Learning to Rank an Assortment of Products

Submitted to *Management Science*
manuscript (Please, provide the manuscript number!)

Learning to Rank an Assortment of Products

Kris J. Ferreira
Assistant Professor, Harvard Business School, Boston, MA 02163, kferreira@hbs.edu

Sunanda Parthasarathy
Wayfair, Boston, MA 02116, sparthasarathy@wayfair.com

Shreyas Sekar
Harvard Business School, Boston, MA 02163, ssekar@hbs.edu

We consider the product ranking challenge that online retailers face when their customers typically behave as "window shoppers": they form an impression of the assortment after browsing products ranked in the initial positions and then decide whether to continue browsing. We design online learning algorithms for product ranking that maximize the number of customers who engage with the site. Customers' product preferences and attention spans are correlated and unknown to the retailer; furthermore, the retailer cannot exploit similarities across products owing to the fact that the products are not necessarily characterized by a set of attributes. We develop a class of online learning-then-earning algorithms that prescribe a ranking to offer each customer, learning from preceding customers' clickstream data to offer better rankings to subsequent customers. Our algorithms balance product popularity with *diversity*: the notion of appealing to a large variety of heterogeneous customers. We prove that our learning algorithms converge to a ranking that matches the best-known approximation factors for the offline, complete information setting. Finally, we partner with Wayfair - a multi-billion dollar home goods online retailer - to estimate the impact of our algorithms in practice via simulations using actual clickstream data, and we find that our algorithms yield a significant increase (5-30%) in the number of customers that engage with the site.

Key words: Online Learning, Product Ranking, Assortment Optimization, E-Commerce

1. Introduction

Over the last decade, online retail has experienced significant growth and is becoming a larger portion of the retail industry. Online retailers[1] face many new operational opportunities to drive customer engagement and revenue growth. One such opportunity is the decision of how to rank products on their website. It is well established in the academic literature and known by retailers that more customers view products ranked in top positions compared to products ranked towards the bottom of the page where customers must scroll down to view. Furthermore, the product ranking can have a causal effect on the number of products searched (see, e.g., Ursu (2018)). Thus, the ranking of products is an important decision since this allows the retailer to control which products are seen by most customers, impacting the customers' browsing behavior, purchase decisions, and long-term interest in the retailer.

In this paper, we study the ranking challenge in a particular setting that, to the best of our knowledge, is yet to be studied in the academic literature. We consider a retailer selling products that are not necessarily characterized by a set of attributes; this could be because, e.g., (*i*) the products are not substitutes and thus do not share common attributes, (*ii*) the products are stylistic in nature and thus attributes would be subjective (e.g., whether or not an item of furniture is considered contemporary), or (*iii*) the retailer simply does not record attribute information. The typical customer (she) does not have a good idea of the assortment or style of products offered, and may not have a particular product in mind that she is looking for. She forms an opinion about the assortment and style after browsing products in the initial ranks; we define the customer's "first impression window" as the number of products she views to help form her initial opinion, which may differ by customer and is unknown to the retailer. Many retailers face such customers. In general, retailers who change their assortment of products frequently relative to the frequency of a customer's visit likely face such customer behavior.

[1] We will refer to any retailer with an online presence, regardless of whether they also have brick-and-mortar stores, as an "online retailer". When the context is clear, we will refer to online retailers as simply "retailers" for brevity.

We assume that each customer browses products in the initial ranks up to her first impression window, and if she sees a product that she likes, she is "hooked" and continues browsing in hopes of finding additional products that she likes. If she does not see a product she likes within her first impression window, she believes she may not find anything she likes in the assortment and therefore leaves the site[2] early. Customers click on products that they like in order to learn details about the product (e.g., sizes available, dimensions, material, etc.). In contrast to other literature on product ranking, we do not assume that customers are shopping with the intent to purchase at most one product from a set of substitutable products offered by the retailer, although our model allows for this behavior; instead, we are motivated by settings where customers browse a set of products that are not necessarily substitutable and may like, and perhaps purchase, multiple products. The retailer observes the set of products that customers like via clicks and dynamically changes the product ranking over time to learn the ranking that maximizes the number of hooked customers, or equivalently, minimizes the number of customers who do not see anything they like and abandon the site prematurely.

Customers in our setting are analogous to "window shoppers" in brick-and-mortar retail. In a physical mall, window shoppers stroll through the mall looking at products in the retailers' display cases to learn about the assortment of products the retailer is offering. If they see a product they like, they are hooked and enter the store to browse more products; otherwise, they avoid the store altogether. The online retailer's goal of maximizing the number of hooked customers - the number of customers who engage with the site - is analogous to the brick-and-mortar retailer's goal of maximizing store traffic, a key customer engagement metric for many retailers. Although the objective of profit maximization is more commonly studied in the academic literature, many retailers in competitive markets have the primary objective of maximizing market share and use such customer engagement metrics to do so, including the retailer we partnered with for this

[2] "Site" refers to the page(s) that display products in an assortment, which may include only a subset of products the retailer offers.

research, Wayfair (www.wayfair.com) - a multi-billion dollar home goods online retailer. That said, we show that we can extend our model, analysis, and results to other common objectives such as maximizing revenue (or profit) and maximizing total clicks in Appendix B.

In order to maximize the number of hooked customers, the retailer must consider both product popularity and diversity in its ranking decisions. Popularity is an obvious consideration: by including a very popular product in a top rank, the retailer can hook many customers. Interestingly, ranking by popularity is one of the most common ranking approaches studied in the academic literature and implemented in practice. However, *diversity* (or variety) of products is also an important notion; here, we consider two products as being diverse if the set of customers that each hooks has little or no overlap. Two products that are very popular may hook the same customers. Incorporating diversity allows the retailer to hook a broader base of customers. Importantly, we note that our definition of diversity does not require *a priori* knowledge about how similar the products in the assortment are, which in other diverse ranking literature is often calculated as a function of product attributes; in fact, we do not require that products are characterized by a set of attributes. Rather, in our work, we must learn diversity over time by observing customer clickstream data.

Our main contributions in this paper are three-fold. First, we introduce and study the product ranking challenge for retailers who often face "window shopping" consumer behavior. This type of consumer shopping behavior is well recognized in the consumer psychology literature as "hedonic browsing" (see, e.g., Moe (2003)), which cites the importance of offering such consumers a diverse set of products, and we are the first to consider the ranking decision for retailers facing such customers. Our model highlights the importance of both product popularity and diversity considerations when making ranking decisions.

Second, we develop a class of prescriptive, dynamic ranking algorithms that balance popularity and diversity to learn the ranking that maximizes the number of hooked customers. In the (unrealistic) complete information setting - where the retailer knows the joint distribution of customers' product interests along with their first impression windows - we relate a special case of our model

to the maximum coverage problem and extend its well-known greedy algorithm to our more general model. Then, when the retailer has no prior knowledge of product interests and first impression windows, we develop algorithms that dynamically learn the greedy ranking from the complete information setting by offering different rankings over time. Our algorithms circumvent having to learn the joint distribution of customers' product interests and their first impression windows, and instead learn the optimal ranking directly. The algorithms that we propose vary based on their tradeoff of optimality gap vs. speed of learning and can thus be applied to retailers of varying sizes.

For our third contribution, we partner with Wayfair to develop a realistic model and implementable algorithms, and ultimately estimate their impact via simulations using actual clickstream data. Across six different product assortments, our algorithms hook an average of 5-30% more customers than Wayfair's static, popularity-based ranking. Furthermore, our algorithms hook 89-95% of the total number of customers hooked by an (omniscient) offline benchmark. These simulations using real clickstream data illustrate that our algorithmic contributions can make a significant impact in practice.

1.1. Related Literature

Connections to Product Ranking Product ranking has been well-studied in the academic literature, although there are key differences in modeling assumptions and analysis between this stream of work and ours. We compare the major aspects of the most relevant product ranking papers in Table 1 and speak to their implications below.

First, all product ranking papers assume that customers are shopping with the intent to purchase at most one product from a set of substitutable products offered by the retailer. Making this assumption and considering sequential, costly search, many of these papers build upon the seminal paper by Weitzman (1979) and model the consumer's shopping behavior as an optimal stopping problem, trading off the cost of searching one more product vs. its expected utility. Most other papers use a multinomial logit (MNL) model. As mentioned previously, we do not assume that customers shop with the intent to purchase at most one product from a set of substitutable

	Assumes customers buy at most one product	Models as optimal stopping problem (OSP) or MNL	Assumes independent utility / interest across products	Assumes *a priori* full knowledge of products' intrinsic utilities and distribution of random utilities	Descriptive (D) vs. Prescriptive (P)
Aouad and Segev (2020)	✓	MNL	✓		P
Chen and Yao (2016)	✓	OSP	✓	✓	D
De Los Santos and Koulayev (2017)	✓	OSP	✓	✓	P
Donnelly et al. (2020)	✓		✓		P
Dzyabura and Hauser (2019)	✓	OSP			D
Gallego et al. (2020)	✓	MNL			P
Ghose et al. (2012)	✓	MNL	✓	✓	P
Derakhshan et al. (2020)	✓	OSP	✓		P
Kim et al. (2016)	✓	OSP	✓	✓	D
Koulayev (2014)	✓	OSP		✓	D
Ursu (2018)	✓	OSP	✓	✓	D

Table 1 Modeling Assumptions and Analysis Techniques in Product Ranking Literature

products, although our model allows for this behavior; rather, we are motivated by settings where customers exhibit hedonic browsing behavior (window shopping) and products are not necessarily substitutable. Due to this difference, modeling our problem using the optimal stopping framework of Weitzman (1979) or a choice model like the MNL model is not appropriate for our setting.

Hedonic browsing shopping behavior is well recognized in the consumer psychology literature (see, e.g., Moe (2003)). Browsing has been further described in the consumer behavior literature as a screening activity that is independent of any purchase needs (Bloch et al. 1989). Research has shown that first impressions matter; for example, Dawson and Kim (2010) suggests that if a shopper is not hooked within 30 seconds of browsing the site, she will leave. Cho and Youn-Kyung (2012) suggests that shoppers form a quick judgment of the assortment after viewing a few products, which may or may not warrant further exploration. We capture this initial screening behavior and time spent on the site via the customer's first impression window.

Second, most product ranking papers assume for tractability that customer interest in any two products is independent. We do not make this assumption, and in fact, our learning algorithm converges to the well-studied popularity ranking when independence holds.

Third, most other papers assume for tractability that the intrinsic utilities of products, as well as the distribution of random utilities, are known to the retailer and customers before customers actually view the products. We do not make this assumption and instead use a basic model of customer learning. In Dzyabura and Hauser (2019), customers learn their preferences for product attributes as they search. In Derakhshan et al. (2020), consumers learn their intrinsic product utilities as they browse products. Although Aouad and Segev (2020) and Gallego et al. (2020) do not consider customer learning, they do assume exogenous first impression windows as do we. Even with critical differences in modeling assumptions, Gallego et al. (2020) proposes an algorithm that is similar in spirit to our offline algorithm; that said, their analysis does not extend to our model.

Finally, many of these papers are descriptive in nature, evaluating the impact of a static ranking on consumer behavior. In contrast, our work is prescriptive, in that we aim to find a ranking that maximizes a retailer's objective. The other papers that are prescriptive in nature consider offline algorithms that produce a static ranking, whereas ours presents an online learning algorithm that dynamically changes rankings to learn the best ordering over time.

Connections to Choice Models with Consideration Sets: The concept of the first impression window in our consumer behavior model bears similarities to consumer choice models with consideration sets,[3] where our consumer's decision is whether or not to continue browsing past her first impression window. Recently, consider-then-choose and similar two-stage models have been gaining popularity (see, e.g., Wang and Sahin (2018), Derakhshan et al. (2020), Jagabathula and Rusmevichientong (2017), and Aouad et al. (2015)); in such models, consumers first decide the subset of products that they will consider (i.e., their consideration set) and subsequently view

[3] In such models, consumers incur a non-zero search cost to view and consider products, and thus may only view a subset of all products offered.

these products to learn their utilities prior to making a purchase decision. Often, consider-then-choose models assume that customers know their intrinsic utilities or certain attributes for each product prior to viewing products and use this information to form the consideration set. When this assumption is not made, as in our work, the consider-then-choose model is also known as a fixed sample size model (De Los Santos et al. 2012), where the customer finds the products to be a priori indistinguishable. She then chooses any sample of k products to consider before making a decision. Part of our consumer behavior model is analogous to a fixed sample size model where the "fixed sample size" is her first impression window, k_t, and the "decision" is whether or not she is hooked and continues browsing; namely, our customer browses an a priori fixed number of products, k_t, prior to deciding whether or not she is hooked. Empirical work in an e-commerce setting has shown that such fixed sample size models may be more appropriate for certain settings than consideration set models based on sequential search (De Los Santos et al. 2012). It is worth noting that other product ranking papers also use a fixed sample size consumer behavior model (see, e.g., Aouad and Segev (2020) and Gallego et al. (2020)).

We highlight that the main difference between our model and choice models with consideration sets is the shopping intention of the customer. Choice models with consideration sets are appropriate for customers who intend to purchase at most one product from a set of substitutable products; whereas our model is more appropriate for customers who exhibit hedonic browsing behavior and for products that are not necessarily substitutes. Consequently, the customer's goal in our consumer behavior model - where customers view products within their first impression window - is primarily aimed at deciding whether or not to continue browsing, whereas the customer's goal in the first stage of consider-then-choose models is to narrow down the set of products that they will ultimately consider for purchase.

Connections to Learning Algorithms Motivated by applications in information retrieval, web search, and recommendation systems, there are many recent papers on learning algorithms for information ranking. Much of this work uses clickstream data to infer optimal static rankings, so we

only highlight the recent work pertaining to online learning algorithms, namely under the cascade and position-based user behavior models. In the cascade model (Kveton et al. 2015a,b) there is only one user type with a fixed but unknown set of click probabilities for each item and a known first impression window (k), and the objective is - analogous to ours - to select k items to minimize the probability of customer abandonment. However, such a model does not account for varying and unknown first impression windows that lead to the behavioral effect that fewer customers view products at lower ranks; in addition, these models assume that customer interest in a product does not depend on previous products clicked on, whereas we do not make this assumption. The cascade model essentially captures a subset selection problem since altering the order in which the k items are displayed does not affect the reward.

In the position-based model of learning-to-rank, the customer's utility or click probability for an item is the product of a position-specific and item-specific component (Lagrée et al. 2016). More generally, this model is motivated by the literature on sponsored search and position auctions (Varian 2007); similarly, Abeliuk et al. (2016) study an assortment selection problem pertaining to online advertising where the consumer behavior model is MNL, with the weights being the product of position-specific and item-specific components. Interestingly, position-based models could be considered a special case of our consumer behavior model where (i) customers' product interest probabilities and first impression windows are independent, (ii) customers have homogeneous product interest probabilities, and (iii) customer interest in a product does not depend on previous products clicked on. Our more general model relaxes these strict assumptions, which leads to materially different results. Recent research has attempted to bridge the gap between the cascade and position-based models by proposing more general approaches (Zoghi et al. 2017). In all of these works, the optimal ranking is simply the 'popularity ranking' obtained by sorting the items in descending order of click probabilities.

Our work is also related to the relatively small literature on learning optimal assortments (see, e.g., Agrawal et al. (2016), Bernstein et al. (2018), and Ulu et al. (2012)). Similar to ours, papers

in this area propose offering dynamic assortments over time to learn an optimal assortment. Key differences include (i) these papers either assume customers buy at most one product or that product interest is independent for each product, and (ii) customers observe all products in the assortment, i.e. the first impression window is known and identical for all customers. Finally, there is a larger literature on offline, static assortment optimization without learning, some of which results in diverse assortments (see, e.g., Li et al. (2015a)); the same key differences exist.

Connections to Online Submodular Optimization: Our model bears strong similarities to the well-studied stochastic submodular optimization problem (Asadpour and Nazerzadeh 2015), and in particular, a special case of our model has been studied in Radlinski et al. (2008), which focused on designing online algorithms for the selection of diverse search results. Although this work allows for heterogeneous customer types, all customers examine the same number of products (i.e., identical first impression windows) and customer interest in a product is assumed to be independent of previous products clicked on. Hence, the problem is more analogous to optimal selection (i.e., the maximum coverage problem) than ranking. In contrast, our more general model is not just concerned with subset selection, as the order in which products are presented can affect which customers are hooked. Because of this, our results and techniques do not follow directly from Radlinski et al. (2008) and the broader submodular optimization literature. Moreover, while the greedy algorithm for the maximum coverage problem provides a natural intuition on how to rank the selected products, in the online learning setting, this could take too many customers to learn. To address this concern, we draw inspiration from Badanidiyuru and Vondrák (2014) and develop a threshold-based algorithm well-suited for the ranking problem and show that it performs well both theoretically and in practice; such threshold-based algorithms are not explored in Radlinski et al. (2008).

2. Model

We consider an online retailer who offers a set of n products for sale during a selling season. For convenience, the products are labeled as $[n] = \{1, 2, \ldots, n\}$. A set $\mathcal{T}$ of customers ($|\mathcal{T}| = T$) arrive

sequentially and the retailer selects a ranking of products to offer to each customer. Formally, a ranking is a permutation $\pi : [n] \to [n]$ that maps each product i to a position $1 \leq \pi(i) \leq n$. We use $\pi^{-1}(j) \in [n]$ to denote the product that is ranked in the j^{th} position under ranking π. Clearly, if $\pi^{-1}(j) = i$, then $\pi(i) = j$ by definition. We will use π_t to denote the ranking of products displayed to customer t and leave off the subscript when the context is clear. The assortment of products is assumed to be fixed beforehand as are their prices, which reflects the fact that product ranking decisions are typically made after assortment and pricing decisions in practice. Furthermore, we assume that inventory is plentiful such that no products sell out before the end of the season.

Customers *browse* products sequentially and *click* on products that they like in order to learn more about the product (e.g., its size, material, availability, etc.), and retailers observe these clicks to help inform their subsequent ranking decisions. The set of products that a customer clicks on depends on both her intrinsic preferences and the ranking of products presented to her. Formally, each customer $t \in \mathcal{T}$ is characterized by $(\boldsymbol{p_t}, \boldsymbol{f}_t, k_t) \sim_{iid} \mathcal{D}$, where we refer to $\boldsymbol{p_t} = (p_{1t}, p_{2t}, ..., p_{nt})$ as her vector of *intrinsic click probabilities*, representing the probability that the customer would click on a product if she had not clicked on any other products prior to this. For all $i \in [n]$, $f_{it} : 2^{[n]} \to [-p_{it}, 1 - p_{it}]$ is a *bias function* that maps the set of products that the customer clicked on prior to product i to a bias on the click probability of i (i.e., additive shift on p_{it}). We refer to $k_t \in [n]$ as her *first impression window*, which represents the number of products that the customer is guaranteed to browse, regardless of whether or not she likes any of the products she sees.

More concretely, customer t browses all products within her first impression window (positions 1 to k_t) in order to learn about the assortment and general style of products offered. If the customer does not like any of the products she sees within her first impression window, she stops browsing and exits the site; otherwise, she continues browsing.[4] Given ranking π and product $i \in [n]$, we use $\chi_{it}(\pi)$ to denote the random set of products that appear prior to i in the ranking π which customer t clicks on. With probability $p_{it} + f_{it}(\chi_{it}(\pi))$, customer t likes product i and - conditional

[4] The customer may continue to browse all or only a subset of the remaining products.

on seeing product i - she clicks on it to learn more about the product.[5] The additive term $f_{it}(\chi_{it}(\pi))$ captures how the set of products that the customer has clicked on in the past, $\chi_{it}(\pi)$, affects her propensity to click on product i. We assume that for all i,t, $f_{it}(\emptyset)=0$. We impose no restrictions on the bias function $f_{it}(\chi)$, which allows us to model situations where a customer's probability for clicking on product i may decrease if she has already clicked on a similar product in the past, i.e. $f_{it}(\chi_{it}(\pi))<0$, increase if she has already clicked on a complementary product ($f_{it}(\chi_{it}(\pi))>0$), or remain unchanged ($f_{it}(\chi_{it}(\pi))=0$). Additional details on specific formulations that the bias function could take in practice can be found in Section 2.1.

The retailer has no prior knowledge of $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$ or $\mathcal{D}$, and only observes customer clicks. Note that our model allows for customer heterogeneity as the intrinsic click probabilities, bias function, and first impression window can be different across customers; in addition, these three parameters may be jointly correlated with each other. For each product i and customer t, we define the Bernoulli random variable $C_{it}(\pi)$ as $C_{it}(\pi)=1$ if customer t clicks on product i under ranking π and $C_{it}(\pi)=0$ if not. We use also $\overline{C}_{it}(\pi)$ as shorthand to denote $1-C_{it}(\pi)$ $\forall i,t$. We will be primarily concerned with the customer's clicks within her first impression window, and therefore, we define $C_{it}(\pi)=0$ when $\pi(i)>k_t$ for notational convenience. We implicitly assume that the random variable $C_{it}(\pi)$ is conditional on customer t's type, $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$, and will use this shorter notation throughout the paper.

We further define the event that customer t is "hooked" as

$$H_t(\pi)=\begin{cases}1, & \text{if } \sum_{i\in[n]} C_{it}(\pi)\geq 1\\ 0, & \text{if } \sum_{i\in[n]} C_{it}(\pi)=0\end{cases};$$

in other words, customer t is hooked if she clicks on at least one product in the assortment under ranking π. Note that $Pr(C_{it}(\pi)=1)=p_{it}+f_{it}(\chi_{it}(\pi))$ when $\pi(i)\leq k_t$, since customer t is guaranteed to have browsed product i if it is displayed within her first impression window. Since

[5] We equate the customer "liking" a product to the customer "clicking on" a product if she sees it, although our results remain unchanged if we instead assume that a customer clicks on a product she likes with probability $\lambda\in[0,1]$.

$C_{it}(\pi) = 0$ when $\pi(i) > k_t$, a necessary condition for the customer to be hooked is that she clicks on at least one product within her first impression window. Therefore, $H_t(\pi) = 1$ if and only if $\sum_{j \in [k_t]} C_{\pi^{-1}(j)t}(\pi) \geq 1$.

