\gamma} \sim p(\{\theta_j\}, \gamma \mid \mathcal{D}_t),$$

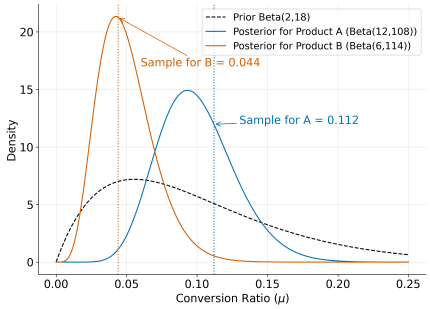
computes sampled scores

$$\tilde{p}_{j,t+1} = \sigma(\tilde{\theta}_j + \tilde{\gamma}^\top X_{j,t+1}), \qquad j = 1, \ldots, M_{t+1}$$

and ranks by $\tilde{p}_{j,t+1}$. This allows the platform to pick high μ_j products while exploring other products too.[8] Regardless of the choice of ranking, the platform collects data on impressions, clicks, and conversions, and updates posterior parameters each period as $t \to t{+}1$.

Illustrative Example. To illustrate how Bayesian ranking algorithms work, suppose there are two products A and B that need to be ranked. Both products have a Beta prior over their conversion rates: $\mu_j \sim \text{Beta}(\alpha_0, \beta_0)$ for $j \in \{A, B\}$ with prior $(\alpha_0, \beta_0) = (2, 18)$. Figure 2 plots the prior density as the dashed black line.

Figure 2: Example of Bayesian Ranking update



Suppose that the products A and B were each shown 100 times on the platform. Product A was purchased 10 times and product B was purchased 4 times. Therefore, the updated posterior distributions for products A and B next period are $Beta(12, 108)$ and $Beta(6, 114)$, respectively[9]

[8] Platforms can also use Upper Confidence Bound (UCB), ε-greedy, and Boltzmann (softmax) exploration for exploitation-exploration.

[9] The posterior updating is: $\mu_A \mid \text{data} \sim \text{Beta}(2 + 10,\ 18 + (100 - 10)) = \text{Beta}(12, 108)$, $\mu_B \mid \text{data} \sim \text{Beta}(2 + 4,\ 18 + (100 - 4)) = \text{Beta}(6, 114)$.

(red and blue lines in Figure 2). Under Thompson Sampling, the platform would sample a score from the posterior distributions instead of taking the posterior predictive mean. For example, for product A, the sampled score could be 0.112 and for product B the score could be 0.044. Therefore, product A would be ranked higher than product B. Had the sampled score been drawn such that the score for B was higher than that for A, then the rankings would change accordingly.

Thus, Thompson Sampling adaptively selects rankings through stochastic draws, balancing exploration and exploitation. While the Bayesian framework and Thompson Sampling demonstrate how platforms manage this tradeoff when estimating conversion probabilities, they overlook products' shipping costs. We therefore introduce our proposed cost-aware scoring rule.

4 Cost-aware ranking score

Building on the baseline Bayesian framework, we now formalize a cost-aware score that combines conversion estimates with product shipping costs. Products would be afterwards ranked according to these scores. While we present one particular formulation of a cost–conversion scoring rule, many alternative specifications are possible.

The intuition behind our scoring function is as follows. We need a ranking score that takes the existing conversion score optimization, while penalizing the high shipping cost items. However, it cannot penalize them uniformly. For some products, the high costs are justified investments into better shipping quality for the product, e.g., fast shipping, free shipping. However, there are parts of the costs that stem from pure inefficiencies. Therefore, the platform needs to incorporate these cost-conversion tradeoffs into the scoring function. Therefore, the proposed scoring rule is:

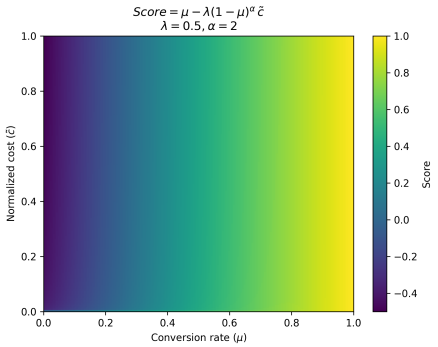
$$\text{score}_{jt} \;=\; \mu_{jt} \;-\; \lambda\,(1-\mu_{jt})^{\alpha}\,\tilde{c}_{jt}, \tag{2}$$

with $\alpha \geq 1$ controlling how quickly the penalty decays for high-conversion items, and $\lambda \geq 0$ governing the cost–conversion trade-off. We normalize costs to place them on a comparable scale with conversion rates: $\tilde{c}_{jt} \;=\; \dfrac{c_{jt}}{\max_j c_{jt}}$.

Figure 3 shows how the rule behaves: high-conversion, low-cost products receive the highest scores (bottom right corner), while low-conversion, high-cost products are penalized most heavily (top left corner). For intermediate cases, the score balances conversion and cost, with penalties decreasing as conversion rates rise or costs fall.

Parameter values (α, λ) must be tuned during the off-policy evaluation of the algorithm. At this stage, the scoring rule provides a principled way to balance conversion and cost, but it implicitly assumes that the underlying estimates of conversion rates (μ_{jt}) are accurate.

Figure 3: Scoring rule as a function of conversion rate and normalized cost



Notes. Lighter colors correspond to higher scores.

4.1 Challenges and existing solutions

In practice, the conversion estimates are rarely unbiased. Logged outcomes reflect not only a product's intrinsic appeal but also systematic distortions introduced by the historical ranking environment. Position and exposure biases are very well-known and established in the literature (Ursu, 2018; Abdollahpouri et al., 2019). In our empirical setting, we observe these biases too. We provide descriptive evidence of these biases in Appendix F. Here, we present the descriptive evidence of competitive context biases, which are more overlooked in the literature. In Section 6, we provide experimental evidence of the importance of these biases.

Competitive context (neighborhood) effects can manifest in several ways. A product may receive disproportionate attention if it is surrounded by high-performing products due to exploration, which could keep it ranked higher over time. Conversely, the same product might be negatively affected by strong neighbors and get low conversions. Had an item be placed in more 'fair' neighborhoods, it would have gotten different posterior distributions.

To investigate this bias in our empirical setting, we take a random sample of products and test for associations between a product's performance and the quality of its neighbors, controlling for the product's own score and average rank. We construct a product-day panel, tracking each product's average rank, its own posterior score, and the posterior scores of products located in its immediate vicinity (± 5 positions) on the same page—chosen to reflect the number of products visible alongside the focal product on a desktop screen.

We estimate the following two-way fixed effects specification:

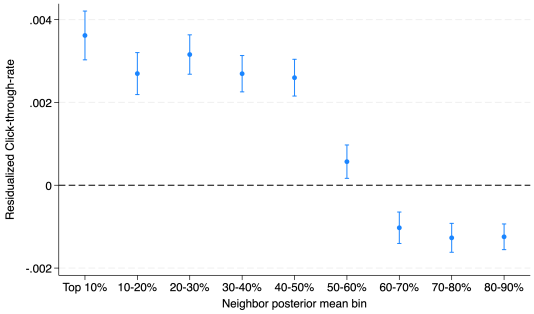
$$y_{jt} = \alpha_j + \alpha_t + \beta_1 rank_{jt} + \beta_2 own\ posterior\ bin_{jt} + \beta_3\ neighbors'\ posterior\ bin_{jt} + \varepsilon_{jt}, \quad (3)$$

where y_{jt} is the outcome of product j at time t, α_j is the product fixed effect, α_t is the time fixed effect,

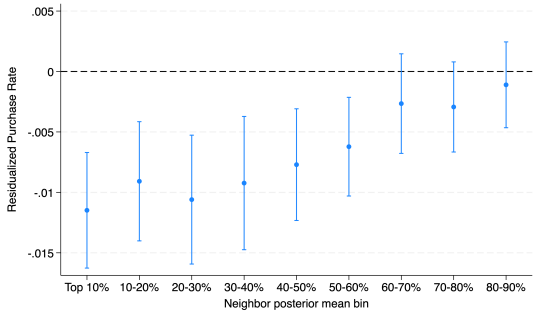
own posterior bin$_{jt}$ is the product j's own posterior mean bin at time t, *neighbors' posterior bin*$_{jt}$ is the product j's neighbors' posterior distribution mean bin. The bins are posterior mean deciles.

Figure 4: Descriptive Neighborhood Effects

(a) Effects on Click-through-rates



(b) Effects on Conversion rates



Notes. Figures plot the estimated neighborhood effects by bins of neighbor posterior scores, controlling for own rank and score. Panel (a) shows that high-scoring neighbors increase click-through rates. Panel (b) shows that these same neighbors reduce conversion rates, suggesting a potential bias in how the platform infers product quality from observed conversions. y-ticks are rescaled for data confidentiality reasons.

We plot the coefficient β_3 by bins of neighbor posterior scores. Figure 4a shows that products with high-posterior-score neighbors receive more clicks, even after controlling for their own rank and score. However, they are less likely to be purchased (Figure 4b).

This is noteworthy because a platform seeking to maximize conversions may interpret low purchase probability conditional on clicking as a signal to deprioritize the product in future rankings. In contrast, had the product been displayed alongside weaker neighbors, the conversion probability conditional on clicks might have been higher, leading the platform to infer better performance and update rankings more favorably.

Thus, given that these neighborhood effects do manifest themselves in the existing ranking systems, we have to take them into account. Under our proposed ranking algorithm, the neighborhood the products are placed in will change considerably. Therefore, we have to take into account that the conversion rates are changing depending on the neighborhood of the product.

At a high level, one way to solve this issue is to directly incorporate the neighborhoods into the conditional purchase probability and extend Equation 1. We begin by defining the true conditional mean outcome. Let $x = X_{jt}$ denote the focal product's features and $v = \phi(\{X_{kt}\}_{k \in S_{jt}})$ summarize the features of its neighbors. Then the conditional purchase probability is

$$m_j(x, v) = \Pr\Big(y_{jt} = 1 \;\Big|\; X_{jt} = x,\ \phi(\{X_{kt}\}_{k \in S_{jt}}) = v,\ \theta_j, \gamma, \delta\Big) = \sigma\Big(\theta_j + \gamma^\top x + \delta^\top v\Big).$$

At time t, the realized probability is therefore $\mu_{jt} = m_j(X_{jt}, \phi(\{X_{kt}\}_{k \in S_{jt}}))$. Let $\hat{m}_j(x, v)$ denote a

fitted estimate of $m_j(x, v)$, and define the display propensity

$$e_j(x, v) = \Pr\Big(j \in M_t \;\Big|\; X_{jt} = x,\ \phi(\{X_{kt}\}_{k \in S_{jt}}) = v\Big).$$

The question is how to correctly incorporate the neighborhoods for estimation and what kind of bias correction method to use. The existing methods for bias correction are: Inverse Propensity Weighting (IPW), Doubly Robust (DR), and Adaptive Doubly Robust (ADR) estimators.

We write all three estimators below.

$$\hat{\mu}_j^{\mathrm{IPW}} = \frac{1}{N_j} \sum_{s: j \in M_s} \frac{y_{js}}{e_j(X_{js}, \phi(\{X_{ks}\}_{k \in S_{js}}))},$$

$$\hat{\mu}_j^{\mathrm{DR}} = \frac{1}{N_j} \sum_{s: j \in M_s} \left[\hat{m}_j(X_{js}, \phi(\{X_{ks}\}_{k \in S_{js}})) + \frac{y_{js} - \hat{m}_j(X_{js}, \phi(\{X_{ks}\}_{k \in S_{js}}))}{e_j(X_{js}, \phi(\{X_{ks}\}_{k \in S_{js}}))} \right],$$

$$\hat{\mu}_j^{\mathrm{ADR}} = \frac{1}{\sum_{s: j \in M_s} w_{js}} \sum_{s: j \in M_s} w_{js} \left[\hat{m}_j(X_{js}, \phi(\{X_{ks}\}_{k \in S_{js}})) + \frac{y_{js} - \hat{m}_j(X_{js}, \phi(\{X_{ks}\}_{k \in S_{js}}))}{e_j(X_{js}, \phi(\{X_{ks}\}_{k \in S_{js}}))} \right],$$

where w_{js} are stabilization weights, such as $w_{js} = 1/e_j(X_{js}, \phi(\{X_{ks}\}_{k \in S_{js}}))$. These estimators target the mean performance of product j, correcting for the policy-dependent distortions in the logged data while incorporating neighborhood effects through the extended feature vector (x, v).

Our approach will build on Adaptive Doubly Robust estimator because it is more stable and less sensitive to mis-specified or extreme propensities, especially in finite samples (Luedtke and van der Laan, 2016).

In the next section, we introduce our proposed ranking policy and describe how neighborhoods are represented through image and textual embeddings together with observable product characteristics.

5 Proposed Ranking Policy

In Section 4 we introduced a cost–conversion scoring rule as the target ranking objective. The challenge is that this rule requires unbiased estimates of conversion probabilities, yet naïve estimators are distorted by position, exposure, and neighborhood biases. As we mentioned above, position and exposure (popularity) biases are well-known and solutions to mitigate them are established. To address the neighborhood or competitive context bias, we develop the Neighborhood Adaptive Doubly Robust (N-ADR) estimator, which produces debiased conversion predictions suitable for use in adaptive ranking. Once these estimates are obtained, they can be combined with product costs through the scoring rule to yield a policy that balances conversion and cost.

5.1 Setup and Target Estimand

We now formalize the ranking environment in order to define the object of interest for estimation. Consider J products, K display positions, and T time periods. Products are indexed by $j \in \{1, \ldots, J\}$, positions by $k \in \{1, \ldots, K\}$, and periods by $t \in \{1, \ldots, T\}$. In each period t, the platform displays an ordered slate—that is, the ranked list of products shown to the user:

$$A_t = (a_{1t}, \ldots, a_{Kt}),$$

where a_{kt} denotes the product shown in position k. Each position contains exactly one product, and each product appears at most once in a slate. For a focal position k, the corresponding neighborhood is defined as

$$S_t^{(-k)} = \{(a_{\ell t}, \ell) : \ell \neq k\},$$

that is, the set of co-displayed products together with their positions. The exposure of product j in position k at time t is indicated by

$$Z_{jt}(k) = \mathbf{1}\{a_{kt} = j\},$$

so that $\sum_j Z_{jt}(k) = 1$ for each k and $\sum_k Z_{jt}(k) \leq 1$ for each j.

Let $y_{jt}(k, S) \in \{0, 1\}$ denote the potential purchase outcome for product j in period t if it were displayed in position k with neighborhood S. In particular, when the realized neighborhood in period t is $S_t^{(-k)}$, the observed outcome is

$$y_{jt} = \sum_{k=1}^{K} Z_{jt}(k)\, y_{jt}(k, S_t^{(-k)}).$$

Let X_{jt} denote covariates associated with product j at time t. The structural regression function is defined as

$$m_{j,k}(X, S) = \mathbb{E}[y_{jt}(k, S) \,|\, X_{jt} = X,\, S]\,,$$

which represents the true conversion probability of product j in position k with covariates X and neighborhood S.

To evaluate products under a counterfactual policy, let $q_{0,k}(S \mid X)$ denote the target distribution of neighborhoods for position k given covariates X. The neighborhood-marginalized conversion probability for product j in position k is then

$$\mu_{j,k}^{\star} = \mathbb{E}_X\, \mathbb{E}_{S \sim q_{0,k}(\cdot|X)}[m_{j,k}(X, S)].$$

Finally, to obtain a position-agnostic performance measure, we aggregate across positions using pre-specified weights $w_k \geq 0$[10] with $\sum_{k=1}^{K} w_k = 1$:

$$\mu_j^\star = \sum_{k=1}^{K} w_k \, \mu_{j,k}^\star.$$

Naïve sample averages of observed outcomes do not identify $\mu_{j,k}^\star$ because neighborhoods are endogenously determined by the ranking policy as discussed in Section 4.

5.2 Neighborhood-Adaptive Doubly Robust Estimator

The target estimand $\mu_j^\star$ requires unbiased conversion probabilities, but logged outcomes are distorted by both adaptive exposure and neighborhood composition. To address this, we propose the *Neighborhood-Adaptive Doubly Robust* (N-ADR) estimator. N-ADR combines predictions from an outcome model with two reweighting steps—one for exposure, one for neighborhoods—so that products are evaluated as if they had been displayed under the counterfactual policy of interest.

Formally, let $m_{t-1}^{(j,k)}(X, S)$ denote an estimated outcome model for the structural regression function $m_{j,k}(X, S)$, fit on data up to time $t-1$. Building on Dimakopoulou et al. (2021), we write the N-ADR pseudo-outcome for product j in position k at time t as

$$\Gamma_t^{(j,k)} = m_{t-1}^{(j,k)}(X_{jt}, S_t^{(-k)}) + \frac{Z_{jt}(k)}{\pi_t(j,k)} \Big[y_{jt} - m_{t-1}^{(j,k)}(X_{jt}, S_t^{(-k)}) \Big] \, w_t(S_t^{(-k)} \mid X_{jt}, j, k),$$

where $\pi_t(j,k)$ is the exposure propensity and $w_t(S \mid X, j, k)$ is the density ratio between the target and logged neighborhood distributions. The first term imputes the expected outcome, while the second term adds a residual correction scaled by the inverse exposure probability and the neighborhood weight. Averaging $\Gamma_t^{(j,k)}$ over time yields $\hat{\mu}_{j,k}^{\text{N–ADR}}$, and aggregating across positions gives $\hat{\mu}_j^{\text{N–ADR}}$.