The problem faced by the retailer is to design a non-anticipatory algorithm that selects a ranking π_t for each arriving customer $t \in \mathcal{T}$ in order to maximize the total number of hooked customers over the season, or equivalently, to maximize the total number of customers who engage with the site by clicking on at least one product:

$$\max_{\pi_1, \pi_2, \ldots, \pi_T} \mathrm{E}\big[\sum_{t \in \mathcal{T}} H_t(\pi_t)\big]. \tag{1}$$

To summarize the sequence of events, customers arrive sequentially and for each customer $t \in \mathcal{T}$,

1. The retailer selects and offers ranking π_t.

2. The customer browses all products within her first impression window (positions 1 to k_t) and clicks on products that she likes.

3. If the customer does not like any of the products within her first impression window, she leaves the site without browsing any additional products. Otherwise, she is hooked and continues browsing, and potentially clicking on, products beyond her first impression window.

4. The retailer observes all customer clicks (but not $(\boldsymbol{p_t}, \boldsymbol{f}_t, k_t)$) and can use this information when choosing rankings to offer new customers.

We will refer to the retailer's ranking problem under the consumer behavior model described above as the *online assortment ranking* problem (OnAR). We focus on a common special class of algorithms for OnAR known as learning-then-earning algorithms - these methods focus on rapidly converging to a (near-)optimal ranking that is then fixed for the remaining customers. Such algorithms are often preferable in online retail when these sites lack accurate foreknowledge of the number of customers $|\mathcal{T}|$ who visit during the selling period. Before presenting our class of dynamic ranking algorithms for the online assortment ranking problem in Section 4, we first study the *offline assortment ranking* problem (OffAR) in Section 3 where we make the (unrealistic) assumption that the retailer knows the distribution $\mathcal{D}$ at the start of the season. Since $\mathcal{D}$ is known ahead of time,

no learning is necessary and a single, static ranking can be chosen at the beginning of the season and offered to all customers.

2.1. Model Discussion

Click Dependence on Ranking: We emphasize that although each of the three parameters that define a customer's type $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$ are independent of the ranking π_t offered, the customer's resultant clicks do in fact depend on the ranking presented to her. First, the ranking determines the set of k_t products that the customer is guaranteed to browse within her first impression window, which impacts whether she continues to browse (and potentially click on) additional products beyond the first k_t. Second, the customer's click probability for browsed product i $(p_{it} + f_{it}(\chi_{it}(\pi_t)))$ is a function of all products that were offered earlier in the ranking which she clicked on, $\chi_{it}(\pi_t)$.

Importantly, we note that our model permits arbitrary bias functions and as will be seen in Section 4, does not require the estimation of the bias function for our learning algorithms. That said, we share two natural bias functions as motivating examples for our model. First, $f_{it}(\chi_{it}(\pi_t))$ may be expressed as the sum of pairwise terms, each representing the relationship between i and some product in $\chi_{it}(\pi_t)$; more concretely, $f_{it}(\chi_{it}(\pi_t)) = \sum_{i' \in \chi_{it}(\pi_t)} a_{ii'}$ where $a_{ii'}$ is analogous to the cross-product demand elasticity (e.g., see Li et al. (2015b)) between products i and i', and could be positive or negative[6]. Second, Ahmed et al. (2012) and Srikant et al. (2010) model notions such as fatigue in recommendations and web search, respectively, by assuming that the bias function depends only on the number of prior clicks, e.g., $f_{it}(\chi_{it}(\pi_t)) = g(|\chi_{it}(\pi_t)|)$ for some function $g(\cdot)$.

Finally, we note that our model implicitly assumes that the bias function depends only on the products that received clicks, as opposed to dependence on products the customer did not click on. Although in practice it is plausible for the bias function to depend on all products in earlier ranks, we believe the first-order drivers of the bias function are more likely to be products she clicks on; due to this as well as tractability reasons, we included only the dependence on clicks in our model. The rationale behind this modeling choice is that a non-zero bias stems from the customer

[6] Note that our model allows for $a_{ii'}$ to depend on the customer t, which we have omitted here for simplicity.

gathering information about a product from its 'specification page' following a click; this action may affect her inclination towards similar or complementary products. Additionally, a customer's intention to avoid clicking on correlated products is already embedded in the distribution $\mathcal{D}$. For example, if a customer is less likely to click on a product i in the first position, then her propensity for clicking on a similar product i' at a later position would also be low according to $\boldsymbol{p}_t$. Finally, we note that even in the absence of any bias function, whether or not a customer clicks on any products affects her click probabilities of products ranked outside of her first impression window; specifically, if the customer does not click on any products within her first impression window, her click probabilities for the remaining products become zero.

Objective Function: In this paper - as is the case with the product ranking work by De Los Santos and Koulayev (2017) - our focus is on maximizing the number of customers who engage with the site, specifically the number of customers who click on at least one product. Implicit in this approach is the notion that 'hooked' customers then proceed to browse more products (beyond rank k_t), which in turn leads to a greater level of customer engagement and ultimately more future visits and transactions. In the offline setting, our objective is analogous to the goal of maximizing store traffic, a key customer engagement metric for many retailers. Our objective can also be interpreted as "minimizing abandonment", a common goal in service systems, where in our case, abandonment refers to a customer not finding any products she likes within her first impression window and thus leaving the site early before being "served" a product she likes.

Although the objective of profit maximization is more commonly studied in the academic literature, there are a variety of reasons why our objective is suitable for many online retailers, including our partner Wayfair. First, many retailers in highly competitive online markets have the primary objective to maximize market share[7] and can use such a customer engagement metric to do so. Second, online retailers frequently run special events (e.g., thematic assortments, flash sales, etc. - see Section 5 for a case study) containing discounted products aimed at boosting customer engagement and retention rather than short-term profit; again, our objective and algorithms can be used

[7] This is true even in the case of large players such as Amazon (Hoffman and Yeh 2018, Bezos 1998).

for product ranking within these events. Finally, we note that click-based measures are widely employed as proxies for customer engagement (see Bucklin and Sismeiro (2009) for a survey) and can be particularly useful in settings involving real-time decision making such as ours, as opposed to sparser, time-delayed data sources such as transactional data. Among click-based metrics, we aim to select rankings that influence users to make at least one click. Arguably, additional clicks have diminishing returns for metrics of interest such as transactions and return visits (see, e.g., Moe and Fader (2004) and Montgomery et al. (2004)), so we chose to emphasize first clicks in our objective.

We conclude our discussion by remarking that our algorithms can be applied to derive similar results for other objectives that have been studied in the literature; we devote Appendix B to extending our work to the commonly studied revenue (or profit) maximization objective, and an extension to the objective of maximizing total clicks follows similarly.

3. Offline Assortment Ranking

In this section, we consider the *offline* version of the assortment ranking problem (OffAR), where the retailer knows the distribution $\mathcal{D}$ from which customers' preferences and first impression windows are drawn at the beginning of the season. However, the retailer does not know the parameters $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$ before the arrival of each customer $t \in \mathcal{T}$. Since $\mathcal{D}$ is known ahead of time, no learning is necessary and a single, static ranking can be chosen at the beginning of the season and offered to all customers. Although our primary interest lies in the online assortment ranking problem where the retailer must learn the customers' preferences, the methods developed in this section will provide a clear understanding of the core techniques involved in the more sophisticated learning algorithms that we will present in Section 4.

PROPOSITION 1. *The offline assortment ranking problem is NP-Hard.*

The proof of this result utilizes an important special case of OffAR where $p_{it} \in \{0, 1\}$, $f_{it}(\chi) = 0$ for all $\chi \subseteq 2^{[n]}$, and $k_t = k \ \forall t \in \mathcal{T}, i \in [n]$. This special case admits the following intuitive interpretation: each customer $t \in \mathcal{T}$ is endowed with a set of preferred products $S_t \subseteq [n]$ and first impression

window $1 \leq k \leq n$ and clicks on any product $i \in S_t$ that is present within ranks $[1, k]$. The retailer's goal is to select a ranking π to maximize the expected number of customers who have at least one product from their preferred subset inside their first impression window. This special case of OffAR is a stochastic version of the *maximum coverage problem*. The maximum coverage problem is known to be NP-Hard (Hochbaum 1997, Chapter 3), and given that OffAR is a generalization, it must also be NP-Hard. Formal proofs of all analytical results are provided in Appendix A.

Given this result, it is difficult to find an optimal solution to OffAR for a reasonable problem size. Thus, a standard optimization approach is to consider approximation algorithms that come close to the optimal solution. The best approximation algorithm for the maximum coverage problem - a special case of OffAR - is a greedy algorithm that achieves a $(1 - \frac{1}{e})$ approximation factor (Theorem 3.8 in Hochbaum (1997)). Motivated by this greedy algorithm, we formally present our algorithm for the more general offline assortment ranking problem in Algorithm 1, which we will refer to as the Greedy Algorithm for OffAR (or simply Greedy Algorithm when the context is clear).

Algorithm 1 sequentially fixes products in ranks $\{1, 2, \ldots, n\}$. Assuming that ranks 1 through $r-1$ have been fixed, in iteration r, the algorithm considers all the remaining (unranked) products for position r. The product i with the maximum marginal benefit (Δ_{ir}) - i.e., that increases the probability of hooking an incoming customer by the largest amount - is fixed at rank r and the algorithm continues on to the next rank. The marginal benefit can also be interpreted as the probability that the customer clicks on the product at rank r but not on any products in ranks $\{1, 2, \ldots, r-1\}$. Note that Δ_{ir} depends on three key factors: (1) the probability that a customer clicks on product i, (2) the probability that a customer does not click on products in ranks $\{1, 2, \ldots, r-1\}$, and (3) whether or not rank r is within the customer's first impression window. The combination of the first two factors can be interpreted as the balance of *popularity* and *diversity*. On the one hand, the retailer wants to offer popular products that are likely to be clicked on; on the other hand, the retailer wants to select diverse products to hook a more heterogeneous set of customers.

Algorithm 1: Greedy Algorithm for OffAR

Initialize null ranking $\pi^g(i) = \emptyset$ for all $1 \leq i \leq n$, and initialize unranked products $U = [n]$;

for $r = 1$ *to* n **do**

  for $i \in U$ **do**

    Let $\tilde{\pi}^g \leftarrow \pi^g$;

    $\tilde{\pi}^g(i) \leftarrow r$;

    $\Delta_{ir} = E_{t\sim\mathcal{D}}[H_t(\tilde{\pi}^g) - H_t(\pi^g)]$;

  end

  Let $i_r^* = \arg\max_{i\in U} \Delta_{ir}$;

  Set $\pi^g(i_r^*) \leftarrow r$;

  $U \leftarrow U \setminus i_r^*$.

end

Algorithm 1 reduces to the standard greedy algorithm for the maximum coverage problem for the special case of OffAR where $p_{it} \in \{0,1\}$, $f_{it}(\chi) = 0$ for all i, t and $k_t = k\ \forall t \in \mathcal{T}, i \in [n]$, which is known to provide a $(1 - \frac{1}{e})$-approximation factor to the optimal offline solution. The important yet subtle generalization in our algorithm can be seen in the definition of Δ_{ir}, which incorporates varying first impression windows and probabilistic interests that may depend on past products clicked on via the bias function. The following result shows that for our more general offline assortment ranking problem, the Greedy Algorithm still achieves a $\frac{1}{2}$-approximation factor.

THEOREM 1. *The Greedy Algorithm for OffAR results in ranking π^g such that the probability that an incoming customer is hooked is at least one-half that of the optimal ranking π^*, i.e. Algorithm 1 achieves a $\frac{1}{2}$-approximation factor for OffAR.*

The proof involves establishing a relationship between the fraction of customers hooked due to each rank in π^* with the corresponding rank in π^g; this allows us to prove a bound on $H_t(\pi^*)$ in terms of $H_t(\pi^g)$ since the objective function can be written as the sum over all ranks of the

fraction of customers hooked. To establish the relationship, we find it beneficial to introduce a hypothetical "composite" ranking for any fixed customer t, $\tilde{\pi}_t^g$, whose first k_t products coincide with the top products in π^g and second k_t products with those in π^*. We further consider a hypothetical customer $\tilde{t}$ who has the same click probabilities and bias functions as customer t but with first impression window $2k_t$. We then show that the probability that customer t gets hooked by ranking π^* is smaller than the probability that our hypothetical customer $\tilde{t}$ gets hooked by $\tilde{\pi}_t^g$; thus we can focus on comparing any fixed customer t's hook probability under π^g with customer $\tilde{t}$'s hook probability under $\tilde{\pi}_t^g$.

First, we note that the probability that customer $\tilde{t}$ is hooked by any of the first k_t products in $\tilde{\pi}_t^g$ is identical to the probability that customer t is hooked by π^g, which is by definition $H_t(\pi^g)$. To complete the approximation factor of two (or one-half), we show that the probability that customer $\tilde{t}$ is hooked by any of the second k_t products in $\tilde{\pi}_t^g$ is smaller than the probability that customer t is hooked by π^g ($H_t(\pi^g)$). To do this, we argue rank-by-rank that the probability that customer $\tilde{t}$ is hooked by product i at rank $k_t + r$ in $\tilde{\pi}_t^g$ is less than or equal to the probability that customer t is hooked by the r^{th} ranked product in π^g. The intuition behind this claim is that when product i is ranked after the top $r-1$ products in π^g, it cannot hook more customers than the product fixed by our algorithm at rank r since our greedy algorithm sequentially adds products with the maximum hook probability.[8] Formally proving this argument requires some additional machinery to account for the correlations among clicks in π^* and π^g due to the bias functions, which we present in Lemma 1 in the Appendix.

The following example shows the $\frac{1}{2}$-approximation factor in Theorem 1 is tight for Algorithm 1.

Example 1. Consider an instance of OffAR with two products $\{1, 2\}$ and $f_{it}(\chi) = 0$ for all i, t and sets $\chi \subseteq [n]$ The distribution $\mathcal{D}$ is defined as follows:

- with probability $0.5 + \epsilon$, customer t has $k_t = 2$ and $p_{1t} = 1$, $p_{2t} = 0$;
- with probability $0.5 - \epsilon$, customer t has $k_t = 1$ and $p_{1t} = 0$, $p_{2t} = 1$.

[8] The claim is trivially true when product i is also ranked within the top $r-1$ products in π^g.

The optimal ranking is $\pi^* = \{2, 1\}$, which hooks all customers with probability one. On the other hand, $\pi^g = \{1, 2\}$, which only hooks an incoming customer with probability $0.5 + \epsilon$. As $\epsilon \to 0$, the approximation factor for the Greedy Algorithm for OffAR approaches one-half. □

Interestingly, although the $\frac{1}{2}$-approximation factor is tight for general OffAR instances, the following result shows that the Greedy Algorithm for OffAR can achieve a $(1 - \frac{1}{e})$-approximation factor for a broad class of special cases to OffAR, including the maximum coverage problem.

THEOREM 2. *Consider the special case of OffAR where $\boldsymbol{p_t}$, $\boldsymbol{f}_t$, and k_t are all pairwise independent, referred to as OffAR with Independence. The Greedy Algorithm for OffAR achieves a $(1 - \frac{1}{e})$-approximation factor for OffAR with Independence.*

The proof involves showing that for any given first impression window k, the first k ranks in π^g are identical when (i) all customers are included when running the algorithm, and (ii) only customers with first impression window k are included when running the algorithm. This simple yet powerful result follows from the fact that the click probabilities and first impression windows are independent of each other. Consequently, the probability that an incoming customer first clicks on a product i when it is placed at some position $j \leq k$ remains the same when conditioned on each possible value of the first impression window $k_t \geq j$. Therefore, both versions of the greedy algorithm mentioned above would select the same product at rank j since assuming a specific value for the first impression window does not alter the probability that a customer is hooked. Using results from stochastic submodular optimization, we can then show that the latter ranking is a $(1 - \frac{1}{e})-$approximation to the optimal ranking for customers with first impression window k. Finally, we relate the hook probability of the optimal ranking for all customers to the optimal ranking for customers with first impression window k for all $k \in [n]$ via a decomposition argument.

For the special case of OffAR where $p_{it} \in \{0, 1\}$, $f_{it}(\chi) = 0$ for all i, t, and allowing for dependent $\boldsymbol{p_t}$ and k_t, one can develop an alternative algorithm using LP-pipage rounding that also achieves a $(1 - \frac{1}{e})-$approximation factor (Asadpour et al. 2020). Intuitively, if $\boldsymbol{p}_t$ and k_t are correlated, there may be more opportunity to prioritize hooking different customer types due to varying k_t. Our algorithm does not account for such dependencies, whereas work by Asadpour et al. (2020) does. Unfortunately, their algorithm does not extend to the learning setting in OnAR.

4. Online Assortment Ranking

In this section, we study the more realistic *online* assortment ranking problem (OnAR), where the retailer does not know the distribution $\mathcal{D}$ from which customers' preferences and first impression windows are drawn. A naive approach for tackling this problem is to learn the complete distribution $\mathcal{D}$ by sampling from a sufficiently large number of customers. One could then apply (say) Algorithm 1 for OffAR to retrieve the same approximation factors as in Section 3. Unfortunately, such an approach is likely infeasible as the joint distribution $\mathcal{D}$ may have a large support, whereas the number of customers (available samples) may be relatively small. The problem is further compounded by the fact that the retailer's observations are *censored*: the retailer cannot observe the exact first impression window k_t of customers nor infer the the full vectors $\boldsymbol{p}_t$ and $\boldsymbol{f}_t$.

Driven by these constraints, we develop learning-then-earning algorithms for OnAR that converge to the offline approximately optimal ranking *without learning the distribution* $\mathcal{D}$. Our learning-then-earning algorithms operate in two phases: first, in the learning phase, the algorithm offers different rankings to customers in order to learn the (approximately) best ranking. Then, in the earning phase, the algorithm offers this single, (approximately) best ranking to the remaining customers. Although our proposed methods do sample customers to partially infer their preferences, the goal is to simply learn the (approximately) best ranking of products, not the entire distribution $\mathcal{D}$. As is typical for online learning algorithms, we measure performance in terms of two metrics: (i) an approximation factor with respect to the optimal ranking for the earning phase, and (ii) the length of the learning phase, i.e. the number of customers for which the algorithm learns before converging upon the final ranking. The first metric essentially measures how accurately the algorithm can learn, whereas the second metric measures the speed of learning.

We present two learning algorithms for OnAR: the *Simple Learning Algorithm* and the *Threshold Acceptance Algorithm*. The former leverages multi-armed bandit (MAB) techniques to mimic the Greedy Algorithm for OffAR. These techniques come at the cost of a long learning phase, leading to the question of whether the retailer could sacrifice some performance in favor of speed. With

this motivation, we develop the Threshold Acceptance Algorithm and prove that it achieves an approximation guarantee that is close to that of the Greedy Algorithm for OffAR but with a convergence rate that is an order of magnitude faster than that of the Simple Learning Algorithm.

4.1. Simple Learning Algorithm

We begin with a sequential learning paradigm that yields a class of algorithms for OnAR all of which converge to a one-half approximation to the optimal ranking, yet differ by the length of the learning phase. The class of algorithms that we propose builds upon previous work in the area of multi-armed bandits for the problem of identifying the best arm among a finite set, i.e., the action yielding the highest expected reward. We propose a general procedure that transforms such MAB algorithms into algorithms that compute rankings for OnAR. We use a MAB algorithm as a black-box input and make several calls to this black-box to fix an approximately optimal product at each rank. Our reliance on black-box transformations using MAB algorithms is inspired by the large body of recent research that focuses on computing the best arm using the minimum number of samples (see Jamieson and Nowak (2014) for a survey); this version of the MAB problem is often referred to as MAB for Best Arm Identification, although we will simply use MAB for brevity.

We first describe a few technical concepts required for stating our results. A multi-armed bandit problem comprises of n arms or actions such that upon 'pulling each arm' i, the algorithm observes a reward of ν_i that is drawn independently from some unknown distribution. There is a finite limit, $\tilde{T}$, on the total pulls that can be made. We focus on the objective of identifying the arm that maximizes the expected reward ($i^* = \arg\max_i E[\nu_i]$) while minimizing the number of arm pulls. As is common in online learning, we use the framework of (ϵ, δ)-PAC (probably approximately correct) algorithms that compute an approximately-optimal arm with high probability.

Definition 1. (Even-Dar et al. 2006) An algorithm is an (ϵ, δ)-PAC algorithm for the multi-armed bandit problem with sample complexity $\tilde{T}$ if with probability at least 1 - δ, it outputs an arm i such that $E[\nu_i] \geq \max_{i'} E[\nu_{i'}] - \epsilon$ and the number of times it pulls the arms is bounded by $\tilde{T}$.

Algorithm 2 presents our Simple Learning Algorithm for OnAR, which specifies rankings offered to customers during the learning phase. The input to Algorithm 2 includes an (ϵ', δ')-PAC algorithm, ALG, as a black-box. For any given choice of the parameters ϵ', δ', such an algorithm takes

a collection of products (arms), draws a finite number of samples from the specified reward distribution associated with each arm, and returns with probability at least $1-\delta'$, a product that maximizes the expected reward up to an additive error of ϵ'. We discuss two potential choices for the black-box algorithm *ALG* after Corollary 1.

The Simple Learning Algorithm proceeds sequentially from ranks one through n and for each rank r, it invokes *ALG* as a black-box algorithm to identify the product with the maximum marginal benefit to fix at rank r. Formally, the inputs to *ALG* at rank r comprise of the products that have not been fixed at previous ranks, with the reward of each arm i equal to its marginal benefit at the current rank defined as a Bernoulli random variable equal to one if the customer clicked on product i in rank r but not on any product in ranks $[1, r-1]$, and zero otherwise. In turn, *ALG* learns the marginal benefit of each product by observing the click actions of at most $\tilde{T}$ customers; each customer is equivalent to a single sample (arm pull) in Definition 1. The product returned by *ALG* (with probability at least $1-\delta'$) hooks the same fraction of customers as the maximum marginal benefit product minus an additive error of ϵ', and as such, is *irrevocably* assigned to rank r. This process is then continued for subsequent ranks until the end of the learning phase, at which time the ranking output by Algorithm 2 is offered to all remaining customers. We note that the Simple Learning Algorithm mimics the Greedy Algorithm for OffAR since the product with the approximately-maximum marginal probability of hooking a customer is fixed at each rank with high probability.