Outcome model. In practice we specify $m_{t-1}^{(j,k)}$ as a logistic regression with product fixed effects, position indicators, and neighborhood summaries:

$$m_{t-1}^{(j,k)}(X_{jt}, S_t^{(-k)}) = \sigma\Big(\theta_j + \gamma^\top X_{jt} + \delta^\top \psi(S_t^{(-k)})\Big),$$

where θ_j are product effects, γ are global coefficients on covariates, and δ capture the influence of the neighborhood representation $\psi(S_t^{(-k)})$. More flexible machine-learning models can also be used: semi-parametric (e.g., spline-based regressions); or nonparametric/Machine Learning models (e.g., random forests, neural networks, gradient boosting).

[10] We use w_k that the platform obtained from random ranking holdout experiments.

Exposure propensities. Under Thompson sampling, the probability $\pi_t(j,k)$ that product j is placed in position k given past history has no closed form, because it depends on posterior sampling of model parameters. Let H_{t-1} denote the interaction history up to time $t-1$, and let $p(\theta, \gamma \mid H_{t-1})$ denote the posterior distribution over the outcome model parameters θ (product effects) and γ (covariate coefficients). We approximate $\pi_t(j,k)$ by Monte Carlo simulation: drawing R posterior samples $(\theta^{(r)}, \gamma^{(r)}) \sim p(\theta, \gamma \mid H_{t-1})$, re-ranking products for each draw, and computing

$$\hat{\pi}_t(j,k) = \frac{1}{R} \sum_{r=1}^{R} \mathbf{1}\{j \text{ is assigned to position } k \text{ under draw } r\}\,.$$

Neighborhood representation. To capture neighborhood effects, we summarize the relationship between a focal product and its immediate neighbors. We define the neighborhood as the set of products within ± 5 ranks of the focal item, corresponding to those typically visible on a desktop screen. For each product we construct a joint embedding that combines image and textual representations with numerical attributes such as price, ratings, and reviews. We train image embeddings using the Siamese Neural Network with 10 million product images (see Appendix A for details). We use Google's Universal Sentence Autoencoders to construct textual embeddings from the platform's product titles and textual descriptions (see Appendix B for details). The neighborhood summary $\psi(S_t^{(-k)})$ is defined as the set of similarities between the focal product's embedding and those of its visible neighbors, capturing how strongly a product resembles or differs from its immediate competitors.

Neighborhood weights. Logged neighborhoods differ systematically from those under the target policy $q_0(S \mid X)$. To reweight them, we estimate density ratios using the embedding-based summaries $\psi(S)$. Specifically, we simulate neighborhoods under q_0, compare them to those observed under the logging policy, and train a classifier to distinguish the two. If $h(S)$ is the classifier's predicted probability that S comes from the target policy, the implied density ratio is

$$\hat{w}_t(S) = \frac{h(S)}{1 - h(S)}.$$

Weights are clipped to control variance.

Integration. N-ADR has a double robustness property: it is consistent if either the outcome model is correctly specified, or if both exposure propensities and neighborhood weights are correctly estimated. In online settings, we update running estimates of $\hat{\mu}_{j,k}^{\text{N–ADR}}$ and their variances, and Thompson sampling draws exploration scores from these bias-corrected posteriors. Concretely,

letting $\Gamma_s^{(j,k)}$ denote the N-ADR pseudo-outcome, we maintain

$$\hat{\mu}_{j,k,t}^{\text{N–ADR}} = \frac{\sum_{s=1}^{t} \sqrt{\pi_s(j,k)}\,\Gamma_s^{(j,k)}}{\sum_{s=1}^{t} \sqrt{\pi_s(j,k)}}, \qquad \hat{\sigma}_t^{2,(j,k)} = \frac{\sum_{s=1}^{t} \pi_s(j,k)\left[(\Gamma_s^{(j,k)} - \hat{\mu}_{j,k,t}^{\text{N–ADR}})^2 + 1\right]}{\left(\sum_{s=1}^{t} \sqrt{\pi_s(j,k)}\right)^2}.$$

At each period, Thompson sampling draws scores from

$$\mathcal{N}\left(\hat{\mu}_{j,k,t}^{\text{N–ADR}},\ \hat{\sigma}_t^{2,(j,k)}\right)$$

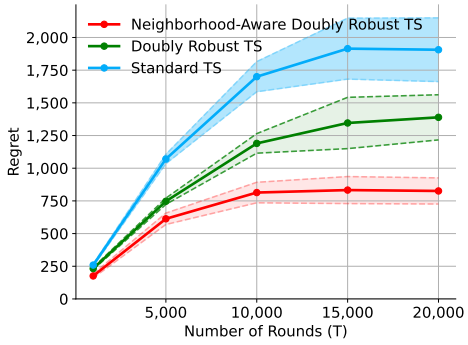
for each (j,k), ensuring continued exploration while ranking products according to bias-corrected, neighborhood-adjusted conversion estimates. We provide assumptions necessary for the consistency of this estimator in the Appendix I.

5.3 Simulations

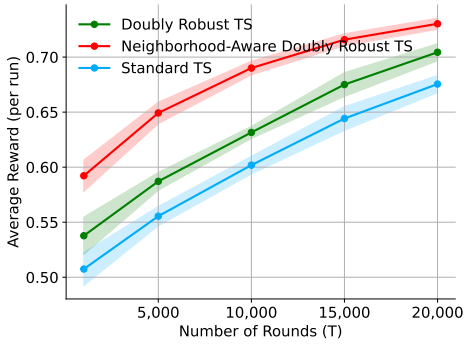
We run simulations to test how well the algorithm performs. Details of the simulation setup are provided in Appendix E. Here, we present the main results. We run the simulations for 20,000 periods and calculate regret and average reward under three approaches: the standard Thompson sampling algorithm, the neighborhood-aware doubly robust Thompson sampling, and the doubly robust Thompson sampling.

Figure 5: Simulation results for algorithm performance

(a) Cumulative regret over time



(b) Average reward over time



Notes. Results are from simulated experiments comparing algorithm performance. Panel (a) plots cumulative regret, where lower values indicate better performance relative to the optimal policy. Panel (b) shows the evolution of average reward, where higher values indicate better alignment with the optimal ranking.

Figure 5 shows that the neighborhood-corrected ADR can achieve lower cumulative regret (Panel a) and higher average rewards (Panel b) in our simulations. This gives us confidence that the algorithm can work well. In our application, we use the conversion estimates de-biased using this approach and tune the scoring function from Section 4 on the random rankings holdout (5% of the

traffic) that the platform always maintains. This is a standard way to evaluate the model (Wang et al., 2025).

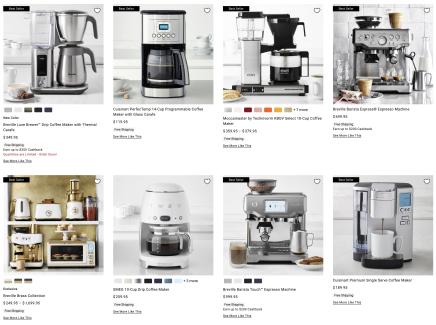
6 Empirical Application

We field test the proposed methodology on a large-scale online retailer's website that sells durable goods. Before presenting the experimental results, we first describe the platform's interface and product ranking algorithm.

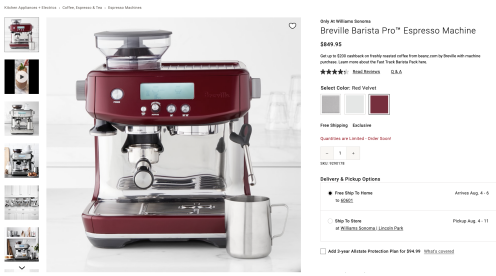
Platform interface. Since the platform we work with chose to stay anonymous, we use the screenshots from another retailer, namely, Williams Sonoma. When consumers arrive on the platform, they can either enter a search query via a keyword box or select from predefined product categories (e.g., headphones, kitchen appliances). Once a category is selected, consumers are directed to ranking pages. Each ranking page displays around 50 products, organized in a matrix form. Figure 6a shows an example of the ranking page grid. Each product tile on a ranking page includes an image, sale price, average rating, number of reviews, shipping information, and the day the product would be delivered. Some products may have 1- or 2-day fast shipping badges if they can be shipped fast. The platform decides which products get fast shipping badges. Most products are under free shipping, in the sense that there is no additional shipping fee at the checkout.

Figure 6: Platform Interface

(a) Product Ranking Page



(b) Product Detail Page



Notes. The figure illustrates a typical product ranking page (left) and product detail page (right) on williams-sonoma.com as an example.

Once on a ranking page, consumer has an option to scroll down the ranking page or click on the next page to see the next set of 50 products. Clicking on a product on the ranking page takes the user to a product detail page (Figure 6b), where they can view more comprehensive information, such as specifications and customer reviews. Following exposure to the ranked product assortment, consumers may continue browsing, complete a purchase, or exit the platform. These interactions

follow a standard pattern common to most e-commerce environments.

Once an order is placed, consumers who choose to purchase a product receive fulfillment through one of two distinct channels. Under the *dropship* model, the third-party seller holds inventory and ships directly to the consumer from its own warehouse. Alternatively, orders may be fulfilled through *centralized warehouses* that the platform owns and maintains. The platform we work with has multiple centralized warehouses across the U.S. Sellers can choose to send their products to these centralized warehouses—i.e., "forward position" them—so that shipping is handled entirely by the platform. Sellers pay the platform a storage fee to place their products in centralized warehouses. However, they do not view this fee as the primary economic cost. Conversations with sellers suggest that the relevant cost they consider is the opportunity cost of not allocating inventory to other platforms, particularly if a positive demand shock were to occur elsewhere.[11] A key benefit of using the platform's centralized warehouses is that products are more likely to receive fast-shipping badges, since they can be delivered to customers more quickly. Importantly, the platform pays for the shipping cost regardless of whether the item is shipped via dropship or centralized warehouses.

Product Ranking Algorithm. The platform uses the Bayesian ranking algorithm with Thompson sampling to rank products. While we cannot disclose every detail of the algorithm, the algorithm is standard and similar to the one described in Section 3. The platform maximizes the probability of conversion subject to some product-level parameters and also accounts for the position effect in their logistic regression. The platform has some prior on the parameters and updates those parameters as more clicks, conversions arrive. The platform uses Thompson sampling, drawing from each product's posterior distribution and ranking them according to these scores. As data arrives, each product's posterior quality distribution is updated. Typically, the algorithm gets updated several times a day, which is why within a day, the rankings might differ across pageloads. Appendix G.3 has some examples of posterior score evolutions.

Next, we present the data we use in the paper.

6.1 Data

In this section, we describe the main components of our data. We have full three-year long data, which covers the equal periods before and after our experiment.

Clickstream data. At the consumer-device-session-level, we observe full consumer browsing history on the website. Starting from the moment when consumers land on the website, we observe which categories they searched for and the ordered list of products they observed on the ranking

[11] From the perspective of storage rent, sellers point out that they would face comparable costs if they stored the products in their own warehouses.

pages. For each product on each ranking page, we observe a product's price, ratings, numbers of ratings, and the availability of the fast shipping badge, the get-it-by dates, and promised delivery days. Thus, we can fully re-create the view of the page as it was shown to the consumers. Next, we observe whether consumers scrolled the page (i.e., we know what they actually viewed on the website), clicked on an item, added it to cart, or purchased.

Transactions and shipping data. At the consumer-level, we observe which products were ordered by a consumer, time of transaction, quantity, and price. We know whether the chosen product had fast delivery promises. We also observe which supplier was chosen to fulfill the order, as well as the address of a warehouse from which a product was shipped and consumers' destination addresses.

Shipment tracking data. We observe shipment tracking information provided by FedEx and UPS. Specifically, we observe the timestamp and origin scan of the shipment and all subsequent tracking scans up to the destination together with the timestamps. This data allows us to calculate full travel history of the product from the seller's warehouse to the consumer or from the seller's warehouse to the centralized warehouse to the final consumer.

Product characteristics. We also use data on product characteristics, such as weight and girth of the products. This allows us to distinguish small, medium, and large parcel items.

Seller characteristics. For each seller on the platform, we observe the complete set of products they offer, the introduction date of each product, and the date the seller first joined the platform and the date they joined centralized warehouses (if applicable). Sellers are typically companies operating multiple warehouses; our data are at the seller-warehouse level. For each warehouse, we observe the opening date and, if applicable, the contract signature date corresponding to the seller's enrollment in the platform's centralized distribution network.

Prices and product wholesale and shippings costs. We observe a daily panel of each product's prices, wholesale costs set by the seller, any discounts/allowances provided by the seller and shipping costs. We use these data to calculate product markups and margins to assess the platform's profitability.

Product returns and incidents data. For each product ordered on the platform, we observe a post-purchase return indicator, capturing whether the product was ultimately returned by the consumer. Additionally, we observe an incidence indicator, identifying whether the product experienced damage or other delivery-related issues during shipment.

Algorithmic posterior scores. We have an hourly panel of algorithm posterior product scores for the entire sample period. This is necessary for the propensity score calculations in the Thompson Sampling.

Overall, we have rich dataset that track consumer actions, platform design, and seller and characteristics, and shipping information. Next, we present and analyze the field experiment.

6.2 Experiment description

The experiment was designed to test multi objective optimization under a de-biased ranking algorithm. It ran for eight days during a non-holiday period on 2.8 million consumers.[12]

The control group consumers were shown the business-as-usual, conversion-based rankings. In the treatment group—labeled the "Proposed Algorithm" group—consumers saw rankings generated by our proposed method.

Table 1 summarizes consumer search patterns pre-experiment, showing that the average consumer clicks and purchases products within the first two pages (median: page 1). Given that consumers rarely browse beyond page 4—and in light of engineering constraints—we implemented the algorithm only for the first four pages of search results.

Table 1: Search patterns

	Min	**Mean**	**Median**	**Max**	**St.Dev.**	**Observations**
# products clicked	0.00	1.32	0.00	1123.00	3.70	2,819,110
clicked rank	1.00	65.28	27.50	8456.00	135.35	1,137,618
clicked page	1.00	1.97	1.00	177.00	2.77	1,137,618
# purchased products	0.00	0.02	0.00	34.00	0.16	2,819,110
purchased rank	1.00	58.93	17.50	9317.00	162.68	40,494
purchased page	1.00	1.88	1.00	195.00	3.34	40,494

Notes. The table summarizes consumer search behavior before the experiment.

Randomization was conducted at the consumer level, ensuring that all devices and browsers associated with a given consumer were consistently assigned to the same treatment condition. The platform cannot account for consumers who delete cookies or otherwise reset their identifiers, but we assume such behavior is balanced across experimental groups. Randomization checks are provided in Appendix K (Table K10).

One potential concern is the violation of the Stable Unit Treatment Value Assumption (SUTVA) in the experiment. For example, SUTVA could be violated if clicks or purchases of products in one treatment condition influenced their overall posterior scores, thereby affecting outcomes in other treatment conditions. Evidence from the firm indicates that the algorithms were separated to avoid such spillovers, and we confirm this using panel data on algorithm posteriors. Another potential

[12]The experiment actually lasted 12 days, but due to engineering constraints and a burn-in period, we analyze only 8 days of clean data.

SUTVA concern involves supply-chain considerations. We test for both issues and find no evidence of violations. See Appendix C for additional details.

6.3 Experimental Results

We now present the experimental results. We first examine the shipping costs displayed at each page load and those visible on consumers' screens (in-view). We estimate the following specification, where the data are at the product–treatment group level (jt):

$$y_{jt} = \alpha + \beta\, treatment_{jt} + \varepsilon_{jt}, \tag{4}$$

where j indexes products, t indexes treatment groups, y_{jt} is the logged shipping cost outcome (page load, in-view, or order), treatment_{jt} is an indicator for assignment to the proposed algorithm, and ε_{jt} is the error term.

Coefficient β measures the average difference in shipping shipping cost outcomes across products when they are shown under the treatment algorithm versus the control, reflecting how the algorithm reallocates which consumers see which products. Note that the shipping cost of product j may be high for some locations but not for others.

Table 2 shows that page-load shipping costs were 4.2% lower in the treatment group compared to the control (Column 1), and in-view costs were 4.1% lower (Column 2). For products that were eventually purchased, shipping costs were even lower—by 4.8% (Column 3).

Table 2: Shipping costs in the experiment

	OLS		
	(1)	(2)	(3)
	log(pageload shipping costs)	log(in-view shipping costs)	log(order shipping costs)
Proposed Algorithm	-0.042***	-0.041***	-0.048***
	(0.000)	(0.001)	(0.012)
Intercept	3.413***	3.416***	3.041***
	(0.001)	(0.001)	(0.010)
Observations	5,663,220	5,319,672	21,726
Clusters	1,670,331	1,585,462	9,425

Notes. This table reports the estimated treatment effects on shipping costs (Equation 4). The unit of observation is the product–treatment group. Standard errors are clustered at the product level, and the number of clusters is reported. Columns (1) and (2) present results for shipping costs displayed at page load and in-view, respectively. Column (3) reports results for products that were purchased. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

These results are largely driven by the fact that most purchases occur within the first two pages of the ranking results, where the treatment intensity was highest. Table 3 shows that, on the first page, both page-load and in-view shipping costs were significantly lower than in the control—both by 4.9%.

Table 3: Shipping costs on the first two pages

	Log(pageload shipping costs)		*Log(in-view shipping costs)*	
	(1)	(2)	(3)	(4)
	page 1	page 2	page 1	page 2
Proposed Algorithm	-0.049***	-0.012***	-0.049***	-0.005
	(0.003)	(0.004)	(0.004)	(0.004)
Intercept	3.201***	3.240***	3.191***	3.240***
	(0.004)	(0.004)	(0.004)	(0.004)
Observations	320,960	320,294	283,168	306,886
Clusters	108,631	120,777	93,591	115,521

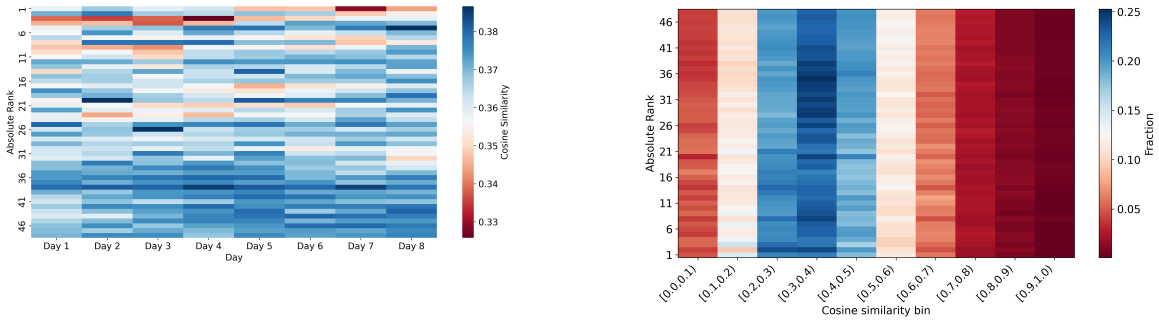
Notes. The table reports treatment effects on shipping costs for products displayed on the first two pages of search results (Equation 4). The unit of observation is the product–treatment group. Standard errors are clustered at the product level, and the number of clusters is reported. Columns (1)–(2) report page-load shipping costs; Columns (3)–(4) report in-view shipping costs. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

The reduction in shipping costs arises from changes in the distribution of products displayed in the rankings. To compare control and treatment ranking results, we use product image and textual embeddings and compute cosine similarities between products in the same rank on the same day across the two groups. Figure 7a shows that the average similarity between products at the same rank in the treatment and control groups was no higher than 0.39, which is quite low. For comparison, in the pre-experimental period, the median similarity of products at the same rank on the same day across page-loads was around 0.7–0.8.[13] Thus, the products displayed in treatment differed substantially from those in control.

Figure 7: Ranking changes in treatment and control groups

(a) by rank and day

(b) by cosine similarity bin and rank



Notes. Panel (a) shows the average cosine similarity between products at the same rank in the treatment and control groups across days, where lower values indicate greater differences in products displayed. Panel (b) shows the distribution of products by cosine similarity bins and their corresponding ranks.

Binning products by cosine similarity into 10 bins and calculating the fraction shown at different ranks confirms that the majority of products in treatment groups had similarities between 0.2 and

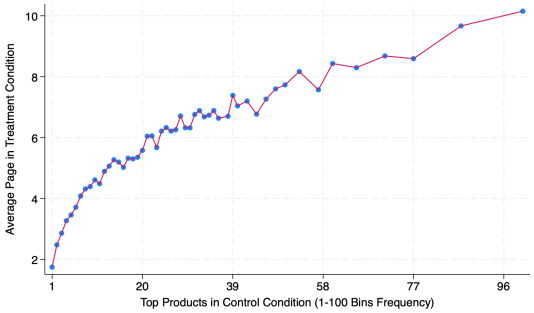
[13]The reason that the rankings differ within a day is that the platform's algorithm gets updated posteriors several times a day.

0.5 to the control group products on the same ranks. Again, this is low relative to typical platform operations (Figure 7b).

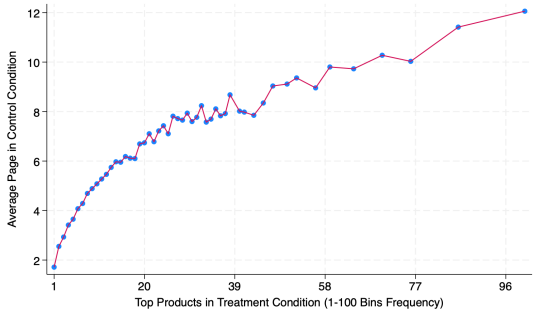
We also examine rank movements of products originally shown in the top positions in the control (treatment) condition and track the average page on which they appeared in the treatment (control) condition. Figure 8 shows that the experiment caused substantial reordering of products: items appearing on the first page in the control (treatment) condition could be relegated to the second page or beyond in the treatment (control) condition.

Figure 8: Rank movements between treatment and control groups

(a) Control products in treatment rankings



(b) Treatment products in control rankings



Notes. Panel (a) shows the average page positions of products that appeared in top ranks in the control group when placed in the treatment ranking. Panel (b) shows the reverse—average page positions of products originally ranked at the top in the treatment group when placed in the control ranking. Both panels indicate substantial reshuffling of products between experimental conditions.

Quality distribution. At the same time, if we look at the distribution of product quality in the control and treatment (proposed algorithm) conditions, we don't see the evidence of degradation. Figures 9a and 9b show that the ratings and number of reviews of products across ranks were quite similar between the two groups. Prices in the treatment group were slightly lower (Figure 9c), reported as rescaled values for data anonymity.[14] The share of products with fast-delivery promises was, on average, higher (Figure 9d). Formal t-tests are provided in Appendix G.2.

Origin of shipping cost minimization. The higher prevalence of fast-delivery promises in the treatment group suggests that the cost-minimization objective promoted products stored in centralized distribution centers to higher ranks.

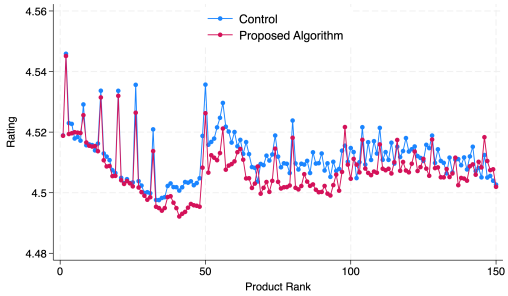
Thus, the algorithm appears to have achieved cost minimization without degrading the quality of products surfaced to consumers. Next, we investigate the downstream effects on shipping distance, conversions, and profits.

Downstream effects. As a result of the proposed algorithm, there were meaningful improvements in the average distances traveled by products. We estimate the following specification for the

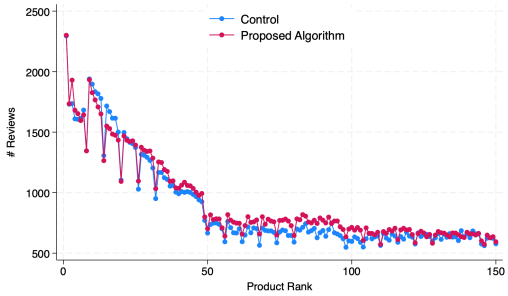
[14] Lower prices are not surprising given that lower shipping costs products are being featured.

Figure 9: Distribution of product quality

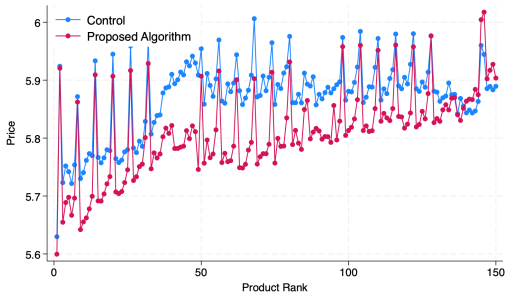
(a) Ratings



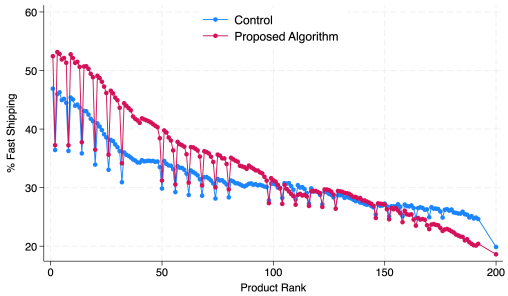
(b) Reviews



(c) Price



(d) Fast Shipping Badges



Notes. The figure compares the distribution of product quality metrics between the control and treatment (proposed algorithm) groups across product ranks. Panel (a) shows average product ratings, and Panel (b) shows the average number of reviews. Panel (c) presents average prices (rescaled for data anonymity), and Panel (d) reports the share of products with fast-shipping badges.

consumers who ordered products:

$$y_{ij} = \alpha + \beta treatment_{ij} + \varepsilon_{ij}, \tag{5}$$

where ij indexes consumer-ordered product.

Table 4 shows that the average shipping distance decreased by 34 miles per order, or 4.6%, compared to the control group (Columns 1 and 2).[15] Delivery time decreased by 1.8% (Column 3) as a result of closer shipments. Moreover, the share of very close shipments—less than 250 miles and 250–1500 miles—increased by 4.95% and 1.8%, respectively (Columns 4 and 5). In contrast, the share of longer-distance orders decreased by 8.1% to 17% (Columns 6–8; logit marginal effects).

Table 4: Effects of the proposed algorithm on shipping distances and delivery times

	OLS			*Logit*				
	(1) distance	(2) log(distance)	(3) log(delivery)	(4) < 250 mi	(5) $250 - 1500$ mi	(6) $1500 - 2000$ mi	(7) $2000 - 2500$ mi	(8) > 2500 mi
Proposed Algorithm	-34.800***	-0.046***	-0.018***	0.062***	0.044***	-0.123***	-0.094***	-0.197***
	(5.300)	(0.008)	(0.004)	(0.018)	(0.015)	(0.027)	(0.024)	(0.055)
Intercept	877.154***	6.298***	1.638***	-1.329***	0.323***	-2.351***	-2.155***	-3.912***
	(3.762)	(0.006)	(0.003)	(0.013)	(0.010)	(0.018)	(0.017)	(0.037)
Baseline probability (control)	.	.	.	20.939	58.015	8.697	10.389	1.960
% Point effect (treatment vs control)	.	.	.	1.037	1.078	-0.926	-0.845	-0.344
Relative % effect (treatment vs control)	.	.	.	4.950	1.859	-10.650	-8.129	-17.561
Observations	75,285	75,285	75,104	75,285	75,285	75,285	75,285	75,285

Notes. The table reports treatment effects from Equation 5 for ordered products. Data is at the consumer-ordered product level. Columns (1)–(3) present OLS estimates for average shipping distance (miles), percentage change in shipping distance, and delivery time (days). Columns (4)–(8) report logit marginal effects for the probability that an order falls within a given distance category. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Indeed, these improvements were driven by the fact that more orders began arriving from centralized warehouses rather than from dropship suppliers. Table 5 shows that orders fulfilled from centralized warehouses increased by 12.8% (Column 1 - Relative % effect of logit). Most of this effect comes from small parcels rather than large parcels, as the majority of orders for this retailer are typically small parcels.

As a result of the decreases in shipping costs and shipping distances, one would expect some improvement in profits—provided demand was not negatively affected. Table 6 reports the results of the following specification ran on the entire 2.8 million consumers:

$$y_i = \alpha + \beta treatment_i + \varepsilon_i, \tag{6}$$

where y_i is the outcome for consumer i, $treatment_i$ is an indicator for the assignment to the proposed

[15] This distance includes both the distance traveled from the dropship warehouse to the consumer and, for the products in the centralized warehouses, it includes the distance from the supplier to the centralized warehouses and to the final consumer.

Table 5: Effects of the proposed algorithm on fulfillment source (centralized fulfillment centers (FC) vs. dropship)

	Logit		
	(1) Centralized FC order	(2) Centralized FC order – Small Parcel	(3) Centralized FC order – Large Parcel
Proposed Algorithm	0.170***	0.172***	0.095
	(0.016)	(0.016)	(0.095)
Intercept	-0.979***	-0.981***	-0.927***
	(0.012)	(0.012)	(0.067)
Baseline probability (control)	27.309	27.278	28.347
Perc.Point effect (treatment vs control)	3.499	3.544	1.973
Relative % effect (treatment/control)	12.814	12.994	6.960
Observations	75,285	73,140	2,145

Notes. The table reports treatment effects on the share of orders fulfilled from centralized warehouses versus dropship suppliers. Data is at the consumer-ordered product-level. Estimates are based on the Equation 5. Column (1) shows the overall effect, while Columns (2) and (3) break down the results for small parcels and large parcels, respectively. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

algorithm, ε_i is the error term.

Table 6 shows that orders and revenue changed very little, while profit increased by 0.2%. This is consistent with the fact that shipping costs account for only for a portion of the retailers' revenue.[16]

Table 6: Effects of the proposed algorithm on orders, revenue, and profit

	Logit	*OLS*		
	(1) clicked	(2) log(# orders)	(3) log(revenue)	(4) log(profit)
Proposed Algorithm	-0.017***	0.000	0.002	0.002**
	(0.002)	(0.000)	(0.001)	(0.001)
Intercept	0.264***	0.016***	0.112***	0.083***
	(0.002)	(0.000)	(0.001)	(0.001)
Observations	2,814,872	2,830,223	2,830,223	2,830,223

Notes. The table reports treatment effects on total orders, revenue, and profit. Estimates are based on Equation 6. Data is at the consumer level. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Customer satisfaction. The primary objective of the algorithm was to minimize costs without degrading the consumer experience. We therefore examine customer satisfaction metrics, including product returns, incident rates, and post-purchase reviews and ratings. Table 7 shows no statistically significant differences in return rates between the proposed algorithm and control conditions. However, the incident rate decreased by 9.9% under the proposed algorithm, likely due to shorter shipping distances, which reduce the likelihood of product damage.

[16] If the treatment reduces shipping costs by 4.8% and shipping constitutes about 4.2% of total revenue, the effect on overall profit will be relatively small. Let total profit be $\pi = R - (\text{COGS} + \text{pick/pack} + S + \text{payments} + \text{support} + \text{ads} + \text{overhead})$, where S denotes shipping costs. Under the treatment, shipping becomes $S' = S(1 - 0.048)$, so the absolute profit change is $\Delta\pi = S - S' = 0.048S$. Expressed as a fraction of revenue, this is $\frac{\Delta\pi}{R} \approx 0.048 \times \frac{S}{R} \approx 0.048 \times 0.042 \approx 0.002$, corresponding to a 0.2% increase in overall profit.

Table 7: Effects of the proposed algorithm on product returns and incidents

	Logit	
	(1) returned	(2) incident
Proposed Algorithm	-0.008	-0.109***
	(0.038)	(0.036)
Intercept	-3.193***	-3.057***
	(0.027)	(0.025)
Baseline probability (control)	3.941	4.490
% Point effect (treatment vs control)	-0.030	-0.447
Relative % effect (treatment vs control)	-0.755	-9.952
Observations	75,285	75,285

Notes. The table reports treatment effects on the probability of a product being returned and the probability of an incident (e.g., product damaged, missing parts) occurring. Estimates are based on Equation 5 at the consumer-order level. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

We also find that consumers in the treatment condition did not provide significantly different ratings compared to those in the control group. Similarly, there were no statistically significant differences in the length of reviews, measured by the number of characters, words, or sentences (Table 8).

Table 8: Effects of the proposed algorithm on review ratings and review length

	Rating	*Review*		
	(1) 1-5	(2) # characters	(3) # words	(4) # sentences
Proposed Algorithm	-0.019	3.231	0.581	0.054
	(0.020)	(2.067)	(0.389)	(0.033)
Intercept	4.444***	66.094***	12.784***	1.986***
	(0.014)	(1.466)	(0.276)	(0.023)
Observations	11,757	11,586	11,586	11,586

Notes. The table reports treatment effects on the average star rating and the length of reviews written by consumers, measured in characters, words, and sentences. Data is at the consumer-ordered product-review level. Estimates are based on Equation 5. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Thus, the algorithm achieved meaningful cost improvements without degrading product quality or post-purchase customer satisfaction. Next, we examine the impact on the environmental emissions.

Environmental impact. This approach has significant potential environmental benefits. Under the proposed algorithm, the average shipping distance fell by 34 miles. This implies a per-order reduction by 3.8% relative to baseline.[17] Annually, the average retailer produces around 2–12 million tons of CO_2 emissions.[18] The 3.8% decrease means approximately 80-100K tons of CO_2 savings,

[17] U.S. Environmental Protection Agency (2024)

[18] Center for Sustainable Systems, University of Michigan (2024)

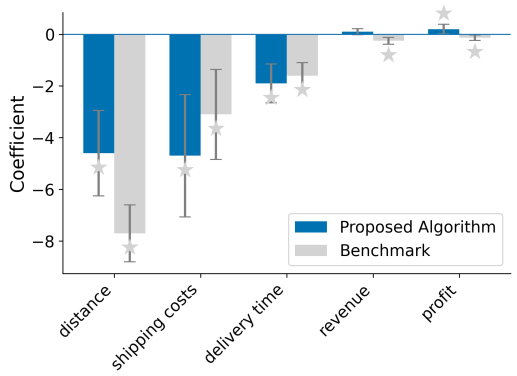
annually. This is equivalent to the 17-100K US household electricity use or 17-100K gasoline car usage.[19] For a retailer approximately similar to the one we partner with, the savings would be equivalent to 50,000 US household electricity use or 58,000 gasoline car usage. If these gains persist in the long-run equilibrium, they could have a highly positive impact on the environment. The key question, however, is why this approach works and under what conditions it succeeds or fails. In the next subsections, we examine these issues in detail.