The following result shows that the ranking produced by the Simple Learning Algorithm hooks at least (approximately) one-half of the customers hooked by the optimal ranking with high probability for general OnAR. Analogous to the offline setting, we define OnAR with Independence to be the special case of OnAR where $\boldsymbol{p}_t$, $\boldsymbol{f}_t$, and k_t are pairwise independent; in this case, the one-half approximation factor can be further tightened to $1-\frac{1}{e}$.

THEOREM 3. *Given any (ϵ', δ')-PAC algorithm as a black box where $\epsilon' = \frac{2\epsilon}{n}$ and $\delta' = \frac{\delta}{n}$, the Simple Learning Algorithm returns a ranking $\tilde{\pi}$ such that with probability $(1-\delta)$:*

Algorithm 2: Simple (Black-Box) Learning Algorithm

Input: (ϵ', δ')-PAC algorithm ALG, Parameters $\epsilon, \delta > 0$;

Initialize null ranking $\tilde{\pi}(i) = \emptyset$ for all $1 \leq i \leq n$, and initialize unranked products $U = [n]$;

for $r = 1$ *to* n **do**

For all $i \in U$, define $\tilde{\pi} \cup i$ to be a ranking identical to $\tilde{\pi}$ in positions $1, ..., r-1$ and with product i in rank r;

Run ALG on products U for $\tilde{T}$ customers, where $\nu_i = H_t(\tilde{\pi} \cup i) - H_t(\tilde{\pi})$ for all $i \in U$;

Let $\tilde{i}_r$ be the product output by ALG;

Set $\tilde{\pi}(\tilde{i}_r) \leftarrow r$;

$U \leftarrow U \setminus \tilde{i}_r$.

end

1. $\mathbb{E}[H_t(\tilde{\pi})] \geq \frac{1}{2}\mathbb{E}[H_t(\pi^*)] - \epsilon$ *for general OnAR.*
2. $\mathbb{E}[H_t(\tilde{\pi})] \geq (1 - \frac{1}{e})\mathbb{E}[H_t(\pi^*)] - \epsilon$ *for OnAR with Independence.*

Our final ranking $\tilde{\pi}$ is essentially an (ϵ, δ)-PAC approximation to the optimal ranking π^*. Since $\tilde{\pi}$ achieves an additive error of ϵ compared to the benchmark, the maximum permissible error for ALG at each individual rank r is $\frac{\epsilon}{n}$ (multiplied by a factor of two); similarly, for each rank r, the product designated at that rank via ALG is suboptimal with probability at most $\frac{\delta}{n}$. Given the lower bounds and the hardness results for OffAR in Section 3, it is not possible for any online learning algorithm to identify the optimal ranking using a polynomial number of samples.

The Simple Learning Algorithm makes exactly n black-box calls to the (ϵ', δ')-PAC algorithm. This allows us to bound the length of the learning phase as follows.

Corollary 1. *Given any (ϵ', δ')-PAC algorithm as per Definition 1 as a black box, the Simple Learning Algorithm returns a ranking $\tilde{\pi}$ after at most $n\tilde{T}$ customers.*

Since an (ϵ, δ)-PAC algorithm is utilized as a black-box, an implementation of the Simple Learning Algorithm could invoke any of numerous learning algorithms (Even-Dar et al. 2006, Jamieson

and Nowak 2014, Kalyanakrishnan et al. 2012) for best-arm identification to develop a corresponding methodology for ranking. Note that all of these algorithms converge to a ranking with the same approximation guarantee. However, the total number of customers they require may depend on different parameters and thus the choice of the best algorithm is instance-specific. The following corollary highlights the number of customers used by our algorithm for two specific instantiations of the black-box (ϵ', δ')-PAC algorithm. In particular, we consider the *Exhaustive Sampling Algorithm* (Even-Dar et al. 2006), where each product is sampled $L = \frac{4}{\epsilon^2}\log(\frac{2n}{\delta})$ times and the product yielding the largest number of hooked customers is returned; note that $\tilde{T} \leq nL$ in this case. We also consider the *LUCB Algorithm* (Kalyanakrishnan et al. 2012), a variant of the popular upper confidence bound (UCB) algorithm.

COROLLARY 2. *The Simple Learning Algorithm returns a ranking $\tilde{\pi}$ with approximation factors as specified in Theorem 3 with probability $(1-\delta)$ and length of learning phase as described below:*

1. $\Theta\big(\frac{n^4}{\epsilon^2}\log(\frac{n^2}{\delta})\big)$ *when the Exhaustive Sampling Algorithm is used as the black box.*
2. $\Theta\big(\frac{n^2}{\Delta_*^2}\log(\frac{n^2}{\Delta_*^2\delta})\big)$ *when the LUCB Algorithm is used as the black box, where Δ_* is a measure*[9] *of the minimum gap between the rewards of the best and second-best arms, where the minima is taken over all ranks $r \in [n]$.*

It is worth noting that the Ranked Explore and Commit Algorithm in Radlinski et al. (2008) is identical to our Simple Learning Algorithm with Exhaustive Sampling for the special case of $f_{it}(\chi) = 0$ and $k_t = k$ for all i, t. Both papers show that the algorithm (approximately) achieves the $(1 - \frac{1}{e})$-approximation factor to the optimal offline solution; note that our result follows from Theorem 3.2, since $f_{it}(\chi) = 0$ and $k_t = k$ for all i, t is a special case of OnAR with Independence. However, due to our generalizations - namely, (i) our Simple Learning Algorithm can use any black-box (ϵ', δ')-PAC algorithm, not just the Exhaustive Sampling Algorithm, (ii) our bias function

[9] Technically, let $[\tilde{\pi}]_r$ denote the sub-ranking of $\tilde{\pi}$ with only the first r ranks filled and $[\tilde{\pi}]_{r-1} \cup i$ be the ranking that coincides with $\tilde{\pi}$ in its first $r-1$ positions and has product i in its r^{th} rank. For any r let U_r be the set of products that are not ranked in the first $r-1$ positions in $\tilde{\pi}$. We define $\Delta_* = \min_{r=1}^{n} \min_{i,i' \in U_r} \left(H_t([\tilde{\pi}]_{r-1} \cup i) - H_t([\tilde{\pi}]_{r-1} \cup i')\right)$.

allows for click probabilities to depend on the products previously clicked on, and (*iii*) our first impression windows vary across customers - the techniques required to prove our more general results are different. For example, the arguments in Radlinski et al. (2008) do not apply to our model as the total value of a solution cannot be decomposed into marginal probabilities if we rearrange the order in which products are presented. Instead, our proof relies on a position-by-position comparison of the marginal probability that a product hooks a customer in the optimum and greedy rankings. This in turn requires carefully handling the correlation between the products that customers click on in different rankings, which we formalize in Lemma 1 in the Appendix.

To summarize, the Simple Learning Algorithm balances popularity and diversity and (approximately) recovers the performance guarantees achieved in the offline setting for the earning phase. The algorithm requires as many as $\Theta(n^4)$ customers in the learning phase before converging to an approximately-optimal ranking for the earning phase. Requiring such a long learning phase may be infeasible; thus, the next subsection is devoted to modifying the Simple Learning Algorithm to decrease the length of the learning phase at a slight loss of optimality in the earning phase.

4.2. Threshold Acceptance Algorithm

We propose a novel method that we term the *Threshold Acceptance Algorithm*, which achieves a significant speed-up in convergence time at the cost of only a small loss in performance. The approach is fully described in Algorithm 3 but its core idea is rather simple: *instead of exhaustively trying all products at a given rank in order to identify the best one, the algorithm simply selects a product that is 'good enough'*. Although this appears to be a simple tweak, the convergence time is an order of magnitude faster.

At each stage of the algorithm, we maintain a threshold τ, which is monotonically decreasing. The algorithm terminates when the threshold goes below a pre-specified value $\tau^{\min}$. Any product whose marginal benefit exceeds the threshold is fixed at the current rank and we move on to the next rank. Unlike the Simple Learning Algorithm, the Threshold Acceptance Algorithm allows us to fix products at consecutive ranks in a single pass through the products: this is the primary

Algorithm 3: Threshold Acceptance Algorithm

Input: Parameters $\epsilon, \delta, \alpha > 0$, Initial and Final Thresholds: $\tau^{\max} \geq \tau^{\min} > 0$;

Initialize null ranking $\tilde{\pi}(i) = \emptyset$ for all $1 \leq i \leq n$, and initialize $\tau = \tau^{\max}$;

Initialize unranked products $U = [n]$, $CurrentRank = 1$;

while $\tau \geq \tau^{\min}$ *and* $CurrentRank \leq n$ **do**

  for $i \in U$ **do**

    Define $\tilde{\pi} \cup i$ to be a ranking identical to $\tilde{\pi}$ in positions $1, ..., r-1$ and with product i in r, where $r = CurrentRank$;

    Display ranking $\tilde{\pi} \cup i$ to $L = \frac{n^2}{\epsilon^2} \log(\frac{n}{\sqrt{\delta}})$ customers;

    Let Δ_{ir} be the fraction of the above customers who only click on i;

    if $\Delta_{ir} \geq \tau$ **then**

      Set $\tilde{\pi}(i) \leftarrow CurrentRank$;

      $U \leftarrow U \setminus i$, $CurrentRank \leftarrow CurrentRank + 1$;

    end

  end

  $\tau \leftarrow \frac{\tau}{1+\alpha}$;

end

cause of the speed-up. Once the algorithm takes a full pass through the available products, the threshold is then lowered geometrically (i.e., from τ to $\frac{\tau}{1+\alpha}$) and thus the algorithm terminates after a logarithmic number of rounds. The choice of α allows the algorithm designer to trade off between performance and speed: small (large) α leads to more (less) conservative thresholds which result in a longer (shorter) learning phase yet better (worse) performance in the earning phase. Finally, in some cases, the ranking $\tilde{\pi}$ output by the Threshold Acceptance Algorithm may contain fewer than n products assigned to ranks. In this case, one can simply append the remaining products to $\tilde{\pi}$ (e.g., according to estimated click probabilities) without affecting the theoretical bounds.

THEOREM 4. *The Threshold Acceptance Algorithm with* $\tau^{\max} = 1, \tau^{\min} = \frac{\epsilon}{n}$ *returns a ranking* $\tilde{\pi}$ *such that with probability* $1 - \delta$

$$\mathbb{E}[H_t(\tilde{\pi})] \geq \frac{1}{2+\alpha}\mathbb{E}[H_t(\pi^*)] - \epsilon,$$

where π^* *is the optimal solution to OffAR. The length of the learning phase is* $O\left(\frac{n^3}{\epsilon^2}\log_{1+\alpha}(\frac{n}{\epsilon})\log(\frac{n}{\sqrt{\delta}})\right)$.

The key idea behind the proof is two-fold: first, the product fixed by our algorithm at position r in the ranking $\tilde{\pi}$ has an empirical marginal benefit of at least τ. Due to our choice of L, its actual expected marginal benefit can be lower bounded by $\tau - \frac{\epsilon}{n}$. Second, we bound the loss due to only selecting a product that is 'good enough' by recognizing that no product can provide more marginal benefit than $(1+\alpha)\tau$, due to how the threshold is decreased at each iteration of the algorithm. The rest of the proof involves summing up the above ideas over all ranks and deriving a relationship between the marginal benefit of adding a product at rank r in $\tilde{\pi}$ vs. π^*.

Note that the length of the learning phase ($\Theta(n^4)$) for the Simple Learning Algorithm with Exhaustive Sampling is a strict and instance-agnostic limit, whereas the Threshold Acceptance Algorithm's learning phase length of $O(n^3\log(n))$ is only a worst-case bound: in practice, the actual click probabilities tend to be closely clustered in which case the learning phase is significantly shorter. Furthermore, we note that in both algorithms, *the performance improves substantially throughout the learning phase*, since products are fixed in ranks iteratively and the initial ranks are responsible for hooking the most customers due to both (i) expiring first impression windows, and (ii) customers' increased likelihood of being hooked by an earlier-ranked product.

Additional Considerations There are a few practical considerations worth noting regarding both our algorithms and model. First, although our algorithms specify which products to rank only in positions $[1, r]$ for iteration r, in practice the retailer can simply append the remaining products (e.g., according to estimated click probabilities) without affecting the theoretical bounds. Second, the ranking returned by our algorithms would not be affected even if the retailer only

observes the first product that a customer clicks on. This has key implications: for one, it implies that companies only need to track a customer's first click, rather than the entire set of products the customer clicks on. In addition, it highlights that our algorithms are robust to situations where customers change their search paths and click behavior after clicking on the first product.

4.3. Regret Analysis

The performance of our learning algorithms was stated in terms of two metrics commonly used to evaluate learning-then-earning algorithms, namely the approximation to the optimal ranking and the speed of convergence. Breaking down the performance in such a manner highlights the tradeoffs inherently involved, e.g., the Threshold Acceptance Algorithm requires much fewer customers than the Simple Learning Algorithm to converge, but its approximation factor is slightly worse. That said, all of our results in this section could be stated in terms of the underlying regret bound by assuming that the retailer incurs unit regret during the learning phase and fixed per-round regret in the earning phase.

Formally, the regret incurred by a learning policy that selects ranking π_t for customer t benchmarked against a static ranking π^s can be written as $\mathbb{E}[R_{\pi^s}] = \sum_{t=1}^{T} \left(\mathbb{E}[H_t(\pi^s)] - \mathbb{E}[H_t(\pi_t)]\right)$. First, consider the Simple Learning Algorithm with Exhaustive Sampling as described in Corollary 2. Assuming that the total number of customers exceeds the length of the learning phase, i.e., $T > \frac{n^4}{\epsilon^2}\log(\frac{2n^2}{\delta})$, its regret compared to an arbitrary static ranking π^s can be expressed as:

$$\mathbb{E}[R_{\pi^s}^{SIMPLE}] \leq \left(\frac{n^4}{\epsilon^2}\log(\frac{2n^2}{\delta}) + \epsilon T + \delta T\right) + T\left(\mathbb{E}[H_t(\pi^s)] - \mathbb{E}[H_t(\pi^g)]\right),$$

where π^g is the ranking returned by the Greedy Algorithm for OffAR from Section 3. Substituting $\epsilon^3 = \frac{n^4}{T}$ and $\delta = \frac{1}{T}$, we get the following corollary.

Corollary 3. *The Simple Learning Algorithm (Algorithm 2) with Exhaustive Sampling achieves a regret of at most* $O\left(n^{\frac{4}{3}}T^{\frac{2}{3}}\log(nT)\right) + \mathbb{E}[H_t(\pi^*)] \cdot \frac{1}{2}T$ *compared to the optimal ranking* π^* *and* $O\left(n^{\frac{4}{3}}T^{\frac{2}{3}}\log(nT)\right)$ *compared to the ranking* π^g *returned by the Greedy Algorithm for OffAR.*

The regret incurred by the Threshold Acceptance Algorithm, $R_{\pi^s}^{THRESH}$, can be bound similarly. Let π^T denote the static ranking returned by the offline version of the Threshold Acceptance Algorithm.[10] From Theorem 4, it is not hard to deduce that π^T would be a $\frac{1}{2+\alpha}$-approximation to the optimal ranking. Then, we have that:

$$E\left[R_{\pi^s}^{THRESH}\right] \leq \left(\frac{n^3}{\epsilon^2}\left(1+\log_{1+\alpha}(\frac{n}{\epsilon})\right)\log(\frac{n}{\sqrt{\delta}})+\epsilon T+\delta T\right)+T\left(\mathbb{E}[H_t(\pi^s)]-\mathbb{E}[H_t(\pi^T)]\right)$$

Substituting $\epsilon^3=\frac{n^3}{T}(1+\log_{1+\alpha}(T))$ and $\delta=\frac{1}{T}$, we get:

COROLLARY 4. *The Threshold Acceptance Algorithm (Algorithm 3) achieves a regret of at most* $O\left(nT^{\frac{2}{3}}(\log_{1+\alpha}(T))^{\frac{1}{3}}\log(nT)\right)+\mathbb{E}[H_t(\pi^*)]\frac{1+\alpha}{2+\alpha}T$ *compared to the optimal ranking* π^* *and* $O\left(nT^{\frac{2}{3}}(\log_{1+\alpha}(T))^{\frac{1}{3}}\log(nT)\right)$ *compared to the ranking* π^T *returned by the offline Threshold Acceptance Algorithm.*

We retain the dependence of the regret on α in Corollary 4 to emphasize that the value of this parameter is context-specific and not something that can be tuned for optimality.

Compared to the optimal ranking π^*, our regret comprises of a sublinear parameter whose derivative approaches zero as $T\to\infty$ as well as a term that scales linearly in T. Given that the offline version of this problem (OffAR) is NP-Hard, this linear term is unavoidable since *any approximation algorithm* would lead to a constant per-round regret even if the underlying distribution $\mathcal{D}$ was known in advance.

5. Simulations

In this section, we describe our empirical setting and estimate the impact of our Simple Learning Algorithm and Threshold Acceptance Algorithm on actual clickstream data.

5.1. Company and Data Description

We partnered with Wayfair to develop our algorithms and estimate their impact using clickstream data. Although a majority of Wayfair's business is driven via intentional search through product

[10] As in Section 3, the offline version of this algorithm can be constructed by directly computing the value of ranking $\tilde{\pi}\cup i$ in Algorithm 3 instead of evaluating its sample average by displaying this ranking to L customers.

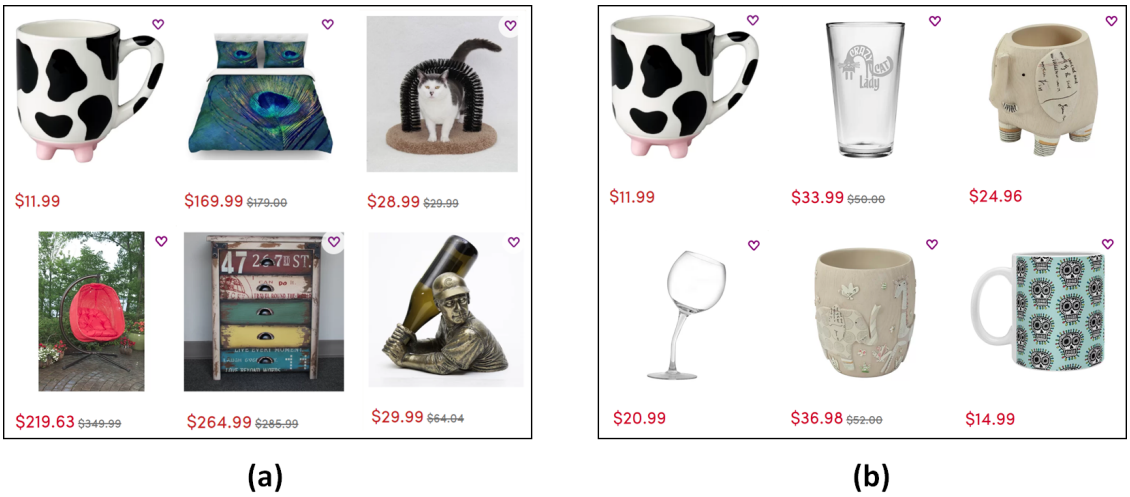

Figure 1 **Chosen from a single Wayfair event, (a) shows six diverse products with respect to style, price point, product category, color, etc. and (b) shows six more similar products across these dimensions.**

category pages (e.g. a customer searching for a new bed), a significant amount of sales and traffic is through Wayfair's "events". These events are often stylistically themed and offer products from multiple product categories; typical events last about 2 weeks and offer approximately 100 products. Figure 1 shows products chosen from a single Wayfair event; panel (a) highlights the diversity of products offered with respect to style, price point, product category, color, etc., whereas panel (b) shows six products that are more similar. A customer with a first impression window of six products would likely have a very different perception of the full assortment of products offered in the event if she were to be shown the products in (a) vs. (b) in the first six ranks.[11]

Wayfair provides three options for customers to sort items in events: recommended (default), customer rating, and price. For some events, Wayfair may offer a refinement bar to allow customers to filter the products to only those they may be interested in. Less than 5% of customers change the default ranking or use filters, supporting our consumer behavior model that most customers are likely engaging in hedonic browsing as opposed to intentional search. We chose to not include customers who change the ranking or use filtering in our simulations since our algorithms are

[11] Products are ranked left to right, then top to bottom, such that the first row in the figure displays ranks 1, 2, and 3, and the second row displays ranks 4, 5, and 6.

Event	A	B	C	D	E	F
Number of product categories offered	29	12	25	15	18	5

Table 2 Number of Product Categories Offered in Each Event

intended to replace Wayfair's default ranking and therefore would only be applied to browsing customers. Wayfair's default ranking is a static ranking (does not change throughout the event) which ranks products in decreasing order of their expected popularity based on pre-event data.

Wayfair selected six historical, stylistically-themed events (e.g., Halloween-themed decor) for which to evaluate our algorithms. The total number of customers ($|\mathcal{T}|$) who shopped at each event ranged between 100,000 and 325,000; due to confidentiality, we disguise events as A, B, C, D, E, and F in increasing order of $|\mathcal{T}|$. Table 2 displays the number of product categories that were offered in each event; each product category represents a set of products that Wayfair's merchandising team considers to be substitutable, at least from a functional perspective (e.g., floor lamps and ceiling lights could constitute two separate product categories in one of the events). The large number of product categories offered in most events illustrates the functional diversity[12] of the offered product set. For each of these events, we were provided clickstream data which includes every product each customer clicked on in the event. We represent this data as $c_{it} = 1$ if customer t clicked on product i and 0 otherwise. We focus only on the first $n = 48$ products in the event since this is the number of products shown per page before having to click to view the second page of products. As per the data, very few customers advance to the second page.

5.2. Implementation

First, we describe how we generate customers endowed with $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$ using the clickstream data provided. Then, we describe the implementation details of the algorithms.

5.2.1. Customer Generation

Since $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$ is unobserved, we must infer it from the clickstream data. To do so, we make a few simplifying assumptions and then adopt a general maximum likelihood methodology to estimate

[12] We note that our definition of diversity extends beyond simply functional use, but nonetheless this is a reasonable proxy for one dimension of product diversity within an event.

the click probabilities and first impression windows subject to these assumptions. First, we assume interest probabilities are independent of first impression windows (OnAR with Independence). Second, we assume that if a customer is hooked, she views the remaining products on the page (i.e., she views the first 48 products in the event). This assumption is very conservative in that it handicaps our algorithms' performance because for hooked customer t, this assumption results in inferring $p_{it} = 0$ for all products i that the customer did not click on; in reality, the customer may not have clicked on a product i simply because she did not view it and p_{it} could have been non-zero. This assumption gives our algorithms less opportunities to hook customers, and thus the performance of our algorithms is likely a lower bound on its potential real-world performance. Third, we assume that $p_{it} \in \{0, 1\}$ and $f_{it}(\chi) = 0 \ \forall i, t$ where $p_{it} = 1$ implies that customer t will click on product i if she views it and $f_{it}(\chi) = 0$ implies that previous clicks do not affect a customer's propensity to click on later products. Finally, we make assumptions on the distribution that customers' first impression windows follow, and on the number of customers who are completely uninterested in the assortment; we discuss these assumptions along with our methodology. In Appendix E, we present additional simulations that were conducted with different assumptions.

Next we describe our inference methodology following general maximum likelihood principles. First we fix the distribution of first impression windows $\mathcal{D}_k$ and the total number of customers who have non-zero preferences, $|\tilde{\mathcal{T}}|$ (i.e., $\sum_i p_{it} \geq 1$ for $t \in \tilde{\mathcal{T}}$). Given $\mathcal{D}_k$ and $|\tilde{\mathcal{T}}|$, we will infer the distribution of interest probabilities, $\mathcal{D}_p$. Subsequently, we will discuss our choices of $\mathcal{D}_k$ and $|\tilde{\mathcal{T}}|$. Let $\tilde{\mathcal{T}}_c$ be the subset of customers from $\tilde{\mathcal{T}}$ who clicked on a product, i.e. $\tilde{\mathcal{T}}_c \triangleq \{t \in \tilde{\mathcal{T}} : \sum_i c_{it} \geq 1\}$. Note that $\tilde{\mathcal{T}}_c \subseteq \tilde{\mathcal{T}} \subseteq \mathcal{T}$ and $p_{it} = c_{it} \ \forall t \in \tilde{\mathcal{T}}_c$ and $i \in [n]$; the challenge is that we do not observe $\boldsymbol{p}_t$ for $t \in \tilde{\mathcal{T}} \setminus \tilde{\mathcal{T}}_c$, when $\boldsymbol{c}_t = \boldsymbol{0}$.

1. *Derive distribution of first impression windows for customers $t \in \tilde{\mathcal{T}}_c$:* We define r_t^1 to be the index of the first product customer t clicks on and let i be the product located at rank r_t^1. We have $c_{it} = 1$ and $c_{i't} = 0$ for any product i' placed at a rank smaller than r_t^1. Since the customer clicked on the r_t^1–th ranked product, her first impression window k_t must be greater than or equal to r_t^1. That

is, for $r < r_t^1$, $Pr(k_t = r | t \in \tilde{\mathcal{T}}_c, r_t^1) = 0$. Therefore, for $r \geq r_t^1, Pr(k_t = r | t \in \tilde{\mathcal{T}}_c, r_t^1) = \frac{Pr(k_t=r)}{\sum_{l=r_t^1}^{n} Pr(k_t=l)}$, where $Pr(k_t = l)$ comes from the distribution of first impression windows, $\mathcal{D}_k$.

2. *Weight each observed preference vector:* For each customer $t \in \tilde{\mathcal{T}}_c$, assign weight $w_t = \frac{1}{\sum_{l=r_t^1}^{n} Pr(k_t=l | t \in \tilde{\mathcal{T}}_c)}$ to preference vector $\boldsymbol{p}_t$. Intuitively, this assigns larger weights to preference vectors whose first clicks are in greater ranks, which accounts for the fact that those preference vectors are less likely to be observed in the click data due to expiring first impression windows. This gives rise to a distribution $\mathcal{D}_p$ with a set of preference vectors and a weight on each vector that is proportional to the probability of a customer having the corresponding preferences.

3. Sample $|\tilde{\mathcal{T}}|$ preference vectors from the empirical distribution of $\mathcal{D}_p$, and compare the fraction of sampled customers who are hooked by Wayfair's ranking against the fraction in the actual data.

Now that we have outlined our approach of estimating $\mathcal{D}_p$ given $\mathcal{D}_k$ and $|\tilde{\mathcal{T}}|$, we discuss our choices of $\mathcal{D}_k$ and $|\tilde{\mathcal{T}}|$. For $\mathcal{D}_k$, we assumed 5% of customers viewed all n products on the page ($k_t = n$) and the remaining customers' first impression windows followed a power law distribution where $Pr(k = r) = 0.95 \frac{ar^{-b}}{\sum_{l=1}^{n-1} al^{-b}}$ $\forall r = 1, ..., n-1$. Power law distributions are well-motivated in the context of consumer behavior and product popularity on the Internet (Clauset et al. 2009). We set $|\tilde{\mathcal{T}}| = 0.8|\mathcal{T}|$ based on the parameter choices given to us by our partners at Wayfair. We conducted sensitivity analysis on these parameter choices to better understand their impact on the algorithms' performance; our analysis gleans similar results so we relegate it to Appendix C.

We tuned parameters a and b of the power law distribution such that the resultant $\mathcal{D}_k$ and $\mathcal{D}_p$ most closely matched the observed data: after generating $|\tilde{\mathcal{T}}|$ customers per the procedure outlined above, we randomly assigned each customer a first impression window according to $\mathcal{D}_k$. We then identified which customers would have been hooked had they been presented with Wayfair's actual ranking in order to generate hypothetical click vectors, and we compared the total number of actual clicks vs. the total number of hypothetical clicks for each product. Figure 2 presents the actual vs. hypothetical number of clicks, where the hypothetical clicks are averages over 100 simulations in step 3. In 99.5% of our simulations, the percent of customers hooked by Wayfair's ranking

was within $[0.98, 1.02]$ times the actual percent of customers hooked. The close fit illustrates that our customer generation procedure is congruent with the observed data and generates realistic customers that we can use to test our algorithms. In Appendix D, we further examine the predictive power of our customer behavior model as well as others proposed in the literature.

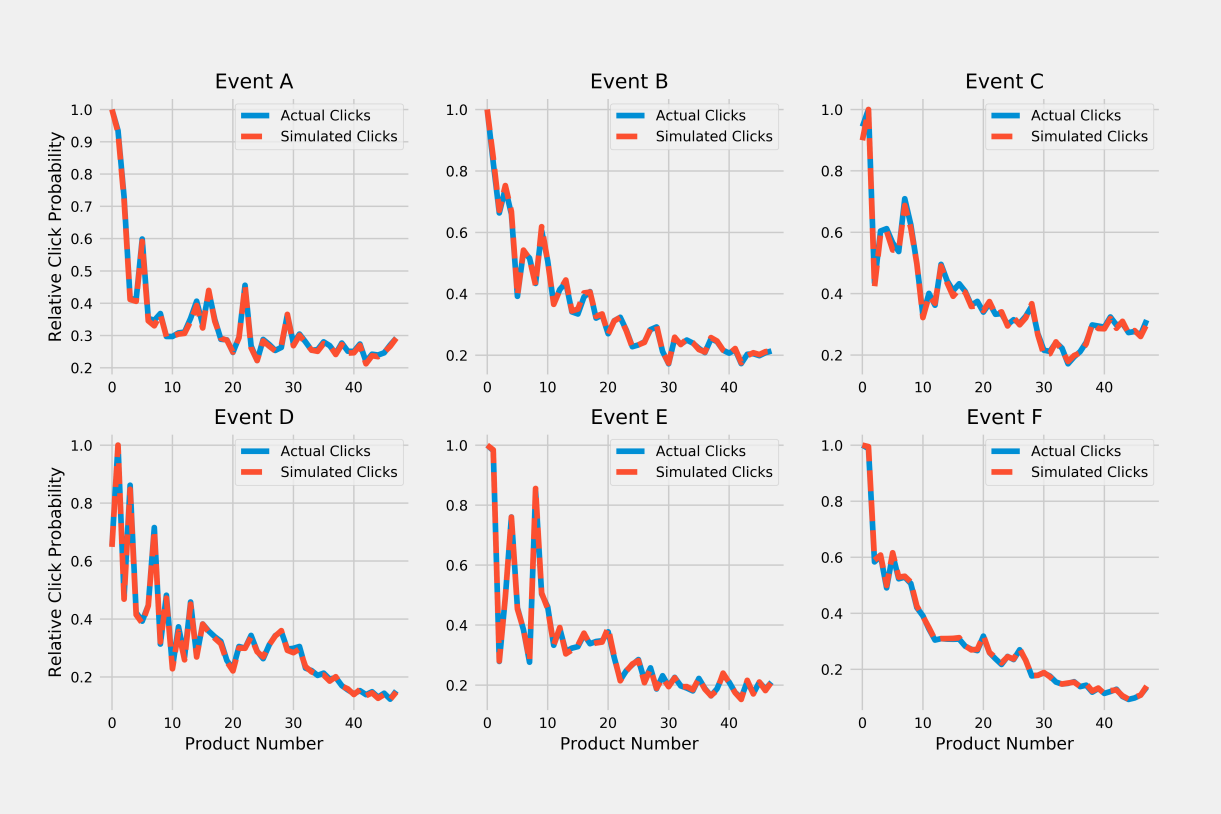

Figure 2 Actual and simulated relative click probabilities, normalized for confidentiality so that the maximum click probability of any product is one. An out-of-sample model comparison can be found in Appendix D.1.

5.2.2. Algorithm Details: Given the consumer behavior model parameters inferred from the data, our goal is to evaluate the performance of our Simple Learning Algorithm (Algorithm 2) with Exhaustive Sampling as a Black-Box as well as our Threshold Acceptance Algorithm (Algorithm 3). We evaluate their performance compared to two static rankings: Wayfair's actual ranking used in the event, and the ranking output by our Greedy Algorithm for OffAR (Algorithm 1). In addition, we also compare their performance to that of two online learning algorithms proposed in the literature for settings most similar to ours: (i) the UCB Algorithm for the Cascading Bandits

problem proposed by Kveton et al. (2015a), and (*ii*) the Ranked Bandits Algorithm with EXP3 proposed by Radlinski et al. (2008). In addition to these papers being closely related to ours in the context of product ranking, we also note that UCB and EXP3 are among the most commonly studied learning algorithms in the literature, and thus make for logical benchmark choices.

For the Threshold Acceptance Algorithm, we chose parameter α to be small and constant across events so that the threshold is lowered conservatively in each iteration, due to click probabilities generally being in a fairly narrow range. We used a sample size of $L = 500$ customers for each ranking offered during the learning phase; we present sensitivity analysis on this modeling choice in Appendix C to evaluate the tradeoff between faster vs. more accurate learning. Finally, the implementation of our algorithm includes two features that were not required to develop the theory yet can make a considerable impact on performance in practice. First, we recognize that the expected number of new customers that can be hooked by product i diminishes with rank due to both expiring first impression windows and because as the rank increases, the customer has a higher probability of getting hooked prior to seeing product i. We can therefore use Δ_{ir} calculated at rank r as an upper bound for $\Delta_{ir'}$ at rank r' for any $r' > r$; when selecting products to test for rank r, we choose products in descending order of their tightest upper bound which allows us to quickly find a product that passes the threshold requirement for fixing in rank r and eliminates having to test clearly suboptimal products for each rank. Second, during the course of the algorithm, as we evaluate the marginal benefit of a product at a given rank, the remaining unranked products are placed in descending order of the upper bound described above. For products not yet tested in any rank, we include them at the end of the ranking in increasing order of their Wayfair rank.

For our Simple Learning Algorithm, we use Exhaustive Sampling as a Black-Box in a manner that ensures maximum consistency with the choices made for the Threshold Acceptance Algorithm. Regarding the other online learning benchmarks, we implement the UCB Algorithm for the Cascading Bandits problem from Kveton et al. (2015a), using a confidence interval of $\sqrt{\frac{2\log(t)}{s_i}}$, where t is the total number of customers encountered so far, and s_i indicates the number of customers who

viewed product i at or before their first click. We also implement the Ranked Bandits Algorithm from Radlinski et al. (2008) and use EXP3 from Auer et al. (2002) as a black-box for selecting the product at each rank. We set the value of the learning parameter γ based on the lower bound provided in Corollary 3.2 in Auer et al. (2002)—i.e., $\gamma = \frac{n\log(n)}{(e-1)g}$, where g is the maximum number of clicks that any one product received in our dataset.

5.3. Results

For each event and set of model parameters, we conducted the following simulation 100 times. First, we generated $|\mathcal{T}|$ customers such that each customer's first impression window was drawn from $\mathcal{D}_k$. With probability $q = \frac{|\mathcal{T}_c|}{|\mathcal{T}|}$, $\boldsymbol{p}_t$ is drawn from $\mathcal{D}_p$; otherwise, $\boldsymbol{p}_t = \mathbf{0}$. We ran parallel copies of each of the learning policies mentioned above as the customers $t = 1, ..., |\mathcal{T}|$ arrived sequentially. For each algorithm, we calculated the percent of customers that were hooked. We also identified the percent of customers that Wayfair's static ranking hooked, which serves as a practical, popularity-based benchmark for which to compare our algorithms. As mentioned previously, due to our choice of $\mathcal{D}_p$ and $\mathcal{D}_k$, this quantity is consistently very close to the actual percent of customers Wayfair hooked.

Figure 3 shows the simulation results for the performance of our Threshold Acceptance and Simple Learning Algorithms, as well as the UCB Algorithm and Ranked Bandits (EXP3) Algorithm benchmarks, compared to Wayfair's static ranking (π^{WF}). In particular, for each online learning algorithm, it displays the percent increase in the number of customers hooked by the algorithm vs. Wayfair's ranking:[13] (total hooked customers by algorithm - total hooked customers by π^{WF}) / (total hooked customers by π^{WF}). Across all six events, both our Threshold Acceptance and Simple Learning Algorithms hook an average of 5-30% more customers than Wayfair's ranking. Interestingly, although theory predicts that the Simple Learning Algorithm would outperform the Threshold Acceptance Algorithm when the number of customers is very large, we find that the performance of the two algorithms are quite similar in practice, highlighting that any loss in optimality incurred by the Threshold Acceptance Algorithm is offset by its shorter learning

[13] Due to confidentiality, we report only relative metrics rather than disclosing the actual percent of hooked customers.

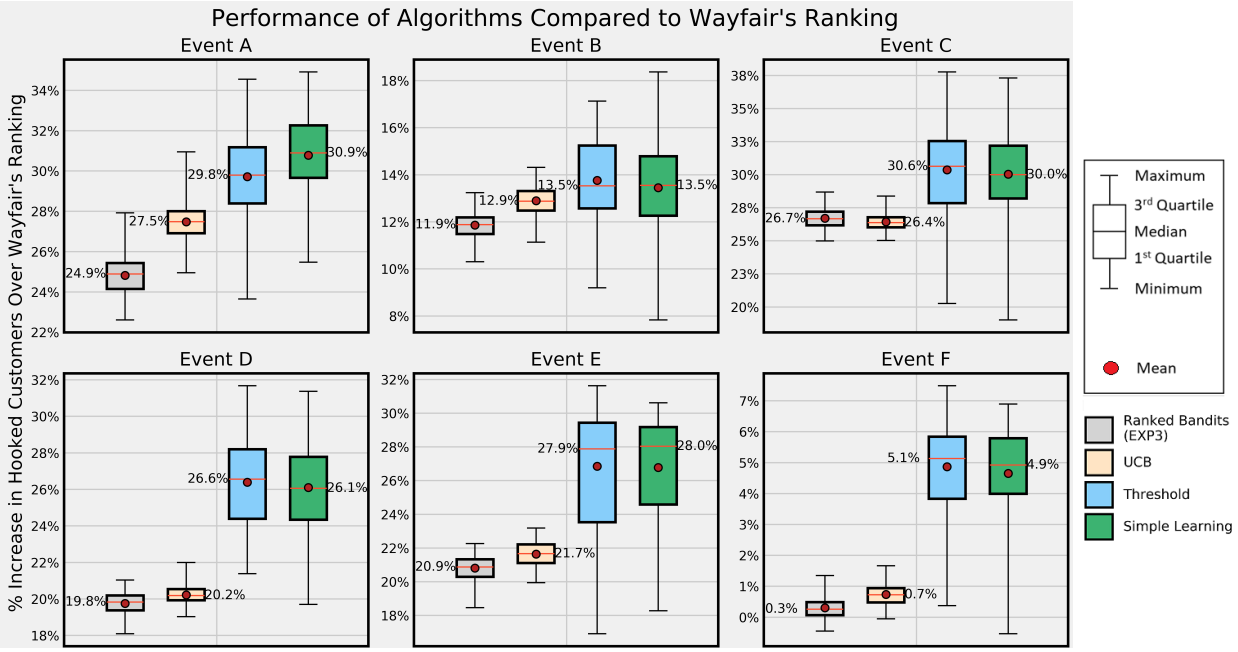

Figure 3 **Box & whisker plots for the percent increase in hooked customers over π^{WF} achieved by the Ranked Bandits, UCB, Threshold Acceptance, and Simple Learning Algorithms.**

phase. We observe that on average, the Threshold Acceptance Algorithm required only 51% of the customers that the Simple Learning Algorithm required for the learning phase, i.e., the Simple Learning Algorithm takes almost twice as long to converge to an approximately-optimal ranking.

Compared to the UCB and Ranked Bandits (EXP3) Algorithms, we observe that our Threshold Acceptance and Simple Learning Algorithms outperform the two benchmarks by a substantial margin for five out of six events. The performance gap partially stems from the fact that both the UCB and Ranked Bandits (EXP3) Algorithms are typically not very effective over small time horizons, $|\mathcal{T}|$. For the UCB Algorithm, this is further compounded by the fact that this policy does not identify (approximately-)optimal rankings for our problem. In fact, this algorithm appears to select solutions that interpolate between popular and diverse rankings, because of which we see an improvement over Wayfair's baseline but not our methods. Although the Ranked Bandits (EXP3) Algorithm takes diversity into account, it still hooks fewer customers than our algorithms, likely owing to its slower convergence.

Looking at Figure 3, there is variability in the performance of our algorithms both across simulations and across events. Variability across simulations primarily stems from a relatively low

number of customers L who are offered each ranking compared to the number of possible preference vectors in $\mathcal{D}_p$; thus, our algorithms may converge to a (slightly) different ranking in each simulation. We note that both our Threshold Acceptance and Simple Learning Algorithms led to a positive increase in the percent of hooked customers for nearly every simulation; this is not trivial - if Wayfair's ranking was optimal, our algorithms would perform worse due to exploring suboptimal rankings before converging.

To better understand the variability in the performance of our algorithms across events, we implemented the Greedy Algorithm for OffAR (Algorithm 1) as an offline benchmark and calculated the percent of customers it would have hooked had it been offered (as a static ranking) to each of the customers; we use this as an upper bound for the number of customers hooked by our algorithms and π^{WF}. Figure 4 compares the performance of the Threshold Acceptance Algorithm[14] and π^{WF} to the static ranking output by the Greedy Algorithm for OffAR (π^g): specifically, (total number of hooked customers by our algorithm (or π^{WF})) / (total number of hooked customers by π^g). Across all six events, our Threshold Acceptance Algorithm has a consistently strong performance, hooking an average of 89-95% of the total number of customers that π^g hooks, which highlights our algorithm's ability to (approximately) converge to and achieve the results in the offline setting. In contrast, there is considerable variability in the performance of Wayfair's ranking across all six events: π^{WF} hooks an average of 68-90% of the total number of customers that π^g hooks. For events where Wayfair's ranking was quite strong (B and F), our algorithm improved on Wayfair's ranking by 5-15%. For events where Wayfair's ranking was not as strong (A, C, D, and E), our algorithm improved on Wayfair's ranking by 26-30%. Interestingly, the events where Wayfair's ranking was already quite strong (B and F) are possibly the two least diverse events, in that they include products from the fewest number of product categories (Table 2). This suggests that our algorithms can be most beneficial to retailers who offer a diverse set of products, as our algorithms take such diversity into account when identifying the best ranking.

[14] The performance of our Simple Learning Algorithm is very similar so we have omitted it for brevity.