6.4 Benchmarking the importance of bias-correction

We claim that our proposed ranking approach works precisely because we corrected for the competitive context effects, which were previously ignored. To show that, we benchmark our 'Proposed Algorithm' results against an alternative cost–conversion ranking experiment that was ran by the platform separately. Several weeks before our experiment, the platform tested an alternative: they ran an experiment for two weeks on 5.2 million consumers, where in the treatment group they used exact same scoring rule (Section 4) but used non-de-biased conversion estimates. Control group was the baseline conversion-maximizing ranking rule.

Detailed results for this biased benchmark are reported in the Appendix L; here we summarize the main comparison. As shown in Figure 10, the benchmark algorithm reduced shipping costs, shipping distances, and delivery times, much like the proposed approach. However, in contrast to our method, it also produced *significant decreases* in both revenue and profit.

Figure 10: Proposed bias-aware algorithm versus a cost–conversion benchmark without bias correction



Notes. This figure compares the proposed bias-aware algorithm with a cost–conversion benchmark that omits the neighborhood bias correction. Stars show the significant results.

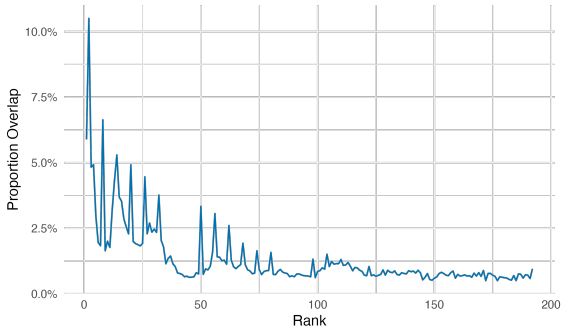
The performance gap stems from the benchmark's reliance on biased conversion estimates: without debiasing, the algorithm tended to over-promote low-cost products whose predicted

[19] See Center for Sustainable Systems, University of Michigan (2024)

conversion rates were inflated by favorable historical neighborhood placement, and to demote high-quality products whose performance was underestimated under the old policy. This misallocation reduced overall demand and eroded profitability despite achieving operational cost savings.

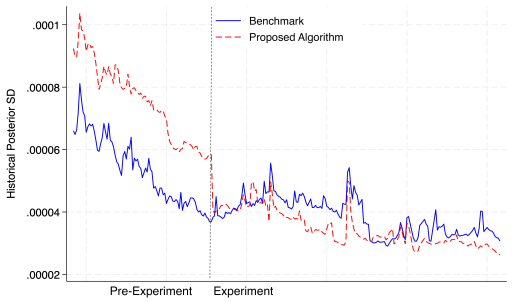
Next, we compare how often the products in the Proposed Algorithm were shown at the same ranks in the Treatment condition in biased algorithm, i.e., overlap rate between product-ranks. Figure 11 shows that the products the overlap rate between our algorithm and the biased version is very low (less than 10%), i.e., the composition of the products between our proposed algorithm and the biased one was very different.

Figure 11: Overlap rate



We next compare products shown under the Proposed algorithm with those in the Benchmark condition. Prior to the experiment (defined as the same number of weeks before each experiment), products later assigned to the Proposed algorithm exhibited significantly higher posterior uncertainty (Figure 12).

Figure 12: Posterior uncertainty of products in treatment and control groups over time



Notes. The figure plots average posterior standard deviations for products in the treatment and control groups before and during the experiments. y-ticks are rescaled for data confidentiality reasons.

This shows that the Proposed algorithm sampled and surfaced higher-uncertainty products more often than the Benchmark algorithm. Once the experiment began, uncertainty for these products declined sharply to levels equal to, or even lower than, those of Benchmark products.

This indicates that previously higher-uncertainty products received more precise signals under the de-biased algorithm, thereby altering the ranking distribution. Consequently, the Proposed algorithm achieved better results than the Benchmark.

Overall, the proposed algorithm was able to achieve lower costs without degrading consumer experience. The primary reason was that the new algorithm surfaced products that were de-prioritized by the biased algorithm.

6.5 Heterogeneity and managerial implications

Although the proposed bias-aware, cost–conversion ranking algorithm improved average outcomes, the gains were not uniform; we therefore investigate the conditions under which the algorithm performs well and when it does not.

Consumer demand flexibility. First, we investigate whether there are significant differences in consumers' readiness to substitute their choices in different categories. Running a consumer-category-level regression, we find that in the categories where there are more big-brand products, the orders went down by 0.3%. Profits were not impacted because the ordered products' shipping costs went down (Columns 1 and 3 of Table 9). Categories where brand is less prevalent, consumers' demand marginally increased and profits increased too (Columns 2 and 4 of Table 9).

Table 9: Consumer heterogeneity by category types

	log(orders)		*log(profit)*	
	(1)	(2)	(3)	(4)
	Branded	Not Branded	Branded	Not Branded
Proposed Algorithm	-0.003***	0.001***	-0.004	0.004***
	(0.001)	(0.000)	(0.004)	(0.001)
Intercept	0.027***	0.023***	0.085***	0.110***
	(0.001)	(0.000)	(0.003)	(0.001)
Observations	108,647	4,115,484	108,647	4,115,484

Notes. Data is at the category-customer level. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

We then check whether for the ordered products, there were differences in the distances traveled by the items across branded and non-branded categories. Among the ordered products, the branded ones traveled 28 miles or 5% more than the non-branded ones (Table 10). This means that the branded products were located in further away warehouses.

These results suggest that the platforms should be cautious: there are certain categories, where the proposed algorithm will work well because consumers' demand is flexible enough and consumers are ready to substitute across products. However, there are categories, where consumers value brand a lot and, therefore, are reluctant to substitute. The results from Table 10 suggest that the

Table 10: Distances traveled by branded and non-branded products

	OLS	
	(1) distance	(2) log(distance)
Proposed Algorithm	-48.640***	-0.071***
	(7.401)	(0.012)
Branded	33.068***	0.021*
	(7.523)	(0.012)
Proposed Algorithm× Branded	28.210***	0.050***
	(10.597)	(0.017)
Intercept	861.056***	6.288***
	(5.249)	(0.008)
Observations	75,285	75,285

Notes. Data is at the consumer-ordered product level. Estimates are based on Equation 5. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

platforms should also consider re-assorting products in a way that would put certain products closer to different consumers. Thus, re-ranking may help sometimes, but in some cases, the platforms may need to pay more attention to their supply chain: warehouse and assortment allocation.

Supplier network flexibility. Naturally, to achieve meaningful improvements in the shipping costs, the retailer should have sufficient room for improvement and flexible supply chain. We split product categories into two groups based on whether they have a small versus large share of unique products stocked in the centralized warehouses (more than 50% of all unique products in the category). Table 11 shows that the main effect actually comes from the categories with large assortment present in the centralized warehouses, which allows platform to source products from there. The orders and profit effect for large centralized share categories is +0.2% and +0.9%, respectively (Columns 1 and 3). The same metrics for the small centralized share categories are 0% and 0.3%, respectively (Columns 2 and 4).

Table 11: Supplier heterogeneity

	log(orders)		*log(profit)*	
	(1) Small Centralized Share	(2) Large Centralized Share	(3) Small Centralized Share	(4) Large Centralized Share
Proposed Algorithm	0.000***	0.002**	0.003***	0.009**
	(0.000)	(0.001)	(0.001)	(0.004)
Intercept	0.023***	0.033***	0.108***	0.145***
	(0.000)	(0.001)	(0.001)	(0.003)
Observations	3,982,767	241,364	3,982,767	241,364

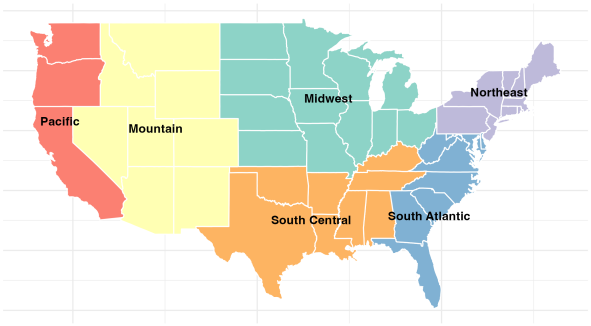
Notes. Data is at the category-customer level. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Thus, in order to be able to improve the shipping outcomes, the platform needs to have sufficient

flexibility in its supply chain, where they can quickly source products from centralized warehouses.

Regional heterogeneity. To explore regional variation in treatment effects, we use the U.S. Census Bureau shipping regions: Pacific, Mountain, South Atlantic, Midwest, South Central, and Northeast (Figure 13).[20]

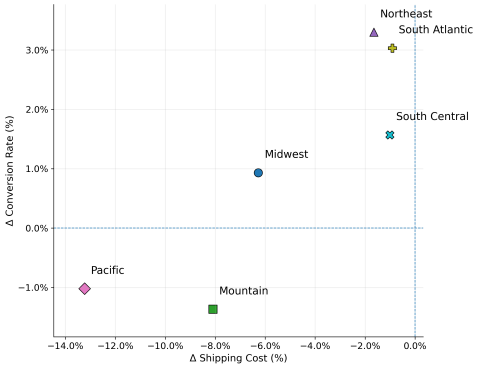
Figure 13: U.S. Census Bureau regional definitions used for heterogeneity analysis



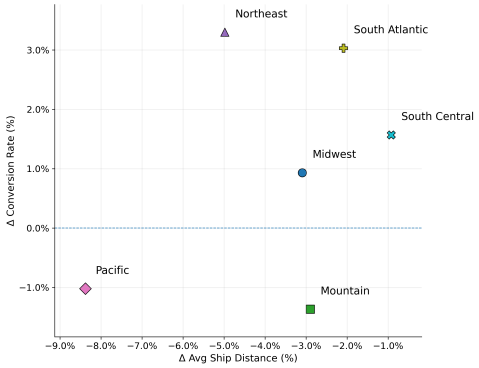
Notes. The map shows the six U.S. regions as defined by the U.S. Census Bureau: Pacific, Mountain, South Atlantic, Midwest, South Central, and Northeast.

Figure 14: Regional Heterogeneity in Cost Savings and Conversion Rate Changes

(a) Shipping Costs vs Conversion Changes



(b) Shipping Distance vs Conversion Changes



Notes. Panel (a) plots the change in average shipping costs (treatment vs. control) on the x-axis against the change in conversion rates on the y-axis for each U.S. Census Bureau region. Panel (b) replaces shipping cost changes with average shipping distance changes.

First, we compare the changes in the shipping distance, costs, and conversion rates across these regions. Figure 14a plots the change in shipping costs (treatment vs. control) against the change in conversion rates for each region. Four regions—Midwest, Northeast, South Central, and South Atlantic—achieved the desired "win–win" outcome: lower shipping costs and even higher conversion rates relative to the control. These gains coincided with reductions in average shipping distances

[20]The platform we work with maintains a different region mapping, however, due to data confidentiality agreement, we cannot report the results based on their region definitions.

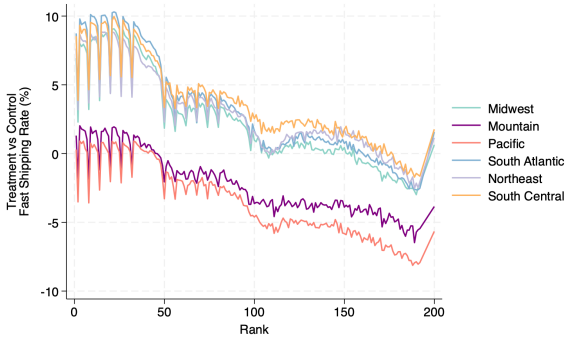
(Figure 14b).

In contrast, the Pacific and Mountain regions exhibited a less favorable pattern: although both shipping costs and average distances decreased, conversion rates also declined. This divergence in outcomes underscores that cost savings alone do not guarantee neutral or positive demand effects.

While in the Pacific the profit impact was minimal even after conversion drop, the Mountain Region experienced significant profit decrease (-2%).

Follow-up analysis revealed a key driver of the underperformance in Pacific and Mountain regions: changes in perceived delivery quality. As shown in Figure 15, the share of products carrying fast-shipping badges increased by 5–10 percentage points in treatment for all regions except Pacific and Mountain, where the share actually fell. Importantly, actual delivery times and incident rates in these regions did not change under treatment, suggesting that the decline in conversions was attributable to the loss of a visible quality signal rather than to any deterioration in service.

Figure 15: Fast Shipping badging by ranks



Notes. This figure shows the difference between treatment and control group badging (% points).

These findings point to a signal–substance tradeoff in multi objective optimization. Even when operational metrics—such as fulfillment cost, delivery time, and incident rates—improve, reducing visible indicators of speed or quality (e.g., fast-shipping badges) can dampen consumer demand. In our setting, this has an additional implication for modeling: when the algorithm changes badge allocation, it alters the composition of neighborhoods in the ranking. If badges become more concentrated in certain areas, the perceived quality gap between neighborhoods increases, and the performance of products shown in badge-rich versus badge-poor neighborhoods will diverge more sharply. This amplifies the role of neighborhood effects, making any bias from ignoring neighborhood composition effects even more pronounced. Correcting for neighborhood bias therefore becomes even more important, because the re-ranking is not only moving products but also restructuring the distribution of strong quality signals across the page.

In sum, the heterogeneity analyses reveal that the effectiveness of the proposed ranking algorithm depends on both demand- and supply-side conditions, as well as regional variation. The gains are strongest in categories where consumers are willing to substitute across products and where the platform has ample inventory in centralized warehouses. By contrast, brand-sensitive categories and regions that lost fast-shipping badges saw weaker or even negative effects, despite lower fulfillment costs. These findings highlight that bias-aware re-ranking can, in some settings, improve outcomes on its own, while in others it must be complemented by supply chain and badging strategies to secure robust gains in both consumer demand and platform profitability.

7 Conclusion

This paper shows that incorporating fulfillment-cost considerations into demand-side product-ranking algorithms can deliver meaningful operational and environmental gains without reducing demand or consumer satisfaction. We develop a Neighborhood-corrected Adaptive Doubly Robust (N-ADR) estimator that removes both exposure and neighborhood biases from historical conversion estimates. We integrate these bias-corrected estimates into a multi-objective ranking framework that balances conversion probability with fulfillment cost and quality constraints derived from operational metrics and embedding-based product attributes. Implemented within Thompson Sampling, this approach preserves the exploration–exploitation balance while producing conversion estimates that are unbiased and comparable across products.

Evidence from a large-scale field experiment on a major e-commerce platform confirms the effectiveness of the method. Relative to the platform's revenue-maximizing status quo, the proposed algorithm reduced page-load shipping costs by 4.2% and order-level shipping costs by 4.8%, with in-view costs declining by 4.1%. These savings are driven by shorter average shipping distances—34 miles fewer per order—and greater reliance on centralized fulfillment. These savings also translated into measurable environmental benefits. The reduction in shipping distances led to a 3.8% decrease in per-order emissions, equivalent to annualized savings of 50,000 U.S. households' electricity use or the emissions of 58,000 gasoline cars.

These cost and emissions reductions were achieved without degrading demand-side performance: orders and revenue remained statistically unchanged, profit increased by 0.2%, review ratings and review content were unaffected, and incident rates fell modestly. The contrast with a biased cost–conversion benchmark underscores the risks of using uncorrected conversion estimates in multi-objective optimization, as such biases can lead to systematic misallocation of rank positions. The bias corrections in our framework reverse these distortions, elevating genuinely comparable

low-cost products rather than those advantaged by historical exposure or neighborhood effects.

The findings suggest two broader lessons. First, significant efficiency gains can be realized on the demand side of online marketplaces through ranking interventions alone, without capital-intensive changes to fulfillment infrastructure. Second, algorithms that integrate cost signals while explicitly correcting for adaptive biases can deliver profitability and sustainability benefits at scale. Given the modest engineering requirements of our approach relative to full profit-maximizing re-optimization, such interventions are a practical option for platforms seeking near-term operational improvements.

Future research could extend this work by developing exact theoretical guarantees under neighborhood interference, examining longer-run deployments to capture equilibrium effects on seller behavior and assortment composition, and testing the approach across different marketplace structures. The present evidence indicates that demand-side cost-aware re-ranking—when paired with rigorous bias correction—is a scalable, low-friction lever for improving both economic and environmental performance in digital marketplaces.