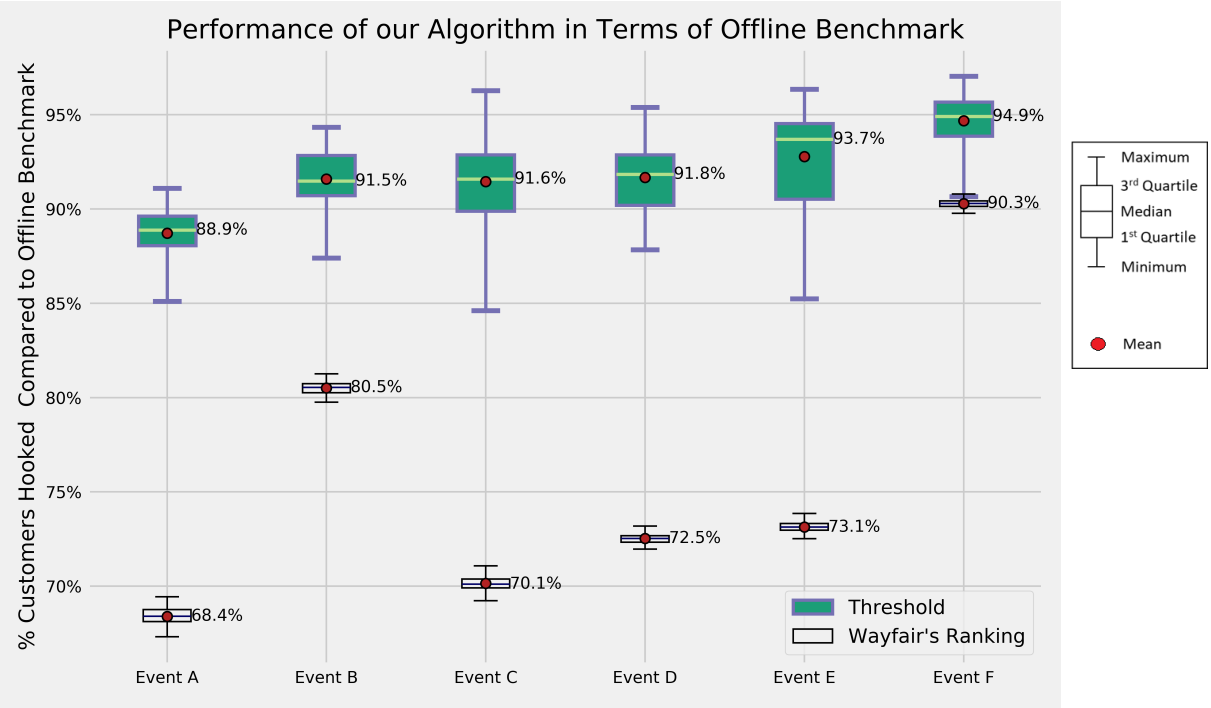

Figure 4 **Box & whisker plots presenting the percent of total customers hooked by π^g (the offline benchmark) that are hooked by our Threshold Acceptance Algorithm and π^{WF}**

REMARK 1. *Estimating Bias Functions.* For the sake of simplicity, we assumed in this section that the bias function $f_{it}(\chi_{it}(\pi))$ is zero for all i,t and $\chi_{it}(\pi)$. It is worth reiterating that this assumption does not in any way affect the performance of our algorithms as both Algorithm 2 and Algorithm 3 do not require estimating $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$. That being said, in situations where click bias is known to be prevalent, it is possible to extend our inference methodology to estimate the bias function under some conditions, should the company want to do so for other purposes. More precisely, suppose that $f_{it}(\chi_{it}(\pi)) = f_i(\chi_{it}(\pi))$ for all t, i.e., the bias function is independent of customer type. Then, one could use the clickstream data to measure $f_i(\chi)$ for any given $\chi \subseteq [n]$ as follows:

$$f_i(\chi) = \mathbb{P}(c_{it} = 1 \mid \mathcal{E}, c_{i't} = 1 \ \forall i' \in \chi) - \mathbb{P}(c_{it} = 1 \mid \mathcal{E}, c_{i't} = 0 \ \forall i' \in \chi), \tag{2}$$

where $\mathcal{E}$ denotes the event that the customer has not clicked on any products ranked ahead of i other than those in χ and that the latter are indeed ranked at positions better than that of i. In simple terms, Equation (2) is the marginal probability of clicking on product i conditional

upon clicking on the products in χ. Unfortunately, the inference method described above may be practically infeasible given the large number of candidate sets χ and the sparsity of clickstream data. Therefore, in practice, retailers may choose some parametric form for the bias function before estimation. For example, under the pairwise product correlation function mentioned in Section 2.1, it is sufficient to solve for $O(n^2)$ parameters in total.

6. Conclusion

We study a crucial problem faced by many online retailers - that of developing *fast online learning approaches* for computing near-optimal product rankings. We propose a novel model for online learning-to-rank problems that captures a number of features unique to online retailers operating in dynamic environments and which have yet to receive attention in the literature. Specifically, we study scenarios where products are not necessarily characterized by a set of attributes and where customers exhibit hedonic browsing behavior. By modeling customers with limited first impression windows, we capture the position effect typically observed in online environments. Yet our framework is more challenging as these first impression windows are unobserved and may be correlated with customers' interest in products. As such, our model is a significant departure from existing work on ranking algorithms.

We present two learning algorithms for the online ranking problem: the Simple Learning Algorithm and the Threshold Acceptance Algorithm. The former uses bandit algorithms for best-arm identification as a black-box to achieve a (nearly) $\frac{1}{2}$-approximation guarantee but may require $O(n^4)$ customer samples in the worst case. The latter algorithm is an order of magnitude faster but its approximation guarantee becomes $\frac{1}{2+\alpha}$ for some parameter $\alpha > 0$. Both of these algorithms require no knowledge of customers' intrinsic utilities, bias functions, or first impression windows, and circumvent having to learn this distribution by directly learning the optimal ranking instead. Another interesting feature of our algorithms is that they optimally balance popularity and diversity, ensuring that the retailer hooks a broad base of customers.

We estimate the impact of our algorithms via simulations using clickstream data from Wayfair. Across six different product assortments, our algorithms hook an average of 5-30% more customers

than Wayfair's static, popularity-based ranking. Furthermore, our algorithms hook 89-95% of the total number of customers hooked by an offline benchmark. These simulations using real clickstream data illustrate that our algorithmic contributions can make a significant impact in practice.

Although our model and results are motivated from the perspective of product ranking, we argue that they may be relevant in any setting with dynamic assortments, products that may not be characterized by a set of attributes, and customers with limited first impression windows. As a concrete example, a digital media outlet could employ our algorithm to highlight 'top articles' - by balancing popularity and diversity, the website could hook a larger fraction of its incoming users. As another example, a retailer could balance popularity and diversity to select the ranking of products to showcase in an email marketing campaign in hopes of maximizing overall click-through rates.

Finally, we share opportunities for future work. First, retailers could benefit from understanding the long-term value of a hooked customer. Although we make the very weak assumption that a hooked customer browses for longer than one who is not hooked, it would be pertinent to understand more details about a hooked customer's browsing, purchasing, and return shopping behavior. Second, "hedonic browsing" is a common customer behavior that many retailers face, yet much of the academic work focuses primarily on customers who engage in "directed buying", i.e. who shop with the intent to purchase a product and have knowledge of what they are looking for (Moe 2003). We hope that our work inspires others to consider studying the impact of hedonic browsing behavior on a firm's operational and marketing decisions.

References

Abeliuk A, Berbeglia G, Cebrian M, Van Hentenryck P (2016) Assortment optimization under a multinomial logit model with position bias and social influence. *4OR* 14(1):57–75.

Agrawal S, Avadhanula V, Goyal V, Zeevi A (2016) A near-optimal exploration-exploitation approach for assortment selection. *Proceedings of the 2016 ACM Conference on Economics and Computation*, 599–600.

Ahmed A, Teo CH, Vishwanathan SVN, Smola AJ (2012) Fair and balanced: learning to present news stories. *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012*, 333–342 (ACM).

Aouad A, Farias VF, Levi R (2015) Assortment optimization under consider-then-choose choice models, working paper.

Aouad A, Feldman J, Segev D, Zhang D (2019) Click-based mnl: Algorithmic frameworks for modeling click data in assortment optimization, working paper.

Aouad A, Segev D (2020) Display optimization for vertically differentiated locations under multinomial logit preferences. *Management Science* Forthcoming.

Arthur D, Vassilvitskii S (2007) k-means++: the advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA)*, 1027–1035.

Asadpour A, Nazerzadeh H (2015) Maximizing stochastic monotone submodular functions. *Management Science* 62(8):2374–2391.

Asadpour A, Niazadeh R, Saberi A, Shameli A (2020) Ranking an assortment of products via sequential submodular optimization, working paper.

Auer P, Cesa-Bianchi N, Freund Y, Schapire RE (2002) The nonstochastic multiarmed bandit problem. *SIAM journal on computing* 32(1):48–77.

Badanidiyuru A, Vondrák J (2014) Fast algorithms for maximizing submodular functions. *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA)*, 1497–1514.

Bernstein F, Modaresi S, Saure D (2018) A dynamic clustering approach to data-driven assortment personalization. *Management Science* 65(5):2095–2115.

Bezos J (1998) Bezos letter to shareholders.

Bloch P, Ridgway N, Sherrell D (1989) Extending the concept of shopping: An investigation of browsing activity. *Journal of the Academy of Marketing Science* 17(1):13–21.

Bucklin R, Sismeiro C (2009) Click here for internet insight: Advances in clickstream data analysis in marketing. *Journal of Interactive Marketing* 23:35–48.

Chen Y, Yao S (2016) Sequential search with refinement: Model and application with click-stream data. *Management Science* 63(12):4345–4365.

Cho E, Youn-Kyung K (2012) The effects of website designs, self-congruity, and flow on behavioral intention. *International Journal of Design* 6(2).

Clauset A, Shalizi CR, Newman MEJ (2009) Power-law distributions in empirical data. *SIAM Review* 51(4):661–703.

Dawson S, Kim M (2010) Cues on apparel web sites that trigger impulse purchases. *Journal of Fashion Marketing and Management: An International Journal* 14(2):230–246.

De Los Santos B, Hortacsu A, Wildenbeest MR (2012) Testing models of consumer search using data on web browsing and purchasing behavior. *American Economic Review* 102(6):2955–2980.

De Los Santos B, Koulayev S (2017) Optimizing click-through in online rankings with endogenous search refinement. *Marketing Science* 36(4):542–564.

Derakhshan M, Golrezaei N, Manshadi V, Mirrokni V (2020) Product ranking on online platforms, working paper.

Donnelly R, Kanodia A, Morozov I (2020) A unified framework for personalizing product rankings, working paper.

Dzyabura D, Hauser JR (2019) Recommending products when consumers learn their preference weights. *Marketing Science* 38(3):417–441.

Even-Dar E, Mannor S, Mansour Y (2006) Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of machine learning research* 7(Jun):1079–1105.

Gallego G, Li A, Truong VA, Wang X (2020) Approximation algorithms for product framing and pricing. *Operations Research* 68(1):134–160.

Ghose A, Ipeirotis PG, Li B (2012) Designing ranking systems for hotels on travel search engines by mining user-generated and crowdsourced content. *Marketing Science* 31(3):493–520.

Hochbaum DS (1997) *Approximation Algorithms for NP-hard problems* (PWS Publishing Co.).

Hoffman R, Yeh C (2018) *Blitzscaling: The Lightning-fast Path to Building Massively Valuable Businesses* (Broadway Business).

Jagabathula S, Rusmevichientong P (2017) A nonparametric joint assortment and price choice model. *Management Science* 63(9):3128–3145.

Jamieson KG, Nowak RD (2014) Best-arm identification algorithms for multi-armed bandits in the fixed confidence setting. *48th Annual Conference on Information Sciences and Systems, (CISS)*, 1–6.

Kalyanakrishnan S, Tewari A, Auer P, Stone P (2012) PAC subset selection in stochastic multi-armed bandits. *Proceedings of the 29th International Conference on Machine Learning, (ICML)*.

Kim JB, Albuquerque P, Bronnenberg BJ (2016) The probit choice model under sequential search with an application to online retailing. *Management Science* 63(11):3911–3929.

Koulayev S (2014) Search for differentiated products: identification and estimation. *The RAND Journal of Economics* 45(3):553–575.

Kveton B, Szepesvari C, Wen Z, Ashkan A (2015a) Cascading bandits: Learning to rank in the cascade model. *International Conference on Machine Learning*, 767–776.

Kveton B, Wen Z, Ashkan A, Szepesvári C (2015b) Combinatorial cascading bandits. *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, 1450–1458.

Lagrée P, Vernade C, Cappé O (2016) Multiple-play bandits in the position-based model. *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*, 1597–1605.

Li G, Rusmevichientong P, Topaloglu H (2015a) The d-level nested logit model: Assortment and price optimization problems. *Operations Research* 63(2):325–342.

Li JQ, Rusmevichientong P, Simester D, Tsitsiklis JN, Zoumpoulis SI (2015b) The value of field experiments. *Management Science* 61(7):1722–1740.

Moe W, Fader P (2004) Dynamic conversion behavior at e-commerce sites. *Management Science* 50(3):326–335.

Moe WW (2003) Buying, searching, or browsing: Differentiating between online shoppers using in-store navigational clickstream. *Journal of Consumer Psychology* 13(1-2):29–39.

Montgomery A, Li S, Srinivasan K, Liechty J (2004) Modeling online browsing and path analysis using clickstream data. *Marketing Science* 23(4):579–595.

Radlinski F, Kleinberg R, Joachims T (2008) Learning diverse rankings with multi-armed bandits. *Proceedings of the 25th international conference on Machine learning*, 784–791 (ACM).

Srikant R, Basu S, Wang N, Pregibon D (2010) User browsing models: relevance versus examination. *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 223–232 (ACM).

Ulu C, Honhon D, Alptekinoglu A (2012) Learning consumer tastes through dynamic assortments. *Operations Research* 60(4):833–849.

Ursu RM (2018) The power of rankings: Quantifying the effect of rankings on online consumer search and purchase decisions. *Marketing Science* 37(4):530–552.

Varian HR (2007) Position auctions. *International Journal of Industrial Organization* 25(6):1163–1178.

Wang R, Sahin O (2018) The impact of consumer search cost on assortment planning and pricing. *Management Science* 64(8):3649–3666.

Weitzman ML (1979) Optimal search for the best alternative. *Econometrica: Journal of the Econometric Society* 641–654.

Zoghi M, Tunys T, Ghavamzadeh M, Kveton B, Szepesvári C, Wen Z (2017) Online learning to rank in stochastic click models. *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 4199–4208.

Appendix A: Proofs

Proof of Proposition 1: Our proof involves showing a reduction from the deterministic maximum coverage problem to a special case of OffAR. Since the deterministic maximum coverage problem is known to be NP-Hard, the same claim follows for the special case of OffAR and thus the more general OffAR problem is also NP-Hard. The deterministic maximum coverage problem is defined by a universe $\mathcal{U}$ of elements $\{u_1, u_2, \ldots, u_m\}$, a collection of n sets $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$, where $S_i \subseteq \mathcal{U}$ for all i, and finally a single integer k such that $1 \leq k \leq n$. The objective is to select at most k sets from $\mathcal{S}$ in order to maximize the number of elements in $\mathcal{U}$ that are covered by (present in) at least one of the sets, i.e.

$$\max_{\mathcal{S}' \subseteq \mathcal{S}; |\mathcal{S}'| \leq k} \left| \cup_{S_i \in \mathcal{S}'} S_i \right|. \tag{3}$$

The reduction is straightforward. Given an instance of the deterministic maximum coverage problem, we construct an instance of OffAR with: (i) n products labeled by $[n]$, such that product i corresponds to set $S_i \in \mathcal{S}$; (ii) distribution $\mathcal{D}$ comprising of m customer types such that each type occurs with equal probability $\frac{1}{m}$, and for a customer t of the j^{th} type, $p_{it} = 1$ if and only if $u_j \in S_i$ in the original problem. In other words, we transform the problem so that for every set S_i, a new product is created and for every element of the universe $\mathcal{U}$, there is a customer type who prefers all of the sets (products) which contain that element of the universe (are liked by that customer type). Finally, all customers have a first impression window equal to k in this reduced instance, and bias function $f_{it}(\chi) = 0$ for all i, t and $\chi \subseteq [n]$

Consider any ranking π and let $S_\pi(k) \subseteq \mathcal{S}$ correspond to the top-k ranked products in π. Recall that $\pi^{-1}(j)$ denotes the j^{th} ranked product in π. Then, formally,

$$S_\pi(k) = \{S_{\pi^{-1}(1)}, \ldots, S_{\pi^{-1}(k)}\}.$$

Let $\mathbb{E}[H_t(\pi)]$ be the probability that a single incoming customer drawn from $\mathcal{D}$ is hooked when she is exposed to the ranking π. We then claim that for the deterministic maximum coverage problem with sets $S_\pi(k)$ chosen, the number of elements in $\mathcal{U}$ that are covered is precisely $\mathrm{m}\mathbb{E}[H_t(\pi)]$. Indeed, consider any customer type j belonging to the distribution $\mathcal{D}$ that is hooked by the ranking π: this implies that there exists some i having $\pi(i) \leq k$ such that $u_j \in S_i$. However, this in turn implies that the collection $S_\pi(k)$ must cover the element u_j in the universe. We further observe that each customer type in $\mathcal{D}$ (corresponding to an element in the universe) occurs with equal probability $\frac{1}{m}$. Therefore, there is a one-to-one correspondence

between the optimal solutions to the original, deterministic maximum coverage problem and the constructed (stochastic) offline assortment ranking problem, which in turn implies the NP-Hardness of the latter. □

Notation for Subsequent Proofs Before proving Theorem 1, we introduce notation that will help streamline our presentation. Specifically, we use $\overline{C}_t(r,\pi)$ to denote the indicator random variable for the event that a customer t does not click on any of the top-r ranked products in π, i.e., for $1 \leq r \leq n$

$$\overline{C}_t(r,\pi) = \prod_{j\in[r]} \overline{C}_{\pi^{-1}(j)t}(\pi).$$

We define $\overline{C}_t(0,\pi) = 1$ for notational convenience. As before, by indexing customer t, we implicitly assume that this random variable is conditional on customer t's type, $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$.

A subtle point bears mentioning here: $\overline{C}_{\pi^{-1}(j)t}(\pi) = 1$ implies that either $j > k_t$ or the customer did not click on product $\pi^{-1}(j)$ after seeing it. However, if the customer did not click on any of the top k_t ranked products, she is *not hooked* and therefore, cannot have clicked on any of the products with rank larger than k_t. Therefore, for any $r > k_t$, $\overline{C}_t(r,\pi) = \overline{C}_t(k_t,\pi)$ and it is accurate to interpret $\overline{C}_t(r,\pi) = 1$ as the event that the customer did not click on any of the top-r ranked products under π.

Finally, we note the following equality that will be used in several proofs.

$$\begin{aligned}\sum_{r=1}^{n} \mathbb{E}\big[\overline{C}_t(r-1,\pi) C_{\pi^{-1}(r)t}(\pi)\big] &= \sum_{r=1}^{n} \mathbb{E}\big[\big(1-\overline{C}_t(r,\pi)\big) - \big(1-\overline{C}_t(r-1,\pi)\big)\big] \\ &= \mathbb{E}\big[\big(1-\overline{C}_t(n,\pi)\big) - \big(1-\overline{C}_t(0,\pi)\big)\big] \\ &= \mathbb{E}[H_t(\pi)]. \end{aligned} \tag{4}$$

Proof of Theorem 1 Our objective is to prove that for any incoming customer $t \sim \mathcal{D}$, the probability that this customer is hooked under the optimal ranking π^* is at most twice the probability that she is hooked under the greedy ranking π^g. Mathematically, we need to prove that

$$\mathbb{E}[H_t(\pi^*)] = \mathbb{E}[1-\overline{C}_t(k_t,\pi^*)] \leq 2\ \mathbb{E}[1-\overline{C}_t(k_t,\pi^g)] = 2\mathbb{E}[H_t(\pi^g)],$$

where $\overline{C}_t(r,\pi)$ is defined immediately prior to the proof of this theorem.

We remark that the expectation involves two sources of uncertainty: (i) the fact that the arriving customer t's preferences $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$ are drawn from distribution $\mathcal{D}$, and (ii) given customer t with preferences $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$, the event that the customer clicks on product i within her first impression window is a random variable.

Since $H_t(\pi^*)$ is conditional on $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$, one could rewrite the above expectation as $\mathbb{E}_{(\boldsymbol{p}_t,\boldsymbol{f}_t,k_t)}[\mathbb{E}[H_t(\pi^*)]]$, where the inner expectation is the probability that a customer t is hooked given $\boldsymbol{p}_t$, $\boldsymbol{f}_t$ and k_t. For brevity, we write $\mathbb{E}_t$ for $\mathbb{E}_{(\boldsymbol{p}_t,\boldsymbol{f}_t,k_t)}$ in the rest of the proofs and capture both sources of uncertainty under a single expectation without a subscript when the meaning is clear from the context. Suppose that π denotes some arbitrary ranking. We begin by decomposing π^* in terms of π and eventually substitute $\pi = \pi^g$,

$$
\begin{aligned}
\mathbb{E}[H_t(\pi^*)] &= \mathbb{E}\big[1-\overline{C}_t(k_t,\pi^*)\big] \\
&= \mathbb{E}\big[1-\overline{C}_t(n,\pi^*)\big] && (5)\\
&\le \mathbb{E}\big[1-\overline{C}_t(n,\pi^*)\overline{C}_t(n,\pi)\big] && (6)\\
&= \mathbb{E}\big[1-\overline{C}_t(n,\pi)\big] + \sum_{r=1}^{n}\mathbb{E}\Big[\big(1-\overline{C}_t(n,\pi)\overline{C}_t(r,\pi^*)\big)-\big(1-\overline{C}_t(n,\pi)\overline{C}_t(r-1,\pi^*)\big)\Big] && (7)\\
&= \mathbb{E}[H_t(\pi)] + \sum_{r=1}^{n}\mathbb{E}\Big[\big(1-\overline{C}_t(n,\pi)\overline{C}_t(r,\pi^*)]\big)-\big(1-\overline{C}_t(n,\pi)\overline{C}_t(r-1,\pi^*)]\big)\Big] \\
&= \mathbb{E}[H_t(\pi)] + \sum_{r=1}^{n}\mathbb{E}\Big[\overline{C}_t(n,\pi)\left(\overline{C}_t(r-1,\pi^*)-\overline{C}_t(r,\pi^*)\right)\Big] \\
&= \mathbb{E}[H_t(\pi)] + \sum_{r=1}^{n}\mathbb{E}\Big[\overline{C}_t(n,\pi)\overline{C}_t(r-1,\pi^*)(1-\overline{C}_{\pi^{*-1}(r)t}(\pi^*))\Big] \\
&= \mathbb{E}[H_t(\pi)] + \sum_{r=1}^{n}\mathbb{E}\Big[\overline{C}_t(n,\pi)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\Big] \\
&\le \mathbb{E}[H_t(\pi)] + \sum_{r=1}^{n}\mathbb{E}\Big[\overline{C}_t(r-1,\pi)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\Big]. && (8)
\end{aligned}
$$

Equality (5) follows from the definition that $C_{it}(\pi) = 0$ when $\pi(i) > k_t$. Inequality (6) simply comes from the fact that $\overline{C}_t(n,\pi) \le 1$ over all realizations. The equality in (7) is a rearrangement of $\mathbb{E}\big[1-\overline{C}_t(n,\pi^*)\overline{C}_t(n,\pi)\big]$ in the form of a telescoping summation, i.e., $\sum_{r=1}^{n}\mathbb{E}\Big[\big(1-\overline{C}_t(n,\pi)\overline{C}_t(r,\pi^*)\big)-\big(1-\overline{C}_t(n,\pi)\overline{C}_t(r-1,\pi^*)\big)\Big] = \mathbb{E}\big[1-\overline{C}_t(n,\pi^*)\overline{C}_t(n,\pi)\big] - \mathbb{E}\big[1-\overline{C}_t(n,\pi)\big]$. The intermediate steps between (7) and (8) follow from simple algebraic manipulations and the definition of $C_{\pi^{*-1}(r)t}(\pi^*)$. Inequality (8) follows because:

$$\overline{C}_t(n,\pi) = \prod_{j\in[n]}\overline{C}_{\pi^{-1}(j)t}(\pi) \le \prod_{j\in[r-1]}\overline{C}_{\pi^{-1}(j)t}(\pi) = \overline{C}_t(r-1,\pi)$$

since the right-hand side refers to only a subset of the events on the left-hand side.

By substituting $\pi = \pi^g$ in (8), we get:

$$\mathbb{E}[H_t(\pi^*)] \le \mathbb{E}[H_t(\pi^g)] + \sum_{r=1}^{n}\mathbb{E}\Big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\Big] \tag{9}$$

To complete our proof, we next show that $\sum_{r=1}^{n}\mathbb{E}\Big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\Big] \le \mathbb{E}[H_t(\pi^g)]$. Consider the term $\mathbb{E}\big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\big]$ for any $r \in [n]$. We claim that

$$\mathbb{E}\big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\big] \le \mathbb{E}\big[\overline{C}_t(r-1,\pi^g)C_{(\pi^g)^{-1}(r)t}(\pi^g)\big]. \tag{10}$$

Note that $(\pi^g)^{-1}(r)$ denotes the product that is placed in the r^{th} position under ranking π^g. Equation (10) is formally presented and proved in Lemma 1 immediately following this proof, as it will be used in subsequent proofs as well. Informally speaking, its proof follows directly from the property that for each position r, the product $(\pi^g)^{-1}(r)$ selected by the greedy algorithm provides more marginal benefit (i.e., $\mathbb{E}[\overline{C}_t(r-1,\pi^g)C_{(\pi^g)^{-1}(r)t}(\pi^g)]$) than any other product that can be fixed in that position including $\pi^{*-1}(r)$.

From Equation (10) and applying Equation (4), we have

$$\sum_{r=1}^{n}\mathbb{E}\big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\big] \le \sum_{r=1}^{n}\mathbb{E}\big[\overline{C}_t(r-1,\pi^g)C_{(\pi^g)^{-1}(r)t}(\pi^g)\big] = \mathbb{E}[H_t(\pi^g)]. \tag{11}$$

Combining (9) and (11), we have

$$\mathbb{E}[H_t(\pi^*)] \le \mathbb{E}[H_t(\pi^g)] + \sum_{r=1}^{n}\mathbb{E}\big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\big] \le \mathbb{E}[H_t(\pi^g)] + \mathbb{E}[H_t(\pi^g)]. \quad \square$$

LEMMA 1. *Suppose that π^* denotes the optimal ranking and π^g denotes the ranking produced by the Greedy Algorithm for OffAR. Let $r \le n$ denote any index. Then, we have that:*

$$\mathbb{E}\big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\big] \le \mathbb{E}\big[\overline{C}_t(r-1,\pi^g)C_{(\pi^g)^{-1}(r)t}(\pi^g)\big].$$

Proof of Lemma 1: First, we define an alternative ranking $\tilde{\pi}^g$ such that for all $j \le r-1$, $(\tilde{\pi}^g)^{-1}(j) = (\pi^g)^{-1}(j)$ and $(\tilde{\pi}^g)^{-1}(r) = \pi^{*-1}(r)$. Recall that $(\tilde{\pi}^g)^{-1}(j)$ and $(\pi^g)^{-1}(j)$ refer to the product placed in the j^{th} rank under $\tilde{\pi}^g$ and π^g, respectively. That is, $\tilde{\pi}^g$ mirrors the ranking π^g in the first $r-1$ positions and shares the r^{th} rank with ranking π^*. Let $(\tilde{\pi}^g)^{-1}(j) = \emptyset$ when $j > r$, i.e. no product is ranked in positions greater than r.

Suppose that i denotes the r^{th} ranked product in π^*, i.e. $i \triangleq \pi^{*-1}(r)$. Going back to the expression in the left hand side of the lemma statement, we can now rewrite it as,

$$\begin{aligned}\mathbb{E}\big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\big] &= \mathbb{E}\big[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)\big] \\ &= \mathbb{E}_t\Big[\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)|\ (\boldsymbol{p}_t,\boldsymbol{f}_t,k_t)]\Big]\end{aligned}$$

In the final term, the outer expectation is with respect to the randomness in the customer's parameters (i.e., $(\boldsymbol{p}_t,\boldsymbol{f}_t,k_t)$) and the inner expectation is with respect to the randomness in the customer's clicks given $\boldsymbol{p}_t,\boldsymbol{f}_t$ and k_t. Note that since $\overline{C}_t(r-1,\pi^g)$, $\overline{C}_t(r-1,\pi^*)$, and $C_{it}(\pi^*)$ are already defined conditional on

$(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$, the inner expectation is somewhat redundant. Fix an arbitrary customer t and the corresponding type $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$. As an intermediate step to proving the lemma, we first show that for this fixed type:

$$\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)|(\boldsymbol{p}_t,\boldsymbol{f}_t,k_t)] \leq \mathbb{E}[\overline{C}_t(r-1,\tilde{\pi}^g)C_{it}(\tilde{\pi}^g)|(\boldsymbol{p}_t,\boldsymbol{f}_t,k_t)]. \tag{12}$$

For brevity, we ignore the conditional operator for the remainder of this argument. Consider the term in the left hand side of (12). This represents the event that customer t does not click on the the first $r-1$ ranked products in both π^g and π^* but does click on the r–th ranked product in π^*, which is i. We can therefore express this probabilistically as:

$$\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)] = Pr(\overline{C}_t(r-1,\pi^g)=1 \cap \overline{C}_t(r-1,\pi^*)=1 \cap C_{it}(\pi^*)=1) \tag{13}$$

$$\leq Pr\left(\overline{C}_t(r-1,\pi^g)=1 \cap C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right). \tag{14}$$

The inequality comes from the basic laws of probability. Namely, for any two events A, B, we have that $Pr(A \cap B) = Pr(A|B)Pr(B) \leq Pr(A|B)$. We now consider two separate cases and prove (12) separately in each of these cases.

- *Case I:* $\pi^g(i) < r$:

As indicated by the above condition, suppose that product i is ranked at a position smaller than r in π^g—let $\pi^g(i) = j \leq r-1$. Consider (14):

$$\begin{aligned}
\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)] &\leq Pr\left(\overline{C}_t(r-1,\pi^g)=1 \cap C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right) \\
&\leq Pr\left(\overline{C}_t(j,\pi^g)=1 \cap C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right). \\
&= Pr\left(\overline{C}_t(j-1,\pi^g)=1 \cap \overline{C}_{it}(\pi^g)=1 \cap C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right). \\
&\leq Pr\left(\overline{C}_{it}(\pi^g)=1 \cap C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1 \cap \overline{C}_t(j-1,\pi^g)=1\right).
\end{aligned}$$

Since product i is ranked at position j in π^g, we have: $\overline{C}_t(r-1,\pi^g) \leq \overline{C}_t(j,\pi^g)$—this gives rise to the second inequality. The final inequality follows from the same reasoning as (14).

Consider the final term above. For any ranking π, we know that conditional on no prior clicks, the event that a given customer t clicks on product i is a Bernoulli random variable with probability $p_{it} + f_{it}(\emptyset) = p_{it}$. Thus, conditional on $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$ and given no prior clicks in either ranking π^g or π^*, for any particular instantiation of the Bernoulli random variables corresponding to customer t, we have that

$$Pr\left(\overline{C}_{it}(\pi^g)=1 \cap C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1 \cap \overline{C}_t(j-1,\pi^g)=1\right) = 0.$$

In conclusion, the required inequality is true trivially:

$$\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)] = 0 \leq \mathbb{E}[\overline{C}_t(r-1,\tilde{\pi}^g)C_{it}(\tilde{\pi}^g)].$$

- *Case II:* $\pi^g(i) \geq r$:

In this case, product i is not present in the first $r-1$ ranks under π^g. Therefore, we note that conditional on $\overline{C}_t(r-1,\pi^*)=1$ the following two events:

$$\mathcal{E}_1 : \overline{C}_t(r-1,\pi^g)=1, \qquad \mathcal{E}_2 : C_{it}(\pi^*)=1$$

are independent of each other. Using this, we can simplify (13) as:

$$\begin{aligned}
&\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)] \\
&= Pr\left(\overline{C}_t(r-1,\pi^g)=1 \cap C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right) Pr(\overline{C}_t(r-1,\pi^*)=1) \\
&= Pr\left(\overline{C}_t(r-1,\pi^g)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right) Pr\left(C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right) Pr(\overline{C}_t(r-1,\pi^*)=1) \\
&= Pr\left(\overline{C}_t(r-1,\tilde{\pi}^g)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right) Pr(\overline{C}_t(r-1,\pi^*)=1) Pr\left(C_{it}(\tilde{\pi}^g)=1 \mid \overline{C}_t(r-1,\tilde{\pi}^g)=1\right) \quad (15) \\
&= Pr\left(\overline{C}_t(r-1,\tilde{\pi}^g)=1 \cap \overline{C}_t(r-1,\pi^*)=1\right) Pr\left(C_{it}(\tilde{\pi}^g)=1 \mid \overline{C}_t(r-1,\tilde{\pi}^g)=1\right) \\
&\leq Pr\left(\overline{C}_t(r-1,\tilde{\pi}^g)=1\right) Pr\left(C_{it}(\tilde{\pi}^g)=1 \mid \overline{C}_t(r-1,\tilde{\pi}^g)=1\right) \\
&= Pr\left(C_{it}(\tilde{\pi}^g)=1 \cap \overline{C}_t(r-1,\tilde{\pi}^g)=1\right) \\
&= \mathbb{E}[\overline{C}_t(r-1,\tilde{\pi}^g)C_{it}(\tilde{\pi}^g)].
\end{aligned}$$

The second expression comes from the fact that the events $\mathcal{E}_1$ and $\mathcal{E}_2$ are independent of each other. Equation (15) is a consequence of the following two observations:

1. The rankings π^g and $\tilde{\pi}^g$ contain the exact same products in the first $r-1$ positions. Therefore, $\overline{C}_t(r-1,\pi^g) = \overline{C}_t(r-1,\tilde{\pi}^g)$ for any particular instantiation of customer t's clicks.

2. Consider the two rankings π^* and $\tilde{\pi}^g$: in both of these, product i is ranked at position r. Further, for both of these rankings, the event that customer t clicks on product i conditional on not clicking on the previous $r-1$ ranks is simply a Bernoulli random variable with probability p_{it}. So, it must be the case that $Pr\left(C_{it}(\pi^*)=1 \mid \overline{C}_t(r-1,\pi^*)=1\right) = Pr\left(C_{it}(\tilde{\pi}^g)=1 \mid \overline{C}_t(r-1,\tilde{\pi}^g)=1\right)$. Note that if $k_t < r$, this equality is trivially true.

To summarize, whether or not $\pi^g(i) < r$, we have shown that $\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)] \leq \mathbb{E}[\overline{C}_t(r-1,\tilde{\pi}^g)C_{it}(\tilde{\pi}^g)]$. We are now ready to complete the proof of the lemma.

$$\begin{aligned}
\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)] &= \mathbb{E}_t\big[\mathbb{E}[\overline{C}_t(r-1,\pi^g)\overline{C}_t(r-1,\pi^*)C_{it}(\pi^*)|\ (\boldsymbol{p}_t,\boldsymbol{f}_t,k_t)]\big] \\
&\leq \mathbb{E}_t\big[\mathbb{E}[\overline{C}_t(r-1,\tilde{\pi}^g)C_{it}(\tilde{\pi}^g)|\ (\boldsymbol{p}_t,\boldsymbol{f}_t,k_t)]\big] \\
&= \mathbb{E}[\overline{C}_t(r-1,\tilde{\pi}^g)C_{it}(\tilde{\pi}^g)\big] \\
&\leq \mathbb{E}[\overline{C}_t(r-1,\pi^g)C_{(\pi^g)^{-1}(r)t}(\pi^g)\big]. \qquad (16)
\end{aligned}$$

The last inequality comes from the fact that for each rank r, the Greedy Algorithm for OffAR tries all of the remaining items in position r keeping the previous ranks fixed. The product having the maximum marginal utility (probability that the customer clicks on rank r but not the ones before) is selected. Mathematically, this implies that if $\pi^g(i) \geq r$, we have $\mathbb{E}[\overline{C}_t(r-1,\tilde{\pi}^g)C_{it}(\tilde{\pi}^g)\big] \leq \mathbb{E}[\overline{C}_t(r-1,\pi^g)C_{(\pi^g)^{-1}(r)t}(\pi^g)\big]$. If $\pi^g(i) < r$, we have $\overline{C}_t(r-1,\tilde{\pi}^g)C_{it}(\tilde{\pi}^g) \leq \overline{C}_{it}(\tilde{\pi}^g)C_{it}(\tilde{\pi}^g) = 0$, so (16) follows trivially. □

Proof of Theorem 2: We first introduce some notation that will be helpful in our proof. Since the intrinsic probabilities, bias functions, and the first impression windows are all independent of each other, let $\mathcal{D}_p, \mathcal{D}_f, \mathcal{D}_k$ denote the distributions of these three parameters, respectively, such that $\mathcal{D} = \mathcal{D}_p \times \mathcal{D}_f \times \mathcal{D}_k$. As before, let π^* denote the optimal ranking and π^g denote the ranking that results from the Greedy Algorithm for OffAR for any instance where $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t) \sim \mathcal{D}$. With slight abuse of notation and for this proof only, we let π_k^* and π_k^g denote the optimal and greedy rankings, respectively, for instances where $\boldsymbol{p}_t \sim \mathcal{D}_p, \boldsymbol{f}_t \sim \mathcal{D}_f, k_t = k$, i.e., where all customers have a first impression window exactly equal to k. Finally, for any ranking π, we use $[\pi]_r$ to denote the sub-ranking comprising only of the first r positions in π, i.e., $[\pi]_r(i) = \pi(i)$ if $\pi(i) \leq r$ and we define $[\pi]_r(i) = \emptyset$ otherwise.

The proof comprises of two components. First, we show that for every k, $\pi_k^g = [\pi^g]_k$, i.e. the ranking returned by the Greedy Algorithm for OffAR when $\boldsymbol{p}_t \sim \mathcal{D}_p, \boldsymbol{f}_t \sim \mathcal{D}_f, k_t = k$ is equivalent to the first k positions[15] in ranking π^g, and hence π^g is a $(1-\frac{1}{e})$-approximation to π_k^*. Second, we prove that $\sum_{k=1}^n E_{\mathcal{D}_p,\mathcal{D}_f}[H_t(\pi_k^*)|k_t = k]Pr(k_t = k) \geq E_{\mathcal{D}}[H_t(\pi^*)]$. That is, the performance of the optimal ranking π^* is not better than a linear

[15] Technically, π_k^g contains products in positions beyond k as well but since all customers have first impression windows equal to k we can safely assume that $(\pi_k^g)^{-1}(j) = \emptyset$ when $j > k$.

combination of the performance of rankings π_k^* weighted according to the probability that a customer has a first impression window equal to k.

We begin by proving that π_k^g is a $(1-\frac{1}{e})$-approximation to π_k^*. Fix any k and suppose that $k_t = k$ for all t. For a constant first impression window ($k_t = k$), OffAR is simply a special case of stochastic submodular optimization and therefore, we can leverage known results (Asadpour and Nazerzadeh 2015) to infer that π_k^g is a $(1-\frac{1}{e})$-approximation to π_k^*.

We next prove that π_k^g coincides with $[\pi^g]_k$ by induction on the ranks, specifically, $(\pi_k^g)^{-1}(r) = (\pi^g)^{-1}(r)$ for every $1 \leq r \leq k$. We remind the reader that the inverse functions here capture the product placed in the r^{th} position in the corresponding rankings. We define $\pi^{-1}(0) = \emptyset$ for every ranking π and thus the claim is trivially true when $r = 0$. Our inductive hypothesis is that $(\pi_k^g)^{-1}(j) = (\pi^g)^{-1}(j)$ for $j = 1, ..., r-1$. Recall that $[\pi]_{r-1}$ refers to the sub-ranking of π with only the first $r-1$ positions filled, and let $[\pi]_{r-1} \cup i$ refer to the ranking where the first $r-1$ positions coincide with π and product i is placed in the r^{th} rank. Now, the r^{th} ranked product in the greedy ranking when $k_t = k$ can be mathematically characterized as:

$$(\pi_k^g)^{-1}(r) = \arg\max_{i\in[n]} \mathbb{E}_{\substack{\boldsymbol{p}_t\sim\mathcal{D}_p \\ \boldsymbol{f}_t\sim\mathcal{D}_f \\ k_t=k}} \left[H_t([\pi_k^g]_{r-1} \cup i) - H_t([\pi_k^g]_{r-1})\right] = \arg\max_{i\in[n]} \mathbb{E}_{\substack{\boldsymbol{p}_t\sim\mathcal{D}_p \\ \boldsymbol{f}_t\sim\mathcal{D}_f \\ k_t=k}} \left[H_t([\pi^g]_{r-1} \cup i) - H_t([\pi^g]_{r-1})\right], \tag{17}$$

where the second equality follows from the inductive hypothesis. Since π^g is the greedy ranking when customer profiles are drawn from $\mathcal{D}$, the product placed in rank r is chosen according to the following equation.

$$\begin{aligned}
(\pi^g)^{-1}(r) &= \arg\max_{i\in[n]} \mathbb{E}_{\substack{\boldsymbol{p}_t\sim\mathcal{D}_p \\ \boldsymbol{f}_t\sim\mathcal{D}_f \\ k_t\sim\mathcal{D}_k}} \left[H_t([\pi^g]_{r-1} \cup i) - H_t([\pi^g]_{r-1})\right] \\
&= \arg\max_{i\in[n]} \mathbb{E}_{\substack{\boldsymbol{p}_t\sim\mathcal{D}_p \\ \boldsymbol{f}_t\sim\mathcal{D}_f}} \left[H_t([\pi^g]_{r-1} \cup i) - H_t([\pi^g]_{r-1}) | k_t \geq r\right] Pr(k_t \geq r) \\
&= \arg\max_{i\in[n]} \mathbb{E}_{\substack{\boldsymbol{p}_t\sim\mathcal{D}_p \\ \boldsymbol{f}_t\sim\mathcal{D}_f}} \left[H_t([\pi^g]_{r-1} \cup i) - H_t([\pi^g]_{r-1}) | k_t \geq r\right] \\
&= \arg\max_{i\in[n]} \mathbb{E}_{\substack{\boldsymbol{p}_t\sim\mathcal{D}_p \\ \boldsymbol{f}_t\sim\mathcal{D}_f}} \left[H_t([\pi^g]_{r-1} \cup i) - H_t([\pi^g]_{r-1}) | k_t = k\right] \qquad (18) \\
&= \arg\max_{i\in[n]} \mathbb{E}_{\substack{\boldsymbol{p}_t\sim\mathcal{D}_p \\ \boldsymbol{f}_t\sim\mathcal{D}_f \\ k_t=k}} \left[H_t([\pi^g]_{r-1} \cup i) - H_t([\pi^g]_{r-1})\right] = (\pi^g)_k^{-1}(r). \qquad (19)
\end{aligned}$$

The second equality is crucially dependent on the independence between the first impression windows and the intrinsic click probabilities and the bias functions. It follows from the fact that when $k_t < r$, $H_t([\pi^g]_{r-1} \cup i) - H_t([\pi^g]_{r-1}) = 0$ for all $i \in [n]$. Moreover, $Pr(k_t \geq r)$ is a constant that is independent of $\mathcal{D}_p$ and $\mathcal{D}_f$. Note that we removed the expectation over $\mathcal{D}_k$ because once we condition on $k_t \geq r$, the exact value of k_t does

not matter since $H_t([\pi^g]_{r-1} \cup i)$ does not contain any products at ranks beyond r. The third equality follows since the index at which a variable is maximized does not change when multiplied by a constant - in this case $Pr(k_t \geq r)$. Equality (18) is due to the fact that customer preferences do not depend on the exact value of the first impression window; moreover, both $[\pi^g]_{r-1} \cup i$ and $[\pi^g]_{r-1}$ do not have any products in ranks greater than r, and therefore do not have any products in ranks greater than k, which by definition is larger than or equal to r. Equation (19) follows from (17) and completes the inductive argument. Thus, we have for all $k \in [1, n]$,

$$\mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim \mathcal{D}_f \\ k_t = k}}[H_t(\pi^g)] = \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim \mathcal{D}_f \\ k_t = k}}[H_t(\pi^g_k)] \geq (1 - \frac{1}{e}) \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim \mathcal{D}_f \\ k_t = k}}[H_t(\pi^*_k)]. \tag{20}$$

We move on to the second part of the proof, where we relate the hook probability of π^*_k to that of π^*. Note that for the instance where $\boldsymbol{p}_t \sim D_p$, $\boldsymbol{f}_t \sim D_f$, and $k_t = k$, π^*_k is the optimal ranking and therefore, its performance must surpass that of π^*. This gives us,

$$\begin{aligned}
&\mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t = k}}[H_t(\pi^*_k)] \geq \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t = k}}[H_t(\pi^*)] \quad \forall k \in [1, n] \\
\implies &\sum_{k=1}^{n} \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t = k}}[H_t(\pi^*_k)] \, Pr(k_t = k) \geq \sum_{k=1}^{n} \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t = k}}[H_t(\pi^*)] \, Pr(k_t = k) \\
&= \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f}} \left[\sum_{k=1}^{n} E[H_t(\pi^*) | k_t = k] Pr(k_t = k) \right] \\
&= \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t \sim \mathcal{D}_k}}[H_t(\pi^*)].
\end{aligned}$$

Combining the above inequality with (20), the theorem follows. That is,

$$\begin{aligned}
\mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t \sim \mathcal{D}_k}}[H_t(\pi^g)] &= \sum_{k=1}^{n} \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t = k}}[H_t(\pi^g)] Pr(k_t = k) \\
&\geq (1 - \frac{1}{e}) \sum_{k=1}^{n} \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t = k}}[H_t(\pi^*_k)] Pr(k_t = k) \\
&\geq (1 - \frac{1}{e}) \mathbb{E}_{\substack{\boldsymbol{p}_t \sim \mathcal{D}_p \\ \boldsymbol{f}_t \sim D_f \\ k_t \sim \mathcal{D}_k}}[H_t(\pi^*)]. \quad \square
\end{aligned}$$

Proof of Theorem 3 We only prove the $\approx \frac{1}{2}$-approximation guarantee for OnAR in the case when the click probabilities, bias functions, and first impression windows are correlated;[16] the proof for the $\approx (1 - \frac{1}{e})$-approximation guarantee for OnAR with Independence follows in an almost identical fashion using the

[16] i.e., $\mathcal{D}$ cannot be decomposed into three separate distributions as in the proof of Theorem 2.

independence arguments outlined in the proof of Theorem 2, so we avoid reproving it to minimize redundancy. Recall that $\tilde{\pi}$ denotes the output of Algorithm 2 while π^* is the optimal ranking. As in the proof of Theorem 2, suppose that for any ranking π, $[\pi]_r$ is the sub-ranking of π such that $[\pi]_r(i) = \pi(i)$ if $\pi(i) \leq r$ and we define $[\pi]_r(i) = \emptyset$ otherwise. Finally, let $[\pi]_r \cup i$ be an augmentation of $[\pi]_r$ such that its first r positions coincide with π and product i is placed in rank $r+1$.