References

Abdollahpouri, Himan, Masoud Mansoury, Robin Burke, and Bamshad Mobasher, "The unfairness of popularity bias in recommendation," *arXiv preprint arXiv:1907.13286*, 2019. (Cited on page(s) 12)

Akerlof, George A, "The market for "lemons": Quality uncertainty and the market mechanism," in "Uncertainty in economics," Elsevier, 1978, pp. 235–251. (Cited on page(s) 4)

Bell, Sean and Kavita Bala, "Learning visual similarity for product design with convolutional neural networks," *ACM transactions on graphics (TOG)*, 2015, *34* (4), 1–10. (Cited on page(s) 44, 45)

Bollinger, Bryan, Randi Kronthal-Sacco, and Levin Zhu, "Sustainable Product Demand and Price Elasticity," 2023. (Cited on page(s) 6)

Brecko, Kristina and Yewon Kim, "Voluntary Provision of Sustainability Claims: Evidence from Consumer Packaged Goods," Technical Report, Working paper 2023. (Cited on page(s) 6)

Center for Sustainable Systems, University of Michigan, "Carbon Footprint Factsheet," https://css.umich.edu/factsheets/carbon-footprint-factsheet 2024. Accessed: 2025-08-14. (Cited on page(s) 29, 30)

Cheng, Sikai, Amira Hijazi, Jeren Konak, Alan Erera, and Pascal Van Hentenryck, "SPOT: Spatio-Temporal Pattern Mining and Optimization for Load Consolidation in Freight Transportation Networks," *Transportation Science*, 2025, *59* (2), 345–360. (Cited on page(s) 6)

Cisternas, Gonzalo, Pradeep K. Chintagunta, and Maria Ana Vitorino, "Shelf design and product demand," *Quantitative Marketing and Economics*, 2020, *18* (4), 385–441. (Cited on page(s) 7)

Cristea, Anca, David Hummels, Laura Puzzello, and Misak Avetisyan, "Trade and the greenhouse gas emissions from international freight transport," *Journal of environmental economics and management*, 2013, *65* (1), 153–173. (Cited on page(s) 6)

Dimakopoulou, Maria, Zhimei Ren, and Zhengyuan Zhou, "Online multi-armed bandits with adaptive inference," *Advances in Neural Information Processing Systems*, 2021, *34*, 1939–1951. (Cited on page(s) 6, 16)

Dror, Moshe, Michael Ball, and Günther Zäpfel, "Shipment Consolidation: Who Pays for It and How Much?," *Management Science*, 2007, *53* (4), 632–645. (Cited on page(s) 6)

Dudík, Miroslav, John Langford, and Lihong Li, "Doubly Robust Policy Evaluation and Learning," in "Proceedings of the 31st International Conference on Machine Learning (ICML)" 2014, pp. 1097–1104. (Cited on page(s) 6)

eBay Inc., "Click Modeling for eCommerce," https://innovation.ebayinc.com/tech/engineering/click-modeling-for-ecommerce/ 2016. (Cited on page(s) 8)

Eckles, Dean, Brian Karrer, and Johan Ugander, "Design and analysis of experiments in networks: Reducing bias from interference," *Journal of Causal Inference*, 2017, *5* (1), 20150021. (Cited on page(s) 7)

Goli, Ali, Anja Lambrecht, and Hema Yoganarasimhan, "A bias correction approach for interference in ranking experiments," *Marketing Science*, 2024, *43* (3), 590–614. (Cited on page(s) 7)

Graepel, Thore, Joaquín Quiñonero Candela, Thomas Borchert, and Ralf Herbrich, "Web-Scale Bayesian Click-Through Rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine," in "Proceedings of the 27th International Conference on Machine Learning (ICML)" 2010. Invited Applications Track. (Cited on page(s) 8)

Greminger, Rafael P, "Optimal search and discovery," *Management Science*, 2022, *68* (5), 3904–3924. (Cited on page(s) 8)

Hadad, Vitor, David A. Hirshberg, Ruohan Zhan, Stefan Wager, and Susan Athey, "Confidence intervals for policy evaluation in adaptive experiments," *Proceedings of the National Academy of Sciences*, 2021, *118* (15). (Cited on page(s) 6)

Hitsch, Günter J, Sanjog Misra, and Walter W Zhang, "Heterogeneous treatment effects and optimal targeting policy evaluation," *Quantitative Marketing and Economics*, 2024, *22* (2), 115–168. (Cited on page(s) 6)

Holtz, David, Felipe Lobel, Ruben Lobel, Inessa Liskovich, and Sinan Aral, "Reducing interference bias in online marketplace experiments using cluster randomization: Evidence from a pricing meta-experiment on airbnb," *Management Science*, 2025, *71* (1), 390–406. (Cited on page(s) 7)

Hortaçsu, Ali, Olivia R Natan, Hayden Parsley, Timothy Schwieg, and Kevin R Williams, "Organizational structure and pricing: Evidence from a large us airline," *The Quarterly Journal of Economics*, 2024, *139* (2), 1149–1199. (Cited on page(s) 1)

Horvitz, Daniel G and Donovan J Thompson, "A generalization of sampling without replacement from a finite universe," *Journal of the American statistical Association*, 1952, *47* (260), 663–685. (Cited on page(s) 2)

Hübner, Alexander H. and Hendrik Kühn, "Integrated assortment, shelf space, and replenishment planning in retailing," *Journal of Retailing*, 2024. (Cited on page(s) 7)

Imbens, Guido W and Donald B Rubin, *Causal inference in statistics, social, and biomedical sciences*, Cambridge university press, 2015. (Cited on page(s) 2)

Jia, Chao, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," in "International Conference on Machine Learning" PMLR 2021, pp. 4904–4916. (Cited on page(s) 44)

Jiang, Nan and Lihong Li, "Doubly Robust Off-policy Value Evaluation for Reinforcement Learning," in "Proceedings of the 33rd International Conference on Machine Learning (ICML)" 2016, pp. 652–661. (Cited on page(s) 6)

Juan, Angel A, Javier Faulin, Elena Pérez-Bernabeu, and Nicolas Jozefowiez, "Horizontal cooperation in vehicle routing problems with backhauling and environmental criteria," *Procedia-Social and Behavioral Sciences*, 2014, *111*, 1133–1141. (Cited on page(s) 6)

Kang, Peng, Guanghan Song, Ming Xu, Travis R Miller, Haikun Wang, Hui Zhang, Gang Liu, Ya Zhou, Junshu Ren, Ruoyu Zhong et al., "Low-carbon pathways for the booming express delivery sector in China," *Nature Communications*, 2021, *12* (1), 450. (Cited on page(s) 6)

Khan, Samir, Martin Saveki, and Johan Ugander, "Off-policy evaluation beyond overlap: Sharp partial identification under smoothness," in "in" International Conference on Machine Learning (ICML) 2024. (Cited on page(s) 2)

Kiyohara, Haruka, Yuta Saito, Tatsuya Matsuhiro, Yusuke Narita, Nobuyuki Shimizu, and Yasuo Yamamoto, "Doubly Robust Off-Policy Evaluation for Ranking Policies under the Cascade Behavior Model," in "Proceedings of the 15th ACM International Conference on Web Search and Data Mining (WSDM)" 2022, pp. 406–414. (Cited on page(s) 6)

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, 2017, *60* (6), 84–90. (Cited on page(s) 44)

Li, Haoxuan, Chunyuan Zheng, Sihao Ding, Peng Wu, Zhi Geng, Fuli Feng, and Xiangnan He, "Be aware of the neighborhood effect: Modeling selection bias under interference," *arXiv preprint arXiv:2404.19620*, 2024. (Cited on page(s) 7)

Luedtke, Alex R and Mark J van der Laan, "Statistical inference for the mean outcome under a possibly non-unique optimal treatment strategy," *Annals of Statistics*, 2016, *44* (2), 713–742. (Cited on page(s) 3, 6, 14)

Madeka, Dhruv, Kari Torkkola, Carson Eisenach, Anna Luo, Dean P Foster, and Sham M Kakade, "Deep inventory management," *arXiv preprint arXiv:2210.03137*, 2022. (Cited on page(s) 1)

Proserpio, Davide, Ali Goli, Tyler Mangini, Ken Lau, and Daniela Yu, "The impact of sustainability programs on consumer purchase behavior: Evidence from Amazon," *International Journal of Research in Marketing*, 2025. (Cited on page(s) 6)

Rendle, Steffen, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in "Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)" 2009. (Cited on page(s) 8)

Rooderkerk, Robert P, Harald J Van Heerde, and Tammo HA Bijmolt, "Optimizing retail assortments," *Marketing Science*, 2013, *32* (5), 699–715. (Cited on page(s) 7)

Rusu, Andrei A, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016. (Cited on page(s) 44)

Saito, Yuta, "Doubly Robust Estimator for Ranking Metrics with Post-Click Conversions," in "Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)" 2020, pp. 92–101. (Cited on page(s) 6)

Simester, Duncan, Artem Timoshenko, and Spyros I Zoumpoulis, "Targeting prospective customers: Robustness of machine-learning methods to typical data challenges," *Management Science*, 2020, *66* (6), 2495–2522. (Cited on page(s) 6)

Sonntag, Danja R., Albert Schrotenboer, and Gudrun P. Kiesmüller, "Stochastic Inventory Routing with Time-Based Shipment Consolidation," *European Journal of Operational Research*, 2023, *306* (3), 1186–1201. (Cited on page(s) 6)

Tran, Nguyen Khoi and Jasmine Siu Lee Lam, "Effects of container ship speed on CO2 emission, cargo lead time and supply chain costs," *Research in Transportation Business & Management*, 2022, *43*, 100723. (Cited on page(s) 6)

Ursu, Raluca M, "The power of rankings: Quantifying the effect of rankings on online consumer search and purchase decisions," *Marketing Science*, 2018, *37* (4), 530–552. (Cited on page(s) 12)

U.S. Environmental Protection Agency, "Greenhouse Gas Equivalencies Calculator," https://www.epa.gov/energy/greenhouse-gas-equivalencies-calculator 2024. Accessed: 2025-08-14. (Cited on page(s) 29)

Wang, Yuyan, Long Tao, and Xian Xing Zhang, "Recommending for a multi-sided marketplace: A multi-objective hierarchical approach," *Marketing Science*, 2025, *44* (1), 1–29. (Cited on page(s) 7, 19)

Wayfair, "Bayesian Product Ranking at Wayfair," https://www.aboutwayfair.com/tech-innovation/bayesian-product-ranking-at-wayfair 2022. (Cited on page(s) 8)

Wei, Lai, Roman Kapuscinski, and Stefanus Jasin, "Shipping Consolidation Across Two Warehouses with Delivery Deadline and Expedited Options for E-commerce and Omni-channel Retailers," *Manufacturing & Service Operations Management*, 2021, *23* (6), 1634–1650. (Cited on page(s) 6)

Xu, Zhou, Feng Li, and Zhi-Long Chen, "Shipment Consolidation with Multiple Shipping Methods Under Nonlinear Cost Structures," *Management Science*, 2024, *70* (5), 2823–2841. (Cited on page(s) 6)

Yang, Yinfei, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil, "Multilingual Universal Sentence Encoder for Semantic Retrieval," 2019. (Cited on page(s) 46)

Online Appendix

Table of Contents

A Image Embeddings **43**

B Textual Embeddings **46**

C Addressing potential SUTVA violation concerns **48**

D Additional Experimental Results **50**

E Simulation Design **52**

F Empirical evidence of position and exposure biases **54**

G Additional Descriptive Statistics **57**

G.1 Differences in products by fulfillment type 57

G.2 *t*-tests of product quality in treatment and control 58

G.3 Examples of posterior score evolutions 60

H Approximate Theoretical Guarantees (with Neighborhood Interference) **61**

I Additional details on N-ADR **65**

J Implementing Post-hoc Debiasing on the Illustrative Example **67**

K Randomization Checks **69**

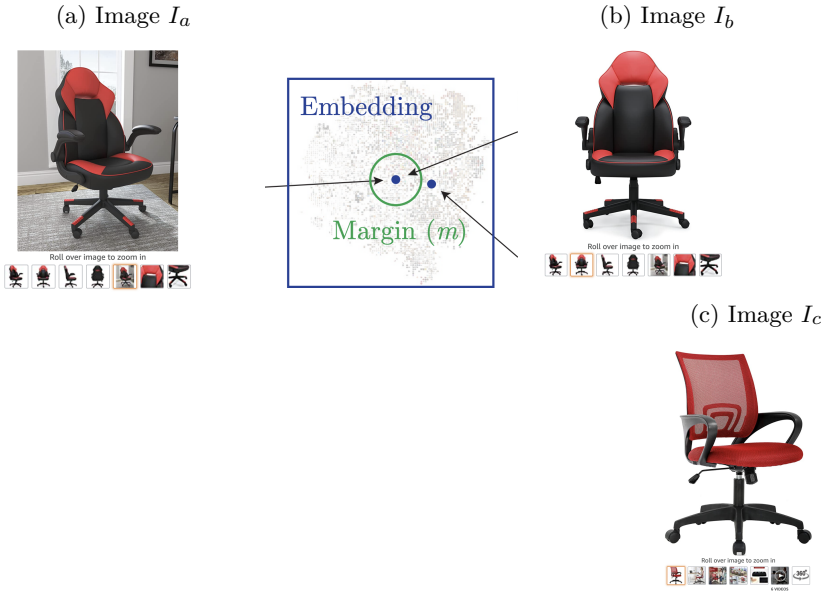
L Results of the experiment with no de-biasing **70**

A Image Embeddings

We use Deep Learning, specifically Convolutional Neural Networks (CNNs), to model image similarity. The objective is to project each image into a finite-dimensional space such that similar images are located close together, while dissimilar images are farther apart.

To illustrate, consider three images, I_a, I_b, and I_c. Images I_a and I_b depict the same object (Panels (a) and (b) of Figure A1), whereas I_b and I_c correspond to entirely different objects (Panels (b) and (c) of Figure A1). In the embedding space, I_a and I_b should be positioned near each other, while I_c should lie at a greater distance.

Figure A1: Illustration of the Image Embedding algorithm



Notes. This figure illustrates the idea behind the image embedding algorithm. There are three images where image in Panel (a) and Panel (b) are highly similar because they feature same product from different angles. Third image in Panel (c) is also a red chair but differs in style. The goal of the algorithm is to project images into some finite-dimensional space where products from Panels (a) and (b) are closely located in that embedding space, and product in Panel (c) is located further away from them.

At a high level, the algorithm operates as follows. The inputs are 10 million images and their ground truth labels, i.e., the y-variables the model aims to predict. The labels are binary: $y = 1$ if the images belong to the same product,[21] and $y = 0$ otherwise. As illustrated above, the algorithm should assign high y values to distinct images of the same product, and low values otherwise.

During training, the model learns a 512×1 vector representation for each image, referred to as the *image embedding*. We then compute distances (or similarities) between products using standard

[21] A product can have multiple images on the website, e.g., different angles, environment.

metrics, such as cosine similarity. Finally, we identify closely matched products based on the level of pairwise cosine similarity.

Next, we provide a more formal description of the trained Convolutional Neural Network (CNN) for image similarity (see Rusu et al. (2016) and Bell and Bala (2015) for more details). A convolutional neural network is a function f that maps each image I into an embedding position x, given parameters θ: $f_\theta : we \rightarrow x$. One could think of embedding as a numerical vector representation of the image. The goal of CNN is to solve for the parameter vector θ such that the produced embeddings x are such that similar images are placed nearby, whereas dissimilar images are more distant.

In the example above (Figure A1), images are mapped into the embedding vectors x_a, x_b, x_c. If the model is trained well and produces 'good' embeddings, then the correct (positive) pair of images represented by vectors x_a and x_b should be closely located in vector space. Meanwhile the incorrect (negative) pair of dissimilar images represented by vectors x_b and x_c should be more distant in vector space, i.e. separated by at least a margin m in the embedding space.

The formal objective function for this type of task is called contrastive loss and was proposed by Jia et al. (2021). we minimize the loss function defined as

$$L(\theta) = \sum_{x_a, x_b} \underbrace{L_b(x_a, x_b)}_{\substack{\text{Penalty if similar images} \\ \text{are far away}}} + \sum_{x_b, x_c} \underbrace{L_c(x_b, x_c)}_{\substack{\text{Penalty if dissimilar images} \\ \text{are nearby}}} \tag{7}$$

where $L_b(x_a, x_b) = ||x_a - x_b||_2^2$ and $L_c(x_b, x_c) = max\{0, m^2 - ||x_b - x_c||_2^2\}$. The objective (loss) function consists of two parts: $L_b(x_a, x_b)$ penalizes a positive pair (x_a, x_b) if the embeddings of the images of the same product are too far apart, and $L_c(x_b, x_c)$ penalizes a negative pair (x_b, x_c) if the images of two different items are closer than the margin m.

We minimize the objective function in Equation 7 with respect to parameters θ using stochastic gradient descent with momentum (Krizhevsky et al., 2017). Parameters θ govern how well the model learns the representation of the images in a vector space.

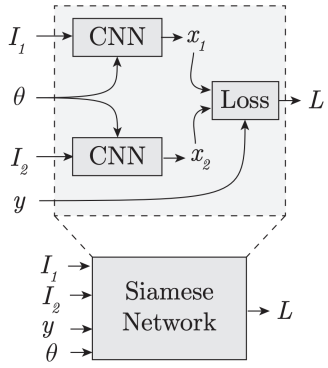
$$v^{(t+1)} \leftarrow \mu \cdot v^{(t)} - \alpha \cdot \frac{\partial L}{\partial \theta}(\theta^{(t)}) \tag{8}$$

$$\theta^{(t+1)} \leftarrow \theta^{(t)} + v^{(t+1)} \tag{9}$$

where θ is the parameters to be estimated, v is the momentum sequence, $\mu \in [0, 1)$ is the momentum and $\alpha \in [0, \infty)$ is the learning rate. To efficiently compute the loss function L and the gradient $\frac{\partial L}{\partial \theta}$, we follow Hadsell et al. (2006). Their approach is to construct a siamese network, which is the two copies of the CNN that share the same set of parameters θ. The network takes as

an input two images I_1, I_2, θ and the indicator variable for whether the images are a positive pair ($y = 1$) or a negative pair ($y = 0$), and outputs the loss function L (Figure A2). The computational efficiency arises from the two copied networks.

Figure A2: Siamese network



Notes. This figure illustrates the architecture of the Deep Neural Network that we used to create image embeddings. Source: Bell and Bala (2015).

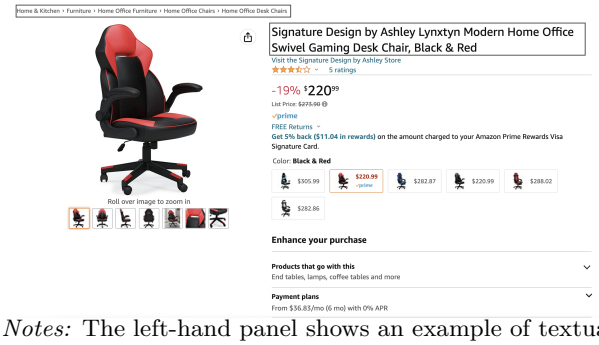
As was mentioned above, we use 10 million images to train the network. The output of the algorithm is the 512×1 vector representation of each image, which we use to calculate pairwise cosine similarities between products.

B Textual Embeddings

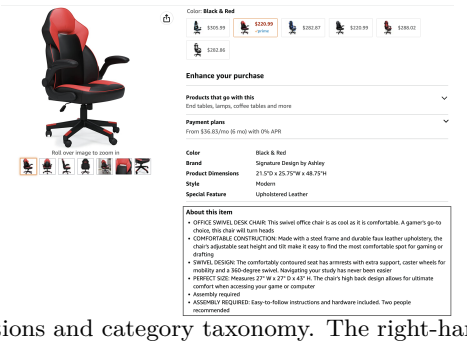
We use the textual data of the company to create embeddings. The company has access to the category and product names (Panel (a) of Figure B3) and detailed itemized product descriptions (Panel (b) of Figure B3). we use this information to create a lower-dimensional vector of the product information. The goal is to represent the unstructured textual information in an $N \times 1$ vector form to be able to calculate the cosine similarities between different products.

Figure B3: Product descriptions

(a) Category and product names



(b) Itemized product description



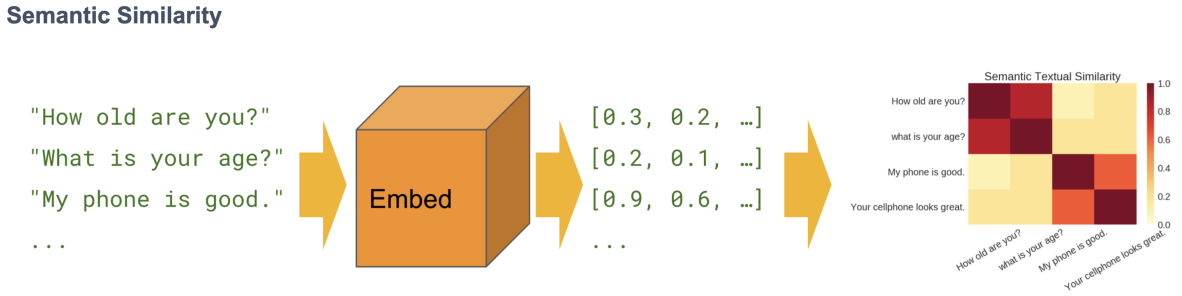
Notes: The left-hand panel shows an example of textual descriptions and category taxonomy. The right-hand side boxed part shows an example of the detailed product descriptions.

To achieve that, we embed the raw text description into a single semantic space using Universal Sentence Encoder developed by Google (Yang et al. (2019). The encoder model embeds raw English text into the high-dimensional vector (512×1) that can be then used for text classification, semantic similarity, clustering and other natural language tasks. The benefit of using the Universal Sentence Encoder model compared to wide-spread word embeddings, is that encoders model the meaning of the word sequences than just individual words and are trained on multiple tasks. Figure B4 shows an example of how the semantic similarity task is performed: the goal is to understand how similar two pieces of text are. In this example, the questions "How old are you?" and "What is your age?" have high similarity, while sentence "our phone is good" is dissimilar. Given good performance of this model that was used in multiple applications, we proceed using it to construct the cosine similarities of the products.

Figure B5 shows an example of how the textual embedding works. The input to the model are the product descriptions. The embedding model outputs 512×1 vector for each product's description. This allows us to calculate the cosine similarities between different products.

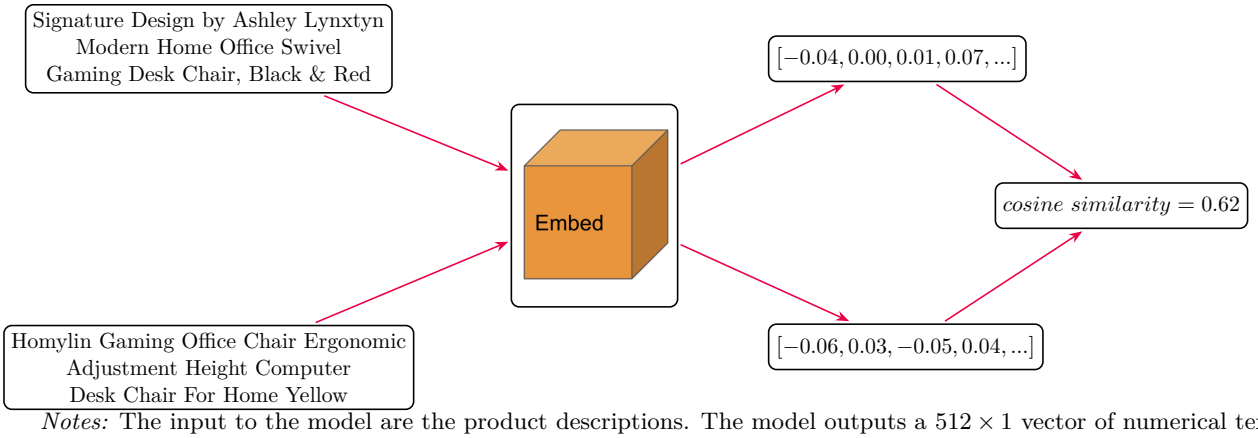
Figure B6 shows the corresponding pictures of the two products. Indeed, chairs do look somewhat similar but differ in color and style a bit, which may justify the cosine similarity of 0.62.

Figure B4: Example of Universal Sentence Encoder



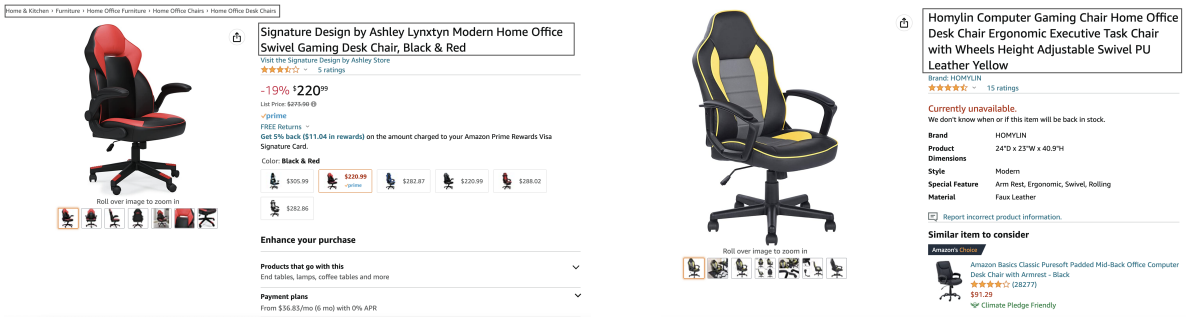
Source: Google Universal Sentence Encoder

Figure B5: Example of how textual embeddings work



Notes: The input to the model are the product descriptions. The model outputs a 512×1 vector of numerical text representation. This allows to calculate the cosine similarity between these vectors.

Figure B6: Example of textual embedding model



C Addressing potential SUTVA violation concerns

Rank changes. To check potential SUTVA violations in ranks, we run the following specification:

$$\log(\text{TreatmentRank}_{it}) = \alpha + \beta_1 \cdot \log(\text{TreatmentScore}_{it}) + \beta_2 \cdot \log(\text{ControlScore}_{it}) + \varepsilon_{it} \quad (10)$$

The dependent variable $\log(\text{TreatmentRank}_{it})$ denotes the natural logarithm of the rank of product i at time t within the treatment group. The variable $\log(\text{TreatmentScore}_{it})$ is the logarithm of the product's own score, while $\log(\text{ControlScore}_{it})$ captures the logarithm of the average score of control group products at the same time. The coefficient β_1 captures how sensitive a product's rank is to changes in its own score, controlling for the performance of control products. The term α denotes the intercept, and ε_{it} is the error term.

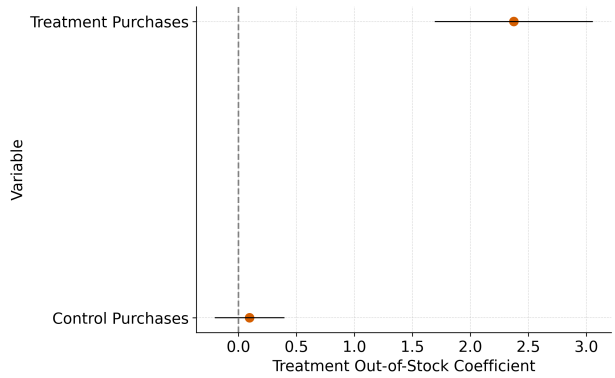
Table C1: Rank Score spillover test: treatment vs. control ranks

	Log(Treatment Rank)
Intercept	6.891***
	(0.001)
log(Treatment Score)	-0.701***
	(0.003)
Log(Control Score)	-0.001
	(0.003)
Observations	2,645,705

Notes. This table reports OLS estimates of the relationship between the treatment and control scores and the log treatment ranks. Standard errors are reported in parentheses. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Out-of-stock changes. This figure plots coefficients from an OLS regression of the out-of-stock rate on purchase counts in control and treatment groups. Points show estimates; horizontal bars denote 95% confidence intervals. The analysis tests for potential interference between units by examining whether purchases in one group predict stockouts in the other.

Figure C7: Out-of-stock spillover test: treatment vs. control purchases



Notes: This figure plots coefficients from an OLS regression of the out-of-stock rate on purchase counts in control and treatment groups. Points show estimates; horizontal bars denote 95% confidence intervals.

D Additional Experimental Results

Table D2: Experimental impact on returns by refinement and personalization

	Logit: returns			
	(1) filtered=0, personalization=0	(2) filtered=0, personalization=1	(3) filtered=1, personalization=1	(4) filtered=1, personalization=0
Proposed Algorithm	-0.137*	-0.045	0.056	0.038
	(0.082)	(0.092)	(0.068)	(0.066)
Intercept	-3.122***	-3.170***	-3.252***	-3.191***
	(0.056)	(0.065)	(0.049)	(0.047)
Observations	15,722	12,452	23,403	23,957

Notes. This table reports the output from the estimation of equation 5 on **purchasing experimental consumers**. Columns (1)-(4) use Logit specification results. Robust standard errors in parentheses. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Table D3: Experimental impact on incidents by refinement and personalization

	Logit: incidents			
	(1) filtered=0, personalization=0	(2) filtered=0, personalization=1	(3) filtered=1, personalization=1	(4) filtered=1, personalization=0
Proposed Algorithm	-0.297***	-0.075	-0.089	-0.050
	(0.086)	(0.093)	(0.063)	(0.061)
Intercept	-3.144***	-3.183***	-2.995***	-2.993***
	(0.057)	(0.065)	(0.044)	(0.043)
Observations	15,722	12,452	23,403	23,957

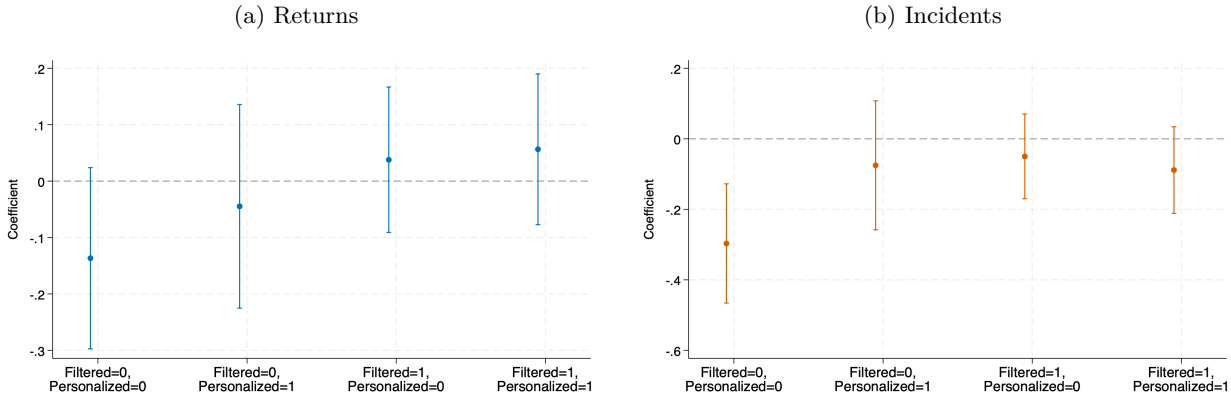
Notes. This table reports the output from the estimation of equation 5 on **purchasing experimental consumers**. Columns (1)-(4) use Logit specification results. Robust standard errors in parentheses. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Table D4: Effects of the proposed algorithm on review ratings and length (customer-level)

	Rating	*Review*		
	(1) 1-5	(2) # characters	(3) # words	(4) # sentences
Proposed Algorithm	-0.021	2.577	0.452	0.050
	(0.027)	(2.412)	(0.454)	(0.039)
Intercept	4.390***	69.120***	13.331***	2.024***
	(0.019)	(1.711)	(0.322)	(0.027)
Observations	6,726	6,672	6,672	6,672

Notes. The table reports treatment effects on the average star rating and the length of reviews written by consumers, measured in characters, words, and sentences. Data is at the customer level. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Figure D8: Experimental impact on returns and incidents by refinement and personalization



Notes. This figure reports the output from the estimation of equation 5 on **purchasing experimental consumers** on returns and incidents by refinement and personalization. Robust standard errors in parentheses. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Table D5: Effects of the proposed algorithm on delivery times by Census regions

	log(delivery time)					
	(1) Midwest	(2) Mountain	(3) Northeast	(4) Pacific	(5) South Atlantic	(6) South Central
Proposed Algorithm	-0.009	-0.006	-0.026***	-0.021**	-0.016**	-0.017*
	(0.009)	(0.012)	(0.008)	(0.009)	(0.007)	(0.009)
Intercept	1.640***	1.754***	1.570***	1.696***	1.611***	1.653***
	(0.006)	(0.008)	(0.006)	(0.007)	(0.005)	(0.006)
Observations	13,059	5,594	16,980	11,139	17,298	10,467

Notes. The table reports treatment effects on the delivery times by Census regions. Data is at the customer-ordered product level. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

E Simulation Design

We describe a concrete simulation procedure to evaluate average reward and regret for Thompson Sampling (TS) ranking with Adaptive Doubly Robust (ADR) and Neighborhood Adaptive Doubly Robust (N-ADR) estimators.

1) Ground-Truth Environment.

- **Products:** Let J denote the number of products (e.g., 200–2000), K the number of display slots (e.g., 6–12), and T the time horizon (e.g., 50,000–500,000 impressions). Draw product embeddings $e_j \in \mathbb{R}^d$ (e.g., $d = 64$) from a spherical Gaussian and L2-normalize. Draw product covariates x_j^{prod} and time-varying contexts X_{jt} independently for each impression.
- **Position and neighborhood effects:** Assign position biases $\alpha_1 > \cdots > \alpha_K$ (e.g., $\alpha_k = \alpha_1 - 0.25(k-1)$ with $\alpha_1 \in [1.0, 1.5]$). Define a neighborhood aggregator:

$$g(j, S_{jt}, k) = \sum_{\ell \neq k} w_{\ell k} \Big(\eta_c \cos(e_j, e_{a_{\ell t}}) + \eta_m \, m(j, a_{\ell t}) \Big),$$

  with $w_{\ell k} = \gamma^{|\ell - k|}$ (e.g., $\gamma = 0.7$), $\eta_c < 0$ for cannibalization, $\eta_m > 0$ for complementarity, and $m(\cdot, \cdot) \in \{0, 1\}$ a pre-sampled complementarity tag.
- **True outcome model:**

$$\Pr(y_{jt} = 1 \mid j@k, S_{jt}) = \sigma\Big(\beta^\top X_{jt} + \alpha_k + g(j, S_{jt}, k) \Big).$$

  Optionally introduce nonstationarity via $\beta_t = \beta + \delta \, t/T$.
- **Target policy distribution:** Define $q_0(S \mid X, j@k)$ as the uniform-without-replacement neighborhood distribution conditional on placing j in slot k. Compute oracle estimands

$$r^\star_{j,k} = \mathbb{E}_X \, \mathbb{E}_{S \sim q_0(\cdot \mid X, j@k)} \left[\Pr(y_{jt} = 1 \mid j@k, S, X) \right],$$

  and $r^\star_j = \sum_{k=1}^K w_k r^\star_{j,k}$ via large Monte Carlo.

2) Policies Compared.

1. **TS-Logistic (naïve):** Bayesian logistic regression without neighborhood features.
2. **TS+ADR:** TS using ADR pseudo-outcomes and exposure-bias correction only.
3. **TS+N-ADR:** TS with both exposure- and neighborhood-bias correction, using density ratios $w_t(S \mid X, j@k)$.

4. **Oracle:** For each slot k, ranks products by $r^{\star}_{j,k}$ and fills the slate without replacement (greedy slot-wise optimizer).

3) Logging of Propensities and Neighborhood Densities. For each impression with context $\{X_{jt}\}_{j=1}^{J}$:

- Estimate pair propensities $\hat{\pi}_t(j,k)$ by resimulating the TS decision R times (e.g., $R = 50$) *holding the context fixed* and resampling $\tilde{\beta}$;

$$\hat{\pi}_t(j,k) = \frac{1}{R}\sum_{r=1}^{R} \mathbf{1}\{j \text{ placed at slot } k\}.$$

- Collect realized neighborhoods S whenever $j@k$ occurs to estimate $\tilde{q}_t(S \mid X, j@k)$.
- Form density ratios

$$w_t(S \mid X, j@k) = \frac{q_0(S \mid X, j@k)}{\tilde{q}_t(S \mid X, j@k)},$$

and apply clipping $w_t^{\in}[w_{\min}, w_{\max}]$ and $\hat{\pi}_t^{\geq}\varepsilon$ for stability.