Now, as per Definition 1 and Algorithm 2, we can infer that for any given rank r, the product $\tilde{\pi}^{-1}(r)$ is a $(\frac{2\epsilon}{n}, \frac{\delta}{n})$-PAC approximation to the best product at rank r subject to the products already fixed at previous ranks. Formally, with probability $1 - \frac{\delta}{n}$, the following inequality holds for each $r \in [1, n]$:

$$\mathbb{E}[\overline{C}_t(r-1, \tilde{\pi}) C_{\tilde{\pi}^{-1}(r)t}(\tilde{\pi})] \geq \max_{i \in [n]} \mathbb{E}[\overline{C}_t(r-1, \tilde{\pi}) C_{it}([\tilde{\pi}]_{r-1} \cup i)] - \frac{2\epsilon}{n}, \tag{21}$$

We note that even though $\tilde{\pi}$ is itself a random ranking due to sampling differences, (21) holds for any particular instantiation of $[\tilde{\pi}]_{r-1}$ with probability $1 - \frac{\delta}{n}$. Recall that $\overline{C}_t(r-1, \tilde{\pi}) = 1$ when customer t does not click on any of the first $r-1$ ranked products in $\tilde{\pi}$ conditional on $(\boldsymbol{p}_t, \boldsymbol{f}_t, k_t)$. Therefore, the left hand side of the above inequality is the probability that an incoming customer t is 'hooked' by the r^{th} ranked product in $\tilde{\pi}$ for some fixed instantiation of $[\tilde{\pi}]_{r-1}$. Suppose that for each r, B_r denotes the event that (21) holds for index r, i.e., $B_r = 1$ if (21) is true and is zero if the inequality is not true. We have $Pr(B_r = 0) \leq \frac{\delta}{n}$ for all $r \in [1, n]$.

From the union bound, we have that:

$$Pr(B_r = 1 \ \ \forall r) = 1 - Pr(\exists r : B_r = 0) \geq 1 - \sum_{r=1}^{n} Pr(B_r = 0) \geq 1 - n\frac{\delta}{n} = 1 - \delta. \tag{22}$$

In simple terms, by applying the union bound, we can infer that with probability $1 - \delta$, the ranking output by our algorithm satisfies the following condition: *for all $r \in [n]$, the product at rank r in $\tilde{\pi}$ is approximately optimal for that rank with respect to the products fixed at previous ranks.* For the rest of this proof, we only concern ourselves with rankings $\tilde{\pi}$ output by our algorithm where $B_r = 1$ for all r, which happens to be the case with probability $1 - \delta$.

Specifically, let us fix the ranking $\tilde{\pi}$ returned by Algorithm 2 such that this ranking satisfies the condition in (22). For any such fixed ranking $\tilde{\pi}$, we aim to show that $\mathbb{E}[H_t(\tilde{\pi})] \geq \frac{1}{2}\mathbb{E}[H_t(\pi^*)] - \epsilon$, or equivalently, $\mathbb{E}[H_t(\pi^*)] \leq 2\mathbb{E}[H_t(\tilde{\pi})] + 2\epsilon$. By applying (8) from the proof of Theorem 1 with $\pi = \tilde{\pi}$, we get that:

$$\mathbb{E}[H_t(\pi^*)] \leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}\Big[\overline{C}_t(r-1, \tilde{\pi}) \overline{C}_t(r-1, \pi^*) C_{\pi^{*-1}(r)t}(\pi^*)\Big]. \tag{23}$$

Fix some rank r and suppose that $\pi^{*-1}(r) = i^*$. Now proceeding exactly as in Lemma 1, for any $r \geq 1$, we have:

$$\mathbb{E}\Big[\overline{C}_t(r-1,\tilde{\pi})\overline{C}_t(r-1,\pi^*)C_{i^*t}(\pi^*)\Big] \leq \mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{i^*t}([\tilde{\pi}]_{r-1} \cup i^*)\big]. \tag{24}$$

To see why this is the case, first suppose that[17] $\tilde{\pi}^{-1}(i) \geq r$. Then, using arguments similar to those in Lemma 1, we can bound

$$\begin{aligned}
&\mathbb{E}\Big[\overline{C}_t(r-1,\tilde{\pi})\overline{C}_t(r-1,\pi^*)C_{i^*t}(\pi^*)\Big] \\
&= Pr\left(\overline{C}_t(r-1,\tilde{\pi})\overline{C}_t(r-1,\pi^*)C_{i^*t}(\pi^*) = 1\right) \\
&= Pr\left(C_{i^*t}(\pi^*) = 1 \mid \overline{C}_t(r-1,\pi^*)\overline{C}_t(r-1,\tilde{\pi}) = 1\right) Pr(\overline{C}_t(r-1,\pi^*)\overline{C}_t(r-1,\tilde{\pi}) = 1) \\
&\leq Pr\left(C_{i^*t}([\tilde{\pi}]_{r-1} \cup i^*) = 1 \mid \overline{C}_t(r-1,[\tilde{\pi}]_{r-1} \cup i^*) = 1\right) Pr(\overline{C}_t(r-1,\tilde{\pi}) = 1) \\
&= Pr\left(\overline{C}_t(r-1,[\tilde{\pi}]_{r-1} \cup i^*)C_{i^*t}([\tilde{\pi}]_{r-1} \cup i^*) = 1\right) \\
&= \mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{i^*t}([\tilde{\pi}]_{r-1} \cup i^*)\big],
\end{aligned}$$

where the inequality relies on the fact that $Pr\left(C_{i^*t}(\pi^*) = 1 \mid \overline{C}_t(r-1,\pi^*)\overline{C}_t(r-1,\tilde{\pi}) = 1\right) = E_t[p_{i^*t}] = Pr\left(C_{i^*t}([\tilde{\pi}]_{r-1} \cup i^*) = 1 \mid \overline{C}_t(r-1,[\tilde{\pi}]_{r-1} \cup i^*) = 1\right)$.

Now that we have shown $\mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{i^*t}([\tilde{\pi}]_{r-1} \cup i^*)\big]$ is an upper bound for $\mathbb{E}\Big[\overline{C}_t(r-1,\tilde{\pi})\overline{C}_t(r-1,\pi^*)C_{i^*t}(\pi^*)\Big]$, we can leverage (21) and continue with the proof:

$$\begin{aligned}
\mathbb{E}\Big[\overline{C}_t(r-1,\tilde{\pi})\overline{C}_t(r-1,\pi^*)C_{i^*t}(\pi^*)\Big] &\leq \mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{i^*t}([\tilde{\pi}]_{r-1} \cup i^*)\big] \\
&\leq \max_{i \in [n]} \mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r-1} \cup i)\big] \\
&\leq \mathbb{E}[\overline{C}_t(r-1,\tilde{\pi})C_{\tilde{\pi}^{-1}(r)t}(\tilde{\pi})] + \frac{2\epsilon}{n}.
\end{aligned} \tag{25}$$

Substituting (25) into (23), we have

$$\begin{aligned}
\mathbb{E}[H_t(\pi^*)] &\leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}\Big[\overline{C}_t(r-1,\tilde{\pi})\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\Big] \\
&\leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \left(\mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{\tilde{\pi}^{-1}(r)t}(\tilde{\pi})\big] + \frac{2\epsilon}{n}\right) \\
&= \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{\tilde{\pi}^{-1}(r)t}(\tilde{\pi})\big] + 2\epsilon \\
&= \mathbb{E}[H_t(\tilde{\pi})] + \mathbb{E}[H_t(\tilde{\pi})] + 2\epsilon
\end{aligned} \tag{26}$$

[17] When $\tilde{\pi}^{-1}(i) < r$, (24) holds trivially at equality since both the left and right hand sides evaluate to zero.

$$= 2\mathbb{E}[H_t(\tilde{\pi})] + 2\epsilon$$

where (26) follows by applying (4). □

Proof of Theorem 4 Without loss of generality, we assume that $\log_{1+\alpha}(\frac{n}{\epsilon})$ is integral, and so during the course of the algorithm, $\tau \in \{1, \frac{1}{1+\alpha}, \frac{1}{(1+\alpha)^2}, \ldots, \frac{\epsilon}{n}\}$.[18] For the purpose of this proof, we will also implicitly assume that the parameters α, ϵ, δ are chosen such that $n \geq \log_{1+\alpha}(\frac{n}{\epsilon})$. Indeed, when this is not the case, the length of the learning phase for the Threshold Acceptance Algorithm becomes $\Omega(n^4)$ and it is advantageous to run Algorithm 2 instead.

Recall that for a fixed threshold τ, our algorithm takes one pass through the set of available products and fixes any product at the smallest open rank as long as its marginal benefit at that rank is at least τ. Therefore, for a given threshold, the algorithm could assign multiple products to successive ranks as long as the marginal benefit of all of these products meet the threshold at the corresponding ranks. Subsequently, the threshold is lowered to $\frac{1}{1+\alpha}\tau$ for the given parameter $\alpha > 0$, and the above process is repeated on the remaining, unranked products.

We first introduce pertinent notation. For any rank $r \in [1, n]$, let τ_r be the threshold value that the product $\tilde{\pi}^{-1}(r)$ had met in order to be assigned to the corresponding rank. Conversely, let $r^{(\tau)}$ be the smallest rank at which our algorithm considers the threshold τ or, equivalently, the rank when the algorithm transitions from a threshold of $(1+\alpha)\tau$ to τ. Due to the fact that the thresholds are monotonically decreasing, this implies that for all $r' < r^{(\tau)}$, $\tau_{r'} > \tau$. As in earlier proofs, suppose that for any ranking π, $[\pi]_r$ is the sub-ranking of π such that $[\pi]_r(i) = \pi(i)$ if $\pi(i) \leq r$ and we define $[\pi]_r(i) = \emptyset$ otherwise. Finally, let $[\pi]_r \cup i$ be an augmentation of $[\pi]_r$ such that its first r positions coincide with π and product i is placed in rank $r+1$.

In the first part of our proof, we will use Hoeffding's inequality to bound the expected number of new customers hooked at each rank. We move the details of Hoeffding's inequality to Lemmas 2 and 3 immediately following this proof for ease of exposition. Consider the product $\tilde{\pi}^{-1}(r)$ that our algorithm fixes at rank r. We know that ranking $[\tilde{\pi}]_r$ was displayed to $L = \frac{n^2}{\epsilon^2}\log(\frac{n}{\sqrt{\delta}})$ customers, and at least a fraction τ_r of these customers clicked exclusively on product $\tilde{\pi}^{-1}(r)$. Consider the Bernoulli random variable $X = \overline{C}_t(r -$

[18] Since $\log_{1+\alpha}(\frac{n}{\epsilon})$ is integral, this implies that there exists an integer a such that $\tau^{\max}\left(\frac{1}{1+\alpha}\right)^a = \tau^{\min} = \frac{\epsilon}{n}$. Therefore, Algorithm 3 tries out exactly $a+1$ different values of τ belonging to the set $\{1, \frac{1}{1+\alpha}, \frac{1}{(1+\alpha)^2}, \ldots, \frac{\epsilon}{n}\}$.

$1, [\tilde{\pi}]_r) C_{\tilde{\pi}^{-1}(r)t}([\tilde{\pi}]_r)$ which equals one if the customer clicks on the product at rank r without clicking on any of the previously ranked products and is zero otherwise. We have $\frac{1}{L}\sum_{i=1}^{L} X_i \geq \tau_r$, where X_i denotes the i–th sample of the Bernoulli random variable X corresponding to customer $i \leq L$. Applying Lemma 2, we have that with probability at least $1 - \frac{\delta}{n^2}$, the following inequality is true:

$$\mathbb{E}\big[\overline{C}_t(r-1, [\tilde{\pi}]_r) C_{\tilde{\pi}^{-1}(r)t}([\tilde{\pi}]_r)\big] \geq \frac{1}{L}\sum_{i=1}^{L} X_i - \frac{\epsilon}{n} \geq \tau_r - \frac{\epsilon}{n}. \tag{27}$$

We note that since $\overline{C}_t(r-1, [\tilde{\pi}]_r) C_{\tilde{\pi}^{-1}(r)t}([\tilde{\pi}]_r)$ evaluates to zero whenever the customer clicks on one of the top $r-1$ ranked products, its expectation does not depend on the exact form of the bias function $f_{it}(\chi)$. Further, even though $[\tilde{\pi}]_r$ is itself a random ranking, the above inequality holds with probability at least $1 - \frac{\delta}{n^2}$ for any particular instantiation of $[\tilde{\pi}]_{r-1}$. Finally, since $[\tilde{\pi}]_r$ coincides with $\tilde{\pi}$ in its first r positions, Equation (27) is equivalent to writing:

$$\mathbb{E}\big[\overline{C}_t(r-1, \tilde{\pi}) C_{\tilde{\pi}^{-1}(r)t}(\tilde{\pi})\big] \geq \frac{1}{L}\sum_{i=1}^{L} X_i - \frac{\epsilon}{n} \geq \tau_r - \frac{\epsilon}{n}. \tag{28}$$

Let $U^{(\tau)}$ be the random set of unranked products when the algorithm lowers the threshold from $(1+\alpha)\tau$ to τ, for any arbitrary τ in the domain $\{\frac{1}{1+\alpha}, \frac{1}{(1+\alpha)^2}, \ldots, \frac{\epsilon}{n}, \frac{\epsilon}{n(1+\alpha)}\}$. For $\tau = 1$, we define $U^{(\tau)} = [n]$, since none of the products are assigned at the beginning of our algorithm. For any $i \in U^{(\tau)}$, there must exist some rank $r' \leq r^{(\tau)}$ at which L customers were shown the ranking $[\tilde{\pi}]_{r'-1} \cup i$ and the fraction that clicked on product i without clicking on any of the previously ranked products was smaller than $(1+\alpha)\tau$. Applying Lemma 3 for some given τ in the above range with r' being the rank where product i did not meet the threshold, we have that with probability at least $1 - \frac{\delta}{n^2}$

$$\mathbb{E}\big[\overline{C}_t(r'-1, [\tilde{\pi}]_{r'-1} \cup i) C_{it}([\tilde{\pi}]_{r'-1} \cup i)\big] \leq \frac{1}{L}\sum_{j=1}^{L} X_j + \frac{\epsilon}{n} < (1+\alpha)\tau + \frac{\epsilon}{n}, \tag{29}$$

where $(X_j)_{j=1}^{L}$ is the instantiation of the random variable $\overline{C}_t(r'-1, [\tilde{\pi}]_{r'-1} \cup i) C_{it}([\tilde{\pi}]_{r'-1} \cup i))$ when the ranking $[\tilde{\pi}]_{r'-1} \cup i$ was displayed to each of the L customers.

Since $r' \leq r^{(\tau)}$ and $[\tilde{\pi}]_{r-1} \cup i$ coincides with $\tilde{\pi}$ in its first $r-1$ positions, by monotonicity, we also have that

$$\mathbb{E}\big[\overline{C}_t(r^{(\tau)}-1, \tilde{\pi}) C_{it}([\tilde{\pi}]_{r^{(\tau)}-1} \cup i)\big] \leq \mathbb{E}\big[\overline{C}_t(r'-1, [\tilde{\pi}]_{r'-1} \cup i) C_{it}([\tilde{\pi}]_{r'-1} \cup i)\big].$$

We remark that the above inequality is independent of the exact functional form of the bias function. To see why, first note that for any fixed customer type, not clicking on the first $r^{(\tau)}-1$ products in $\tilde{\pi}$ implies that the customer does not click on the first $r'-1$ products in $\tilde{\pi}$ as well; similarly with not clicking on product i at ranks

$r^{(\tau)}$ versus r', assuming no prior clicks. Therefore, the set of events where $\overline{C}_t(r'-1,[\tilde{\pi}]_{r'-1}\cup i)C_{it}([\tilde{\pi}]_{r'-1}\cup i)$ evaluates to zero is a subset of the set of events where $\overline{C}_t(r^{(\tau)}-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r^{(\tau)}-1}\cup i)$ evaluates to zero. Hence, the alteration of the click probabilities once the customer gets hooked does not affect the value of either of these quantities. Combining the above inequality with (29), for all τ and all $i\in U^{(\tau)}$, the following holds[19] with probability at least $1-\frac{\delta}{n^2}$:

$$\mathbb{E}\big[\overline{C}_t(r^{(\tau)}-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r^{(\tau)}-1}\cup i)\big]\leq(1+\alpha)\tau+\frac{\epsilon}{n}. \tag{30}$$

Once again, we highlight that the above inequality holds for any particular instantiation of $[\tilde{\pi}]_{r^{(\tau)}-1}$. Let us use $E_i^{(\tau)}$ to denote the event (indicator variable) that (30) is true for a given τ and $i\in U^{(\tau)}$ for the ranking output by our algorithm. We know that $Pr(E_i^{(\tau)}=0)\leq\frac{\delta}{n^2}$. Analogously let F_r be the event that (28) is true for a given rank r at which $\tilde{\pi}^{-1}(r)\neq\emptyset$. We also have $Pr(F_r=0)\leq\frac{\delta}{n^2}$. Applying the union bound as we did in the proof of Theorem 3, we get that for a given τ:

$$\begin{aligned}Pr\Big(\exists i\in U^{(\tau)}\text{ s.t. }E_i^{(\tau)}=0\bigcup\exists r:\tau_r=(1+\alpha)\tau\text{ s.t. }F_r=0\Big)&\leq\sum_{i\in U^{(\tau)}}Pr(E_i^{(\tau)}=0)+\sum_{\substack{r\\ \tau_r=(1+\alpha)\tau}}Pr(F_r=0)\\&\leq\sum_{i\in U^{(\tau)}}\frac{\delta}{n^2}+\sum_{\substack{r\\ \tau_r=(1+\alpha)\tau}}\frac{\delta}{n^2}\\&\leq n\Big(\frac{\delta}{n^2}\Big)=\frac{\delta}{n}.\end{aligned}$$

The final inequality comes from the fact that $|U^{(\tau)}|+|r:\tau_r=(1+\alpha)\tau|\leq n$ because the products that are fixed in the ranks specified by the set $\{r:\tau_r=(1+\alpha)\tau\}$ must by definition belong to the set $[n]\setminus U^{(\tau)}$. We can apply the above inequality over all feasible threshold values τ to get

$$Pr\Big(\exists\tau:\{\exists i\in U^{(\tau)}\text{ s.t. }E_i^{(\tau)}=0\bigcup\exists r:\tau_r=(1+\alpha)\tau\text{ s.t. }F_r=0\}\Big)\leq\log_{1+\alpha}(\frac{n}{\epsilon})\frac{\delta}{n}\leq\delta. \tag{31}$$

Here, we used the fact that the number of feasible values of τ is exactly $\log_{1+\alpha}(\frac{n}{\epsilon})$, which by our assumption is no larger than n. Informally, (31) gives an upper bound on the (error) probability that the empirical marginal click probabilities estimated by our algorithm are not close to the true click probabilities for all thresholds and all ranks and products corresponding to those thresholds.

Now that we have bounded the expected number of new customers hooked at each rank, for the rest of this proof we will consider the regime where these empirical estimates are close to their expected values, i.e.

[19] We note that inequality (30) is trivially true for $\tau=1$ because the right hand side is greater than one, and the click probability cannot exceed this value.

we only consider rankings $\tilde{\pi}$ output by our algorithm such that $E_i^{(\tau)} = 1 \ \forall i \in U^{(\tau)} \bigcap F_r = 1 \ \forall r : \tau_r = (1+\alpha)\tau$ for all τ. We know that with probability $1-\delta$, our algorithm returns such a ranking. We aim to show that in this regime, any ranking $\tilde{\pi}$ computed by our algorithm satisfies: $\mathbb{E}[H_t(\tilde{\pi})] \geq \frac{1}{2+\alpha}\mathbb{E}[H_t(\pi^*)] - \epsilon$.

Consider a fixed ranking $\tilde{\pi}$ returned by our algorithm that satisfies the conditions mentioned above. We apply (8) from the proof of Theorem 1 with $\pi = \tilde{\pi}$ to get that

$$\begin{aligned}
\mathbb{E}[H_t(\pi^*)] &\leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}\Big[\overline{C}_t(r-1,\tilde{\pi})\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\Big] \\
&\leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}([\tilde{\pi}]_{r-1} \cup \pi^{*-1}(r))\big] \qquad (32) \\
&\leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \max_{i\in[n]} \mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r-1} \cup i)\big] \\
&= \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \max_{i\in U^{(\tau_r)}} \mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r-1} \cup i)\big] \qquad (33) \\
&\leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \Big((1+\alpha)\tau_r + \frac{\epsilon}{n}\Big). \qquad (34)
\end{aligned}$$

We note Inequality (32) follows from the same reasoning used in the first half of the proof of Lemma 1, so we do not repeat it here. For (33), we replaced the set $[n]$ by $U^{(\tau_r)}$, the latter being a (superset) of the set of unassigned products when the algorithm considers the rank r. Note that for all $i \notin U^{\tau_r}$, $\tilde{\pi}(i) < r$ and so, $\mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r-1} \cup i)\big] = 0$. Finally, (34) comes from (30).