4) Simulation Loop. For $t = 1, \ldots, T$:

1. Sample $\{X_{jt}\}_{j=1}^{J}$.
2. The policy selects a slate A_t.
3. For each $(j,k) \in A_t$, draw $y_{jt} \sim$ Bernoulli(true model).
4. Log $(Z_{jt}(k), y_{jt}, X_{jt}, S_{jt})$, $\hat{\pi}_t(j,k)$, and $w_t(S_{jt} \mid X_{jt}, j@k)$.
5. Update:

$$\Gamma_t^{(j,k)} = m_{t-1}^{(j,k)}(X_{jt}, S_{jt}) + \frac{Z_{jt}(k)}{\hat{\pi}_t^{(}j,k)}\Big[y_{jt} - m_{t-1}^{(j,k)}(X_{jt}, S_{jt})\Big] \times \begin{cases} w_t^{(}S_{jt} \mid X_{jt}, j@k), & \text{N-ADR}, \\ 1, & \text{ADR}. \end{cases}$$

6. Update ADR heads $(\hat{\mu}_t^{(j,k)}, \hat{\sigma}_t^{2,(j,k)})$ via $\sqrt{\hat{\pi}_t}$-weighting.

5) Metrics.

- **Average reward:** $\overline{R}_{1:t} = \frac{1}{t}\sum_{s \leq t} R_s$, where $R_t = \sum_{(j,k) \in A_t} y_{jt}$.
- **Regret:** Instantaneous $\Delta_t = R_t^{\star} - R_t$, cumulative Regret$(t) = \sum_{s \leq t} \Delta_s$.
- **Estimator quality:** Bias, RMSE, and CI coverage of $\hat{r}_{j,k}$ versus $r^{\star}_{j,k}$ for Naïve, IPS, DR, ADR, and N-ADR.

6) Default Parameters.

$$J = 500, \quad K = 8, \quad T = 20{,}000, \quad R = 50, \quad d = 64,$$

$\alpha_1 = 1.25$ decreasing to $\alpha_K = -0.5$, $\eta_c = -0.8$, $\eta_m = 0.6$, $\gamma = 0.7$, prior $\mathcal{N}(\mathbf{0}, 10I)$, $\varepsilon = 10^{-3}$, $w_{\min} = 0.05$, $w_{\max} = 20$.

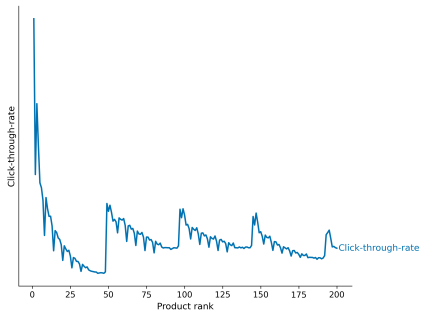
F Empirical evidence of position and exposure biases

As we mentioned above, we partnered with an anonymous online retailer for this paper. As many other retailers, this retailer uses the Bayesian algorithm with Thompson sampling, similar to the one described in Section 3. We provide more details on the institutional context in Section 6, and here just document the existence of the biases in the ranking system.

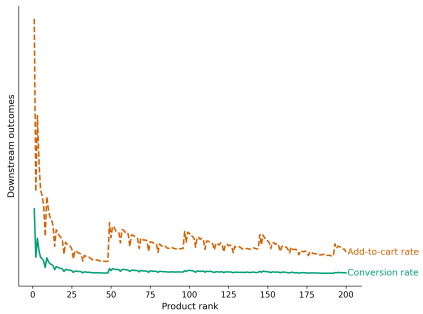
Position Effects. We use the platform's holdout sample, where random rankings were shown, and plot the average click-through, add-to-cart, and conversion rates by rank position. Figure F9 shows that click-through rates are higher at the top of the page and decline further down, with jumps appearing at the top of each subsequent page (each page has 50 products). The same pattern holds for add-to-cart and conversion rates. Thus, even under random rankings, positions impact consumer choices.

Figure F9: Descriptive Ranking effect

(a) Click-through-rate by product rank



(b) Add-to-cart and conversion rate by product rank



Notes. This figure shows the click-through-rate, add-to-cart, and conversion rates per product rank.

In our empirical application, we are less concerned about position effects because the platform has long been aware of the issue and includes product rankings directly in the outcome modeling.[22]

Exposure Bias. To investigate whether Thompson Sampling-based ranking algorithm has exposure biases, we proceed as follows: first, we split products into bins based on their previous month's

[22]The platform is long-aware of the issue and applies additional bias-correction terms to account for the position bias.

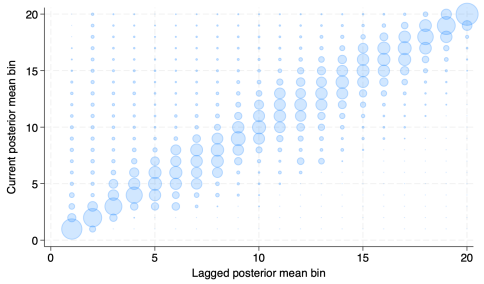
posterior scores. Next, we calculate their current posteriors and again bin the products into 20 bins, where the 1st bin contains the highest scores, and so on.

Figures F10a and F10b show the relationship between 1-month lagged posterior means and standard deviations, and the subsequent month's posterior means and standard deviations, respectively. The size of the bubbles represents the share of products in the corresponding cells.

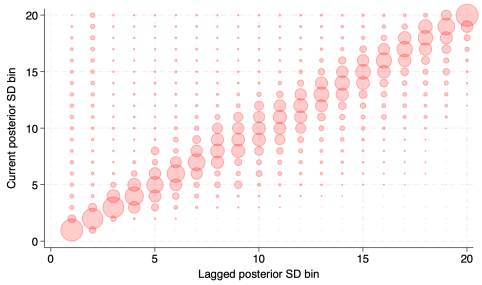
Even with the exploration provided by the Thompson sampling algorithm, it is clear that a product is very likely to remain in the same bin as time progresses. This indicates that products in lower bins are less likely to move up to higher bins, thereby revealing the presence of exposure bias.

Figure F10: Descriptive evidence of exposure biases

(a) Posterior Mean



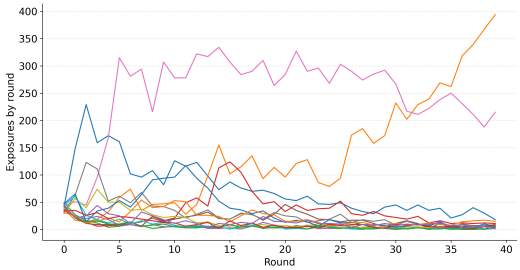
(b) Posterior Standard Deviation



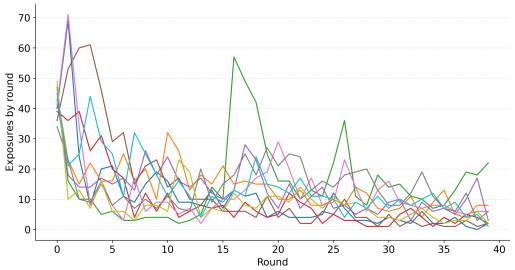
Notes. Figures show the relationship between 1-month lagged posterior statistics (means in panel a; standard deviations in panel b) and the subsequent month's values. Bubble sizes indicate the share of products in each cell. Products tend to remain in the same bins over time, illustrating path dependence and the presence of exposure bias.

Figure F11: Simulations of Exposure

(a) Random Sample



(b) Starved products



Notes. Figures show simulated product exposures over 40 update rounds using the platform's actual algorithm. Panel (a) presents a random sample of products. Panel (b) shows "starved" products, which receive minimal exposure during the simulation.

To further demonstrate the presence of biases, we take the platform's actual algorithm and simulate its behavior for different types of products. The figures plot exposure per round over 40 simulated update rounds, with the y-axis showing the number of exposures for each product. Figure F11a presents results for a random sample of products: some receive high exposure early and maintain it, others gain exposure later, and some lose it over time. Figure F11b shows results for

"starved" products—those receiving minimal exposure during the simulation. Once disadvantaged, these products are highly unlikely, with few exceptions, to appear on the website.

G Additional Descriptive Statistics

G.1 Differences in products by fulfillment type

Table G6: Summary Statistics by fulfillment type

	Min	Mean	Median	Max	SD
	(1)	(2)	(3)	(4)	(5)
Panel A: Rating					
Dropship	1.00	4.39	4.63	5.00	0.81
Centralized	1.00	4.48	4.65	5.00	0.67
Panel B: Review Count					
Dropship	0.00	45.27	3.00	57081.00	530.58
Centralized	0.00	70.46	5.00	107038.00	706.96
Panel C: Return Share					
Dropship	0.00	13.56	1.77	100.00	25.40
Centralized	0.00	7.66	1.75	100.00	16.79
Panel D: Incident Rate					
Dropship	0.00	0.34	0.00	1.00	0.48
Centralized	0.00	0.34	0.00	1.00	0.47

Notes: This table shows summary statistics by fulfillment type. Each panel reports descriptive statistics for a key variable by fulfillment type.

G.2 *t*-tests of product quality in treatment and control

Table G7: *t*-test results: by product

	Control (1)	Treatment (2)	Difference (3)	p-value (4)
rating	4.51	4.51	-0.00 (3.61)	0.00
reviews	149.59	152.77	3.18 (-1.03)	0.30
price	454.48	484.52	30.03 (-1.15)	0.25
Observations	558,844	545,005		

Notes: This table shows the results of the *t*-test between treatment and control groups at the product-level.

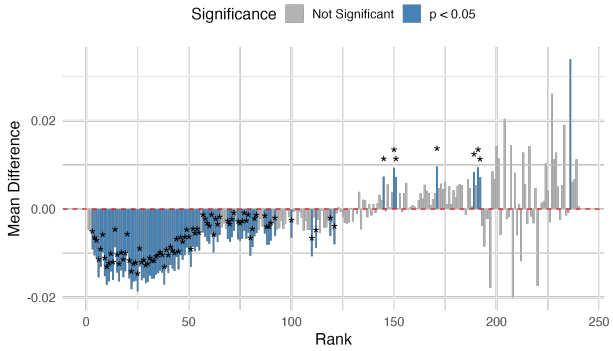
Table G8: *t*-test results: by product-rank

	Control (1)	Treatment (2)	Difference (3)	p-value (4)
rating	4.51	4.51	-0.01*** (36.39)	0.000
reviews	413.13	423.51	10.38*** (-9.55)	0.000
price	389.12	370.61	-18.51*** (13.69)	0.000
Observations	11,019,695	11,280,093		

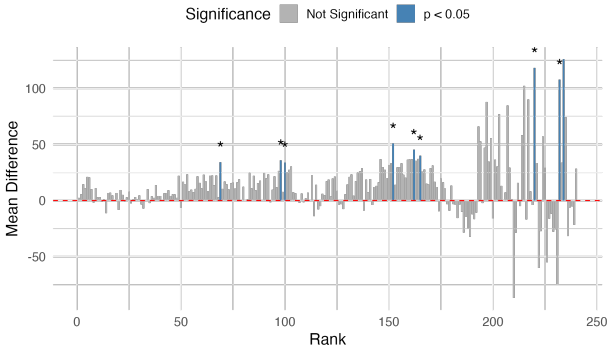
Notes: This table shows the results of the *t*-test between treatment and control groups at the product-rank level.

Figure G12: t-test results by product-rank

(a) Product ratings



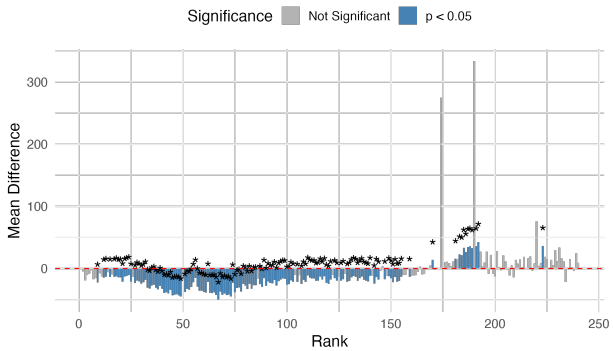
(b) Product reviews



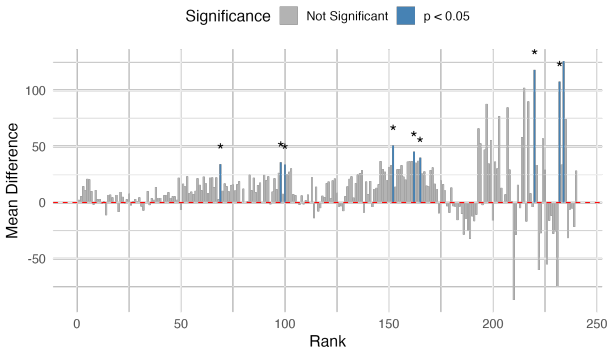
Notes. These figures show the t-test results between the control and treatment groups in the main experiment at the product-rank level. Panel a shows the results for the product ratings. Panel b shows the results for the product reviews.

Figure G13: t-test results by product-rank

(a) Product prices



(b) Product reviews

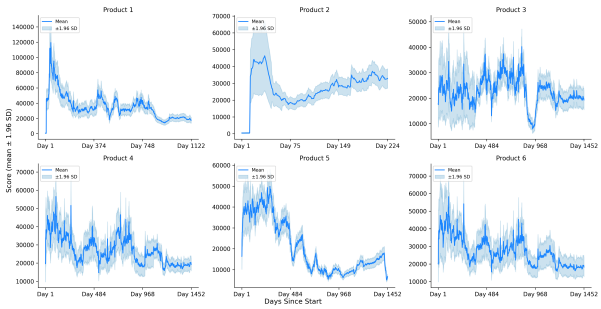


Notes. These figures show the t-test results at the product-rank level. Panel a shows the results for the product prices. Panel b shows the results for the product reviews.

G.3 Examples of posterior score evolutions

Figure G14 illustrates real-life posterior score evolutions for a sample of products over time: the platform may start with high priors for some products and downgrade them over time (Products 5 and 6), or conversely, start with low priors and upgrade/update them as time progresses (Products 1 and 2, for example).

Figure G14: Examples of posterior evolutions



Notes: This figure shows posterior score updates over anonymized dates as the product remains listed on the website.

H Approximate Theoretical Guarantees (with Neighborhood Interference)

We state identification and large-sample properties for the neighborhood-aware IPS/DR/ADR estimators defined above, clarifying what can and cannot be ported from analyses that assume no interference. We then outline additional conditions under which DATS-style regret rates might plausibly hold, emphasizing that these rates are not proved here.

Standing assumptions. Let H_{t-1} be the σ-field generated by past actions and outcomes up to $t-1$. We assume:

(A1) **Consistency.** For each (j,k) and realized neighborhood S_{jt} when $Z_{jt}(k)=1$,

$$y_{jt} = y_{jt}(k, S_{jt}), \quad y_{jt} \in [0,1].$$

(A2) **Positivity / overlap (pair and neighborhood).** For all (X_{jt}, U_t) in the evaluation support,

$$\pi_t\Big(j,k \mid H_{t-1}, X_{jt}, U_t\Big) > 0, \quad q_0(\cdot \mid X_{jt}, U_t,\, j@k) \;\ll\; \tilde{q}_t(\cdot \mid X_{jt}, U_t,\, j@k).$$

(A3) **Ignorability given context and (counterfactual) neighborhood.** For each (j,k) and any neighborhood S in the support,

$$Z_{jt}(k) \;\perp\!\!\!\perp\; y_{jt}(k,S) \;\Big|\; \Big(X_{jt}, U_t, S, H_{t-1}\Big).$$

That is, conditional on $(X_{jt}, U_t, S, H_{t-1})$, pair placement is as-if randomized.

(A4) **Controlled ratios and finite second moments.** The neighborhood density ratio

$$w_t(S \mid X_{jt}, U_t,\, j@k) = \frac{q_0(S \mid X_{jt}, U_t,\, j@k)}{\tilde{q}_t(S \mid X_{jt}, U_t,\, j@k)}$$

is well-defined on the support of $\tilde{q}_t$ and satisfies $\sup_t[w_t^2(S \mid X_{jt}, U_t,\, j@k)] < \infty$ (possibly via clipping or stabilization).

(A5) **Consistent simulation (when used).** If $\pi_t(j,k \mid H_{t-1}, X_{jt}, U_t)$ and/or $\tilde{q}_t(\cdot \mid X_{jt}, U_t,\, j@k)$ are estimated by Monte Carlo resimulation of the logging/target rule at the observed state, the estimators are uniformly consistent on the evaluation support; their error is $o_p(1)$ relative to sampling error.

Identification and double robustness (estimation). Define

$$\Gamma_t^{(j,k)} = m_{t-1}^{(j,k)}(X_{jt}, U_t, S_{jt}) + \frac{Z_{jt}(k)}{\pi_t(j,k \mid H_{t-1}, X_{jt}, U_t)} \Big\{ y_{jt} - m_{t-1}^{(j,k)}(X_{jt}, U_t, S_{jt}) \Big\} w_t(S_{jt} \mid X_{jt}, U_t,\, j@k).$$

Proposition 1 (Identification and double robustness under interference)**.** *Under* (A1)–(A5)*, if either*

(i) $m_{t-1}^{(j,k)}(x,u,S) = \Big[y_{jt}(k,S) \mid x,u,S \Big]$ *a.s., or*

(i) both $\pi_t(j,k \mid H_{t-1}, X_{jt}, U_t)$ *and* $w_t(S \mid X_{jt}, U_t,\, j@k)$ *are correctly specified,*

then the DR estimators are consistent: $\hat{r}_{j,k}^{\mathrm{DR}} \to r_{j,k}^{\star}$ *and* $\hat{r}_{j}^{\mathrm{DR}} \to r_{j}^{\star}$*. If both nuisances are correct,* $\hat{r}_{j,k}^{\mathrm{DR}}$ *is unbiased for* $r_{j,k}^{\star}$*.*

Sketch. The augmentation term has mean zero under either nuisance correctness; interference is handled by conditioning on S and reweighting from $\tilde{q}_t(\cdot \mid X, U, j@k)$ to $q_0(\cdot \mid X, U, j@k)$.

Asymptotic normality for ADR means. Define

$$\hat{\mu}_t^{(j,k)} = \frac{\sum_{s \le t} \sqrt{\pi_s(j,k \mid H_{s-1}, X_{js}, U_s)}\, \Gamma_s^{(j,k)}}{\sum_{s \le t} \sqrt{\pi_s(j,k \mid H_{s-1}, X_{js}, U_s)}},$$

$$\hat{\sigma}_t^{2,(j,k)} = \frac{\sum_{s \le t} \pi_s(j,k \mid H_{s-1}, X_{js}, U_s) \Big[(\Gamma_s^{(j,k)} - \hat{\mu}_t^{(j,k)})^2 + 1 \Big]}{\Big(\sum_{s \le t} \sqrt{\pi_s(j,k \mid H_{s-1}, X_{js}, U_s)} \Big)^2}.$$

Theorem 2 (ADR CLT with neighborhood weighting)**.** *Under* (A1)–(A5)*, either of the conditions in Proposition 1, and* $\sup_t [w_t^2(S \mid X_{jt}, U_t,\, j@k)] < \infty$*, suppose in addition that nuisance estimation error is* $o_p(1)$ *(e.g., via cross-fitting) and that Lindeberg's condition holds for the martingale difference array* $\{\Gamma_t^{(j,k)} - (\Gamma_t^{(j,k)} \mid H_{t-1})\}$*. Then*

$$\sqrt{T} \left(\hat{\mu}_T^{(j,k)} - r_{j,k}^{\star} \right) \;\Rightarrow\; \mathcal{N}(0, \sigma_{j,k}^2),$$

with $\sigma_{j,k}^2$ *finite and consistently estimated by* $\hat{\sigma}_T^{2,(j,k)}$*.*

Sketch. This follows from a self-normalized martingale CLT, with stabilization by $\sqrt{\pi_t}$; the "+1" term prevents early degeneracy.

What we do *not* claim (limitations vs. no-interference proofs). Existing finite-time regret analyses for DATS/ADR-style algorithms assume *no interference*: an arm's reward does not depend on simultaneous choices. In our slate setting, y_{jt} depends on the neighborhood S_{jt}, which is jointly determined by the slate. This breaks key steps in standard proofs:

- The per-arm reward is not conditionally independent of other arms' selections; concentration arguments that treat arms as independent decisions no longer apply verbatim.
- The logged propensity for a pair (j,k) depends on the *full* slate constructor; if the slate rule used for learning does not preserve the pairwise marginals used in ADR weighting, the martingale structure needed for concentration is compromised.
- Variability introduced by the density ratio $w_t = q_0/\tilde{q}_t$ (which can be heavy-tailed without clipping) inflates constants in deviation bounds and may render rates vacuous without additional control.

Hence, we do *not* claim that published DATS regret theorems transfer as-is to our setting with interference.

When regret-style bounds might hold (informal). If one imposes extra regularity that "separates" pairwise values from slate-level dependence in the *evaluation* metric, DATS-style rates may be recovered up to constants. Concretely, suppose:

(R1) **Sufficient exposure mapping.** The summary S_{jt} is sufficient so that the target value $r^\star_{j,k} = {}_{X,U\ S\sim q_0}[\,(y_{jt}(k,S) \mid X,U,S)\,]$ is well-defined and time-invariant once q_0 is fixed.

(R2) **Bounded ratios / clipping.** $\sup_t \|w_t\|_\infty \le B < \infty$ (or weights are clipped so that second moments are bounded uniformly in t).

(R3) **Exploration floor.** The slate builder guarantees marginal pair propensities $\pi_t(j,k \mid \cdot) \ge \gamma_t$ with $\gamma_t \ge \gamma/T^\alpha$ for some $\alpha \in [0,1)$.

(R4) **Matched marginals.** The sequential without-replacement slate construction preserves the pairwise marginals used in ADR.

(R5) **Accurate nuisance simulation.** Monte Carlo estimates of π_t and $\tilde{q}_t$ are uniformly consistent (or their error is dominated by sampling error).

Informal claim. Under (A1)–(A5) and (R1)–(R5), an ADR-driven TS policy with per-pair sampling from $\mathcal{N}(\hat{\mu}_t^{(j,k)}, \hat{\sigma}_t^{2,(j,k)})$ and a probability-matching slate rule with an exploration floor can attain a DATS-style regret rate of order

$$\mathrm{Regret}(T) \;\lesssim\; \tilde{\mathcal{O}}\Big(B^2\, M^2\, \sqrt{T}\Big),$$

where M is the number of admissible pairs (j,k) and $\tilde{\mathcal{O}}$ hides logarithmic factors and constants depending on the exploration floor and gaps. A rigorous proof would require new arguments to handle the slate-induced dependence and the neighborhood ratios.

Practical safeguards. Regardless of asymptotics, we recommend: (i) logging (or consistently simulating) *pair* propensities and neighborhood laws; (ii) clipping w_t and reporting effective sample size; (iii) maintaining an explicit exploration floor in the slate builder; and (iv) monitoring overlap between q_0 and $\tilde{q}_t$.

I Additional details on N-ADR

Assumptions for Consistency. The N-ADR estimator is consistent for $r^\star_{j,k}$ under the following conditions:

A1 Positivity (Overlap). For all (X, j, k) with $\Pr(Z_{jt}(k) = 1 \mid X) > 0$ under the target policy, the logging policy satisfies

$$\tilde{q}_t(S \mid X, j@k) > 0 \quad \text{whenever} \quad q_0(S \mid X) > 0.$$

That is, every neighborhood with positive probability under q_0 must also have been observed under the logging policy.

A2 Neighborhood Sufficiency. The summary $\psi(S_{jt})$ is rich enough that conditioning on $(X_{jt}, \psi(S_{jt}))$ blocks all backdoor paths from neighborhood composition to the outcome:

$$y_{jt}(k, S) \perp\!\!\!\perp S_{jt} \;\; X_{jt}, \psi(S_{jt}).$$

This ensures that the density ratio reweighting by $w_t(S \mid X, j@k)$ removes neighborhood bias.

A3 Correct Specification of Nuisance Models. Either:

(a) The outcome regression $m^{(j,k)}_{t-1}(x, S)$ is correctly specified, *or*

(b) Both the exposure propensity $\pi_t(j, k)$ and the neighborhood density ratio $w_t(S \mid X, j@k)$ are correctly specified.

This is the standard *double robustness* condition.

A4 Stationarity of q_0. The target neighborhood distribution $q_0(S \mid X)$ is fixed over the period of estimation or is updated in a known way.

Implications in Practice. In applied settings, these assumptions have the following practical consequences:

- *Checking overlap (A1):* Verify that for each (X, j, k), the observed logging data covers the range of neighborhoods likely under q_0. In practice, this can be approximated by computing effective sample sizes (ESS) after weighting; extremely small ESS indicates potential violations.
- *Designing $\psi(S)$ (A2):* Include features in $\psi(S)$ that plausibly capture competitive and complementary effects, such as embedding-based similarities, diversity measures, and cluster counts. Validate sufficiency by testing whether adding richer features materially changes residuals in $m^{(j,k)}_{t-1}$.
- *Model choice and validation (A3):* Use flexible, regularized learners (e.g., gradient boosting, neural nets) for m_{t-1}, π_t, and the density ratio model to reduce specification risk. Cross-fitting

or sample-splitting helps avoid overfitting bias in DR estimation.

- *Stability of q_0 (A4):* If q_0 changes (e.g., platform objectives shift), update the density ratio model accordingly and re-estimate $w_t(S \mid X, j@k)$ on the new target distribution.

J Implementing Post-hoc Debiasing on the Illustrative Example

We apply the essence of the proposed de-biasing methodology to the two-neighborhood example with items A and B. To keep the exposition minimal and transparent, we *condition on a fixed item a and position k* so that the marginal exposure propensity cancels (the $\mathbf{1}\{\cdot\}/\pi_t$ factor is constant within (a, k)). The only remaining reweighting comes from the neighborhood distribution shift $q_\star(S \mid a)$ vs. $q_0(S \mid a)$.

Step 1: Choose the model term. Let the *model-based* term for item a be the pooled posterior mean from a Beta$(1, 1)$ prior and the observed (n, c) totals:

$$\hat{m}_a \;=\; \frac{\alpha_0 + c}{\alpha_0 + \beta_0 + n} \qquad \text{with} \quad (\alpha_0, \beta_0) = (1, 1).$$

For Day 1, this gives

$$\hat{m}_A = \frac{165}{1002} \approx 0.16467, \qquad \hat{m}_B = \frac{214}{1002} \approx 0.21360.$$

Step 2: Estimate (or set) neighborhood mixes. From logs (Day 1),

$$q_0^A(\mathrm{S}) = 0.9,\ q_0^A(\mathrm{D}) = 0.1; \qquad q_0^B(\mathrm{S}) = 0.05,\ q_0^B(\mathrm{D}) = 0.95.$$

We evaluate under the target mix $q_\star(\mathrm{S}) = q_\star(\mathrm{D}) = \frac{1}{2}$ and define

$$\rho(s) \;=\; \frac{q_\star(s)}{q_0(s)}.$$

Step 3: Counts-only (conditioning on (a, k)). Aggregate Day 1 counts by neighborhood $s \in \{\mathrm{S}, \mathrm{D}\}$: n_s impressions and c_s clicks. With $N_a = \sum_s n_s$, the Neighborhood-ADR estimator becomes

$$\widehat{\mu}_a^{\mathrm{N-ADR}} \;=\; \hat{m}_a \;+\; \frac{1}{N_a} \sum_{s \in \{\mathrm{S},\mathrm{D}\}} \rho(s) \Big(c_s - n_s\, \hat{m}_a \Big). \tag{11}$$

This is the DR/N-ADR form with (i) the model residual $y - \hat{m}$, and (ii) importance weights that correct only the neighborhood misalignment.

Step 4: Plug-in for A (Day 1). For A,

$$(n_\mathrm{S}, c_\mathrm{S}) = (900, 144), \quad (n_\mathrm{D}, c_\mathrm{D}) = (100, 20), \quad N_A = 1000,$$

$$\rho_A(\mathrm{S}) = \tfrac{0.5}{0.9} = 0.5556, \qquad \rho_A(\mathrm{D}) = \tfrac{0.5}{0.1} = 5.0.$$

Residuals:

$$c_{\mathrm{S}} - n_{\mathrm{S}}\hat{m}_A = 144 - 900 \cdot 0.16467 = -4.2039, \qquad c_{\mathrm{D}} - n_{\mathrm{D}}\hat{m}_A = 20 - 100 \cdot 0.16467 = +3.5329.$$

$$\widehat{\mu}_A^{\mathrm{N-ADR}} = 0.16467 + \frac{0.5556(-4.2039) + 5.0(3.5329)}{1000} \approx 0.18.$$

Step 5: Plug-in for B (Day 1). For B,

$$(n_{\mathrm{S}}, c_{\mathrm{S}}) = (50, 4), \quad (n_{\mathrm{D}}, c_{\mathrm{D}}) = (950, 209), \quad N_B = 1000,$$

$$\rho_B(\mathrm{S}) = \tfrac{0.5}{0.05} = 10, \qquad \rho_B(\mathrm{D}) = \tfrac{0.5}{0.95} \approx 0.5263.$$

Applying (11) yields

$$\widehat{\mu}_B^{\mathrm{N-ADR}} = 0.15.$$

(As expected, N-ADR returns the target-mix means when the neighborhood cell rates are stable and the target mix is 50–50.)

Step 6: Use debiased means for the cost-aware rank. With costs $c_A = 1.15$ and $c_B = 1.00$, a conversions-per-dollar score becomes

$$\mathrm{Score}_A^{\mathrm{N-ADR}} = \frac{\widehat{\mu}_A^{\mathrm{N-ADR}}}{c_A} = \frac{0.18}{1.15} \approx 0.1565, \qquad \mathrm{Score}_B^{\mathrm{N-ADR}} = \frac{0.15}{1.00} = 0.1500,$$

so $A \succ B$. In contrast, using pooled means (biased by q_0),

$$\mathrm{Score}_A^{\mathrm{pooled}} = \frac{0.1647}{1.15} \approx 0.1432, \qquad \mathrm{Score}_B^{\mathrm{pooled}} = \frac{0.2136}{1.00} \approx 0.2136,$$

would incorrectly rank $B \succ A$.

Table J9: Pooled vs. ADR (target-mix) means and cost-normalized scores (Day 1).

Item	$\hat{m}$ (pooled)	$\widehat{\mu}^{\mathrm{ADR}}$	Cost c	Score (pooled / ADR)
A	0.1647	0.1800	1.15	0.1432 / 0.1565
B	0.2136	0.1500	1.00	0.2136 / 0.1500

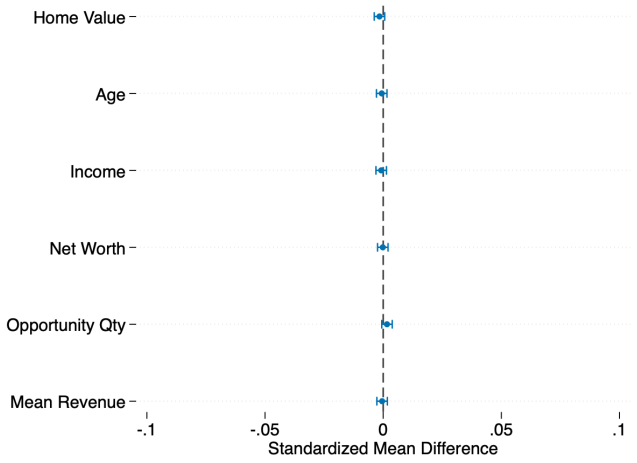
K Randomization Checks

Table K10: Randomization checks (return to p. 22)

	Control (1)	Treatment (2)	Difference (3)	p-value (4)
Mean Historical Purchases ($)	15.44	15.41	0.02 (1.22)	0.22
Historical Quantity Bought	2.17	2.17	-0.00 (-1.18)	0.24
Estimated networth	12.90	12.90	-0.00 (-0.12)	0.90
Estimated income	11.64	11.64	0.00 (0.23)	0.82
Age	3.91	3.91	0.00 (0.38)	0.71
Home Value	12.56	12.56	0.00 (0.60)	0.55
Observations	1,565,497	3,128,060		

Notes: This table shows randomization checks in the experiment.

Figure K15: Out-of-stock spillover test: treatment vs. control purchases



Notes: This figure plots coefficients from an OLS regression of the out-of-stock rate on purchase counts in control and treatment groups. Points show estimates; horizontal bars denote 95% confidence intervals. The analysis tests for potential interference between units by examining whether purchases in one group predict stockouts in the other.

L Results of the experiment with no de-biasing

Table L11: Effects of the biased algorithm on shipping distances and delivery times

	OLS			*Logit*				
	(1) distance	(2) log(distance)	(3) log(delivery)	(4) < 250 mi	(5) $250 - 1500$ mi	(6) $1500 - 2000$ mi	(7) $2000 - 2500$ mi	(8) > 2500 mi
Cost-aware biased algorithm	-49.349***	-0.078***	-0.016***	0.124***	0.007	-0.116***	-0.092***	-0.075**
	(3.724)	(0.006)	(0.003)	(0.013)	(0.010)	(0.017)	(0.015)	(0.035)
Intercept	953.392***	6.415***	1.648***	-1.495***	0.293***	-2.217***	-1.950***	-3.822***
	(2.620)	(0.004)	(0.002)	(0.009)	(0.007)	(0.012)	(0.011)	(0.024)
Baseline probability (control)	.	.	.	18.312	57.275	9.819	12.454	2.141
% Point effect (treatment vs control)	.	.	.	1.935	0.166	-0.979	-0.971	-0.151
Relative % effect (treatment vs control)	.	.	.	10.569	0.290	-9.973	-7.798	-7.057
Observations	161,339	161,339	157,304	161,339	161,339	161,339	161,339	161,339

Notes. The table reports treatment effects from Equation 5 for ordered products. Columns (1)–(3) present OLS estimates for average shipping distance (miles), percentage change in shipping distance, and delivery time (days). Columns (4)–(8) report logit marginal effects for the probability that an order falls within a given distance category. Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Table L12: Effects of the biased algorithm on orders, revenue, and profit

	Logit	*OLS*		
	(1) clicked	(2) log(# orders)	(3) log(revenue)	(4) log(profit)
Cost-aware biased algorithm	-0.016***	-0.000***	-0.003***	-0.001**
	(0.002)	(0.000)	(0.001)	(0.001)
Intercept	0.210***	0.017***	0.114***	0.073***
	(0.001)	(0.000)	(0.000)	(0.000)
Observations	5,723,821	5,728,770	5,728,770	5,728,770

Notes. The table reports treatment effects on total orders, revenue, and profit. Data is at the consumer level (Equation 6). Statistical significance: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.