Now, suppose that Algorithm 3 returns a ranking $\tilde{\pi}$, with products assigned to ranks one through $n' < n$. In this case, we can once again apply (30) and infer that the marginal benefit for all of the unranked products at rank $n'+1$ must be strictly smaller than $\tau^{\min} + \frac{\epsilon}{n}$ - if this were not the case, these products would have already been fixed at rank $n'+1$ or earlier. Thus our algorithm ends by updating $\tau_{n'+1} = \frac{\tau^{min}}{1+\alpha}$; equivalently, when $r > n'$, $(1+\alpha)\tau_r = \frac{\epsilon}{n}$ since $\tau^{\min} = \frac{\epsilon}{n}$ in the theorem statement. We slightly abuse notation and use $U^{(\tau_r)} = U^{(\frac{\epsilon}{n(1+\alpha)})}$ for any $r > n'$, with the implication being that this set contains products that were not fixed at any rank at the conclusion of our algorithm. We now continue with (34):

$$\begin{aligned}
\mathbb{E}[H_t(\pi^*)] &\leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n} \Big((1+\alpha)\tau_r + \frac{\epsilon}{n}\Big) \\
&= \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n'} (1+\alpha)(\tau_r) + \sum_{r=n'+1}^{n} (1+\alpha)(\tau_r) + \epsilon \\
&= \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n'} (1+\alpha)(\tau_r) + (n-n')\frac{\epsilon}{n} + \epsilon \\
&\leq \mathbb{E}[H_t(\tilde{\pi})] + \sum_{r=1}^{n'} (1+\alpha)(\mathbb{E}\big[\overline{C}_t(r-1,\tilde{\pi})C_{\tilde{\pi}^{-1}(r)t}(\tilde{\pi})\big] + \frac{\epsilon}{n}) + (n-n')\frac{\epsilon}{n} + \epsilon \qquad (35)
\end{aligned}$$

$$\leq \mathbb{E}[H_t(\tilde{\pi})] + (1+\alpha)\sum_{r=1}^{n'} \mathbb{E}\left[\overline{C}_t(r-1,\tilde{\pi})C_{\tilde{\pi}^{-1}(r)t}(\tilde{\pi})\right] + (1+\alpha)\epsilon + \epsilon$$

$$= \mathbb{E}[H_t(\tilde{\pi})] + (1+\alpha)\mathbb{E}[H_t(\tilde{\pi})] + (2+\alpha)\epsilon$$

Inequality (35) comes from (28). In the final step, we used (4).

To conclude, we are left with

$$\mathbb{E}[H_t(\pi^*)] \leq (2+\alpha)\mathbb{E}[H_t(\tilde{\pi})] + (2+\alpha)\epsilon,$$

which in turn implies the statement of the theorem. □

The following lemmas are used in the preceding proof of Theorem 4, and both their proofs follow from the standard Hoeffding inequality.

LEMMA 2. *Given $i \in [n]$ and $r \in [n]$, suppose that $X_1, X_2, \ldots, X_L$ are independent samples of the random variable $\overline{C}_t(r-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r-1} \cup i)$. Then, we have that:*

$$Pr\left(\sum_{j=1}^{L} \frac{X_j}{L} - E\left[\overline{C}_t(r-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r-1} \cup i)\right] > \frac{\epsilon}{n}\right) \leq exp(-2\frac{\epsilon^2}{n^2}L).$$

LEMMA 3. *Given $i \in [n]$ and $r \in [n]$, suppose that $X_1, X_2, \ldots, X_L$ are independent samples of the random variable $\overline{C}_t(r-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r-1} \cup i)$. Then, we have that:*

$$Pr\left(E\left[\overline{C}_t(r-1,\tilde{\pi})C_{it}([\tilde{\pi}]_{r-1} \cup i)\right] - \sum_{j=1}^{L} \frac{X_j}{L} > \frac{\epsilon}{n}\right) \leq exp(-2\frac{\epsilon^2}{n^2}L).$$

Proof of Theorem 5 Since the proof is rather similar to that of Theorem 4, we merely highlight the key details and crucial differences. First, let us verify the length of the learning phase from the theorem statement. As we did previously, we assume without loss of generality that $\log_{1+\alpha}(r_{\max}\frac{n}{\epsilon})$ is integral, and so during the course of the algorithm, τ takes on values from the set $\{r_{\max}, \frac{r_{\max}}{1+\alpha}, \frac{r_{\max}}{(1+\alpha)^2}, \ldots, \frac{\epsilon}{n}\}$. It is not hard to deduce that the cardinality of this set is $1+\log_{1+\alpha}(r_{\max}\frac{n}{\epsilon})$. Now, for any fixed value of τ and $CurrentRank$, according to Algorithm 4, we iterate over the products in U and for each $i \in U$, we display the corresponding ranking $\tilde{\pi}_i^r$ to at most L customers. Since $|U| \leq n$ at any given round, the total number of samples required before the learning phase terminates can be expressed as $(1+\log_{1+\alpha}(r_{\max}\frac{n}{\epsilon})) \cdot n \cdot L$. Substituting the value for L gives us the second half of the theorem statement.

Similar to the proof of Theorem 4, suppose that we assign product i at ranking r when the threshold was τ_r. Then, according to Algorithm 4, this was achieved by displaying a rank $\tilde{\pi}_i^r$ to L customers such that

$\tilde{\pi}_i^r(i) = r$, and ensuring that the empirical average of the revenue from the customers who got hooked by first clicking on product i was at least τ_r.

Let $R_t(\pi)$ be a random variable that denotes the revenue due to an incoming customer t when the platfrom selects a ranking π. Given ranking $\tilde{\pi}_i^r$ as mentioned above, suppose that X is a random variable that represents the revenue from a customer t if and only if the customer does not click on any of the top $r-1$ ranked products, and clicks on product i at position r; otherwise $X = 0$. Mathematically, $X = \overline{C}_t(r-1, \tilde{\pi}_i^r) C_{it}(\tilde{\pi}_i^r) R_t(\tilde{\pi}_i^r)$—note that $X = R_t(\tilde{\pi}_i^r)$ if and only if $\overline{C}_t(r-1, \tilde{\pi}_i^r) C_{it}(\tilde{\pi}_i^r) = 1$ and otherwise is 0. The expectation of this random variable is $\mathbb{E}[X] = \mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1, \tilde{\pi}_i^r) C_{it}(\tilde{\pi}_i^r) = 1\big) \sum_{i \in [n]} r_i \phi_t(i, [n])\big]$, where we note that $\overline{C}_t(r-1, \tilde{\pi}_i^r)$, $C_{it}(\tilde{\pi}_i^r)$, and $\phi_t(i, [n])$ are all defined conditional on $(\boldsymbol{p}_t, k_t, \boldsymbol{\nu}_t)$.

Let $X_1, X_2, \ldots, X_L$ denote L i.i.d instantiations of X obtained during the course of our algorithm just before product i was assigned to position r. Since product i was assigned at a threshold τ_r, we have that:

$$\frac{1}{L} \sum_{j=1}^{L} X_j \geq \tau_r.$$

We can now apply Hoeffding's Inequality to prove that the expected marginal revenue due to product i is very close to τ_r with high probability—i.e., i is a 'good' product. Formally, since $X \leq R_t(\pi) \in [0, r_{\max}]$, we invoke Lemma 5 with i.i.d random variables $(X_j)_{j=1}^L$ and $a = 0$, $b = r_{\max}$ so that with (high) probability at least $1 - \exp\left(-2L \frac{\epsilon^2}{n^2 r_{\max}^2}\right)$, the following inequality holds:

$$\mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1, \tilde{\pi}_i^r) C_{it}(\tilde{\pi}_i^r) = 1\big) \sum_{i \in [n]} r_i \phi_t(i, [n])\big] \geq \frac{1}{L} \sum_{j=1}^{L} X_j - \frac{\epsilon}{n} \geq \tau_r - \frac{\epsilon}{n}.$$

Plugging in $L = \frac{n^2 r_{\max}^2}{\epsilon^2} \log(\frac{n}{\sqrt{\delta}})$, we get the following lower bound on the expected marginal revenue due to product i with probability at least $1 - \frac{\delta}{n^2}$:

$$\mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1, \tilde{\pi}_i^r) C_{it}(\tilde{\pi}_i^r) = 1\big) \sum_{i \in [n]} r_i \phi_t(i, [n])\big] \geq \tau_r - \frac{\epsilon}{n}. \tag{36}$$

Next, for any arbitrary $\tau \in \{\frac{r_{\max}}{1+\alpha}, \frac{r_{\max}}{(1+\alpha)^2}, \ldots, \frac{\epsilon}{n}\}$, designate $U^{(\tau)}$ to be the set of unranked products when the algorithm lowers its threshold from $(1+\alpha)\tau$ to τ. For $\tau = r_{\max}$, we define $U^{(\tau)} = [n]$ for completeness. Then, for the given product i and any τ in the above range such that $i \in U^{(\tau)}$, we can apply Lemma 6 to infer that the following inequality holds with probability at least $1 - \frac{\delta}{n^2}$.

$$\mathbb{E}_t\big[Pr\big(\overline{C}_t(r'-1, \tilde{\pi}_i^{r'}) C_{it}(\tilde{\pi}_i^{r'}) = 1\big) \sum_{i \in [n]} r_i \phi_t(i, [n])\big] \leq (1+\alpha)\tau + \frac{\epsilon}{n}, \tag{37}$$

where r' is the rank at which product i did not meet the threshold of $(1+\alpha)\tau$. In fact, since we assigned product i to $\tilde{\pi}$ at rank r with threshold τ_r, we deduce that product i was not fixed at any rank prior to and including r for a larger threshold of $(1+\alpha)\tau_r$. As such, its empirical marginal revenue at that juncture (say rank $r' \leq r$) could not have been larger than $(1+\alpha)\tau_r$. By monotonicity, the same argument must hold for rank r as well. Then, (37) holds[20] with probability at least $1-\frac{\delta}{n^2}$ for rank $r'=r$ and $\tau=\tau_r$.

Let E_1 be the event that (36) holds for every position $r \in [n]$ and threshold τ_r at which a product was assigned to this position. Similarly, let E_2 denote the event that (37) is true for every given τ, $i' \in U^{(\tau)}$. Applying the union bound similarly as in the previous theorem, we can infer that $Pr(E_1=0 \cup E_2=0) \leq \delta$, as long as $\log_{1+\alpha}(r_{\max}\frac{n}{\epsilon}) \leq n$.

For the rest of this proof we will consider the regime where our empirical estimates are close to their expected values, i.e. we only consider rankings $\tilde{\pi}$ output by our algorithm such that $E_1=1 \cap E_2=1$. We know that with probability $1-\delta$, our algorithm returns such a ranking. We aim to show that in this regime, any ranking $\tilde{\pi}$ computed by our algorithm satisfies: $\mathbb{E}[R_t(\tilde{\pi})] \geq \frac{1}{2+\alpha}\mathbb{E}[R_t(\pi^*)] - \epsilon$.

Consider a fixed ranking $\tilde{\pi}$ returned by our algorithm that satisfies the conditions mentioned above. Our first claim is that:

$$\mathbb{E}[R_t(\pi^*)] \leq \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}(\pi^*)=1\big)\sum_{i\in[n]} r_i\phi_t(i,[n])\big]$$

We prove this formally in Lemma 4 after this proof. Note that $\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}(\pi^*)$ is the event that a random customer t does not click on any of the top $r-1$ ranked products under $\tilde{\pi}$ but does indeed click on the r-th ranked product under π^*. Proceeding from this point,

$$\begin{aligned}
\mathbb{E}[R_t(\pi^*)] &\leq \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}(\pi^*)=1\big)\sum_{i\in[n]} r_i\phi_t(i,[n])\big] \\
&= \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}(\tilde{\pi}^r_{\pi^{*-1}(r)})=1\big)\sum_{i\in[n]} r_i\phi_t(i,[n])\big] \\
&\leq \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \max_{i'\in[n]} \mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1,\tilde{\pi})C_{i't}(\tilde{\pi}^r_{i'})=1\big)\sum_{i\in[n]} r_i\phi_t(i,[n])\big] \\
&= \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \max_{i'\in U^{(\tau_r)}} \mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1,\tilde{\pi})C_{i't}(\tilde{\pi}^r_{i'})=1\big)\sum_{i\in[n]} r_i\phi_t(i,[n])\big] \qquad (38)
\end{aligned}$$

[20] We note that inequality (37) trivially holds for any product i that was assigned to rank r for threshold $\tau = r_{\max}$ since the right hand side would be larger than $r_{\max}$ and the expected revenue from a customer cannot exceed this.

$$\leq \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \left((1+\alpha)\tau_r + \frac{\epsilon}{n} \right). \tag{39}$$

For (38), we replaced the set $[n]$ by $U^{(\tau_r)}$, the latter being a (superset) of the set of unassigned products when the algorithm considers the rank r. Note that for all $i' \notin U^{\tau_r}$, $\tilde{\pi}(i') < r$ and so, $\overline{C}_t(r-1,\tilde{\pi})C_{i't}(\tilde{\pi}^r_{i'}) = 0$. Finally, (39) comes from (37).

Now, suppose that Algorithm 4 returns a ranking $\tilde{\pi}$, with products assigned to ranks one through $n' < n$. In this case, we can once again apply (37) and infer that for all $r > n'$, $(1+\alpha)\tau_r = \tau^{\min} = \frac{\epsilon}{n}$. We now continue with (39):

$$\begin{aligned}
\mathbb{E}[R_t(\pi^*)] &\leq \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \left((1+\alpha)\tau_r + \frac{\epsilon}{n} \right) \\
&= \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n'} (1+\alpha)(\tau_r) + \sum_{r=n'+1}^{n} (1+\alpha)(\tau_r) + \epsilon \\
&= \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n'} (1+\alpha)(\tau_r) + (n-n')\frac{\epsilon}{n} + \epsilon \\
&\leq \mathbb{E}[R_t(\tilde{\pi})] + (1+\alpha)\sum_{r=1}^{n'} \left(\mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1,\tilde{\pi})C_{it}(\tilde{\pi}) = 1\big) \sum_{i\in[n]} r_i\phi_t(i,[n]) \big] + \frac{\epsilon}{n} \right) + (n-n')\frac{\epsilon}{n} + \epsilon
\end{aligned} \tag{40}$$

$$\leq \mathbb{E}[R_t(\tilde{\pi})] + (1+\alpha)\mathbb{E}[R_t(\tilde{\pi})] + (2+\alpha)\epsilon \tag{41}$$

where (40) follows similar logic as the proof of Theorem 4 and from an application of (36). This implies the statement of the theorem. □

LEMMA 4. *Suppose that π^* denotes the optimum revenue maximizing ranking and $\tilde{\pi}$ is the ranking returned by Algorithm 4. Then, we have that:*

$$\mathbb{E}[R_t(\pi^*)] \leq \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}_t\big[Pr\big(\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}(\pi^*) = 1\big) \sum_{i\in[n]} r_i\phi_t(i,[n]) \big]$$

Proof of Lemma 4 The proof uses ideas from the proof of Theorem 1, namely:

$$\begin{aligned}
\mathbb{E}[R_t(\pi^*)] &= \mathbb{E}_t\big[\big(1 - \overline{C}_t(n,\pi^*)\big) \sum_{i\in[n]} r_i\phi_t(i,[n]) \big] \\
&\leq \mathbb{E}_t\big[\big(1 - \overline{C}_t(n,\pi^*)\overline{C}_t(n,\tilde{\pi})\big) \sum_{i\in[n]} r_i\phi_t(i,[n]) \big] \\
&= \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}_t\Big[\big\{ \big(1 - \overline{C}_t(n,\tilde{\pi})\overline{C}_t(r,\pi^*)\big) - \big(1 - \overline{C}_t(n,\tilde{\pi})\overline{C}_t(r-1,\pi^*)\big) \big\} \sum_{i\in[n]} r_i\phi_t(i,[n]) \Big]
\end{aligned}$$

$$= \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}_t\Big[\big(\overline{C}_t(n,\tilde{\pi})\overline{C}_t(r-1,\pi^*)C_{\pi^{*-1}(r)t}(\pi^*)\big) \sum_{i\in[n]} r_i\phi_t(i,[n])\Big]$$

$$\leq \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}_t\Big[\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}(\pi^*) \sum_{i\in[n]} r_i\phi_t(i,[n])\Big]$$

$$= \mathbb{E}[R_t(\tilde{\pi})] + \sum_{r=1}^{n} \mathbb{E}_t\Big[Pr\big(\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}(\pi^*)=1\big) \sum_{i\in[n]} r_i\phi_t(i,[n])\Big]. \tag{42}$$

The final step comes from the fact that conditional on t, $\overline{C}_t(r-1,\tilde{\pi})C_{\pi^{*-1}(r)t}(\pi^*) \in \{0,1\}$, and $\sum_{i\in[n]} r_i\phi_t(i,[n])$ is a constant. □

The following lemmas are used in the preceding proof of Theorem 5, and both their proofs follow from the standard Hoeffding inequality when the random variables belong to an arbitrary range $[a,b]$. As such they are generalizations of Lemmas 2 and 3.

LEMMA 5. *Suppose that $X_1, X_2, \ldots, X_L$ are independent samples of a random variable whose expectation is given by $\tilde{X}$. As long as $a \leq |X_j| \leq b$ for all $j \in [1,L]$, we have that:*

$$Pr\left(\sum_{j=1}^{L} \frac{X_j}{L} - \tilde{X} > \frac{\epsilon}{n}\right) \leq exp(-2L\frac{\epsilon^2}{n^2(b-a)^2}).$$

LEMMA 6. *Suppose that $X_1, X_2, \ldots, X_L$ are independent samples of a random variable whose expectation is given by $\tilde{X}$. As long as $a \leq |X_j| \leq b$ for all $j \in [1,L]$, we have that:*

$$Pr\left(\tilde{X} - \sum_{j=1}^{L} \frac{X_j}{L} > \frac{\epsilon}{n}\right) \leq exp(-2L\frac{\epsilon^2}{n^2(b-a)^2}).$$

Appendix B: Extensions to Revenue Maximization

Our work is motivated by settings where customers exhibit hedonic browsing behavior (window shopping) and products are not necessarily substitutable. As a result, we developed ranking policies for retailers to maximize engagement by hooking customers during their first impression window. Implicit in this framework is the notion that hooked customers provide considerable value to the retailer by means of further engagement, which includes scrolling beyond their first impression window to peruse additional products. In this section, we consider a natural extension to our model from Section 2 where the customer - upon getting hooked in the first stage - follows a browsing pattern that culminates in the purchase of at most one product. Given this two-stage model, we focus on the problem of revenue maximization and show how to adapt our learning-to-rank algorithms to achieve similar guarantees for this new objective. One can similarly extend our work to the objective of maximizing total clicks.

Formally, in our extended model, each customer t is characterized by parameters[21] $(\boldsymbol{p}_t, k_t, \boldsymbol{\nu}_t) \sim \mathcal{D}$, and each product $i \in [n]$ provides the retailer with a per-unit revenue (or profit) of r_i upon sale. In the first stage, the customer behaves exactly as outlined in Section 2, i.e., she browses the first k_t ranks and clicks on a product i with probability p_{it}. Conditional upon clicking at least one product within her first impression window and getting hooked, the customer - in the second stage of our extended model - peruses all of the products and purchases product $i \in [n]$ with probability given by a logit model with parameters $\boldsymbol{\nu}_t$, i.e., $\phi_t(i, [n]) = \exp(\frac{\nu_{it}}{\nu_{0t} + \sum_{j \in [n]} \nu_{jt}})$. Here, ν_{it} is a measure of the expected utility that customer t derives from purchasing product i. We let ν_{0t} be the utility the customer derives from the outside or "no purchase" option, and $\phi_t(\emptyset, [n]) = \exp(\frac{\nu_{0t}}{\nu_{0t} + \sum_{j \in [n]} \nu_{jt}})$ is the probability with which the customer does not purchase any of the products.

Our setup allows for heterogeneity in the underlying choice model as the parameters $\boldsymbol{\nu}_t$ are type-dependent. We make no specific assumptions on $\boldsymbol{\nu}_t$ although in practice, one would expect these parameters to be correlated with the initial click probabilities $\boldsymbol{p}_t$. Finally, the above model is more applicable to situations where the products are closely related (or substitutes) and hence, the customer purchases at most one of them.

At first glance, it would appear that our model is similar to the consider-then-choose models and other two-stage models recently studied in the assortment literature (Derakhshan et al. 2020, Aouad et al. 2015, 2019). Furthermore, Aouad et al. (2019) also consider a model where the customer's click probabilities are independent of the parameter that represents her MNL-based preference weights. However, as we describe in Section 2.1, there is a fundamental difference in consumer shopping intention and behavior between our model and other two-stage models, which leads to a divergence in technical challenges and relevant applications. Namely, our customer is unaware of the product assortment prior to shopping and thus the first stage is primarily aimed at hooking the customer, whereas in other two-stage models in the literature, customers are already hooked and the first stage is aimed at narrowing down the set of products the customer will consider for purchase. Therefore we study customers who, upon getting hooked, consider all of the products in the assortment during the second stage. Specifically, this includes the products that they did not click on in the first stage. Since the first stage centers around customers deciding whether or not to engage with the event

[21] For simplicity, we assume that the bias function $f_{it}(\chi) = 0$ for all $\chi \subseteq [n]$ although these results carry over even if $f_{it}(\chi)$ can alter the click probabilities.

itself, it is possible for customers to return to products (that they did not click on) once they have decided to engage with the event. For example, customers might revisit a product that was not clicked on if they are considering purchasing a similar product. Moreover, our model also permits cases where ν_{it} is small when the underlying click probability p_{it} is small, i.e., the customer is less likely to purchase a product which has a small click probability.

Given our focus on learning, we naturally assume that the retailer has no knowledge of the distribution $\mathcal{D}$ from which the parameters $(\boldsymbol{p}_t, k_t, \boldsymbol{\nu}_t)$ are drawn. We then focus on developing online learning policies for maximizing the retailer's total expected revenue,

$$\max_{\pi_1,\pi_2,\ldots,\pi_T} \mathrm{E}\big[\sum_{t\in\mathcal{T}} R_t(\pi_t)\big], \tag{43}$$

where the revenue earned from customer t is

$$R_t(\pi) = H_t(\pi) \sum_{i\in[n]} r_i \phi_t(i,[n]), \tag{44}$$

and the expectation over all customer types can be expressed as

$$\mathbb{E}[R_t(\pi)] = \mathbb{E}_{(\boldsymbol{p}_t,k_t,\boldsymbol{\nu}_t)}\left[\big(1-\prod_{j=1}^{k_t}(1-p_{\pi^{-1}(j)t})\big)\sum_{i\in[n]} r_i \exp\big(\frac{\nu_{it}}{\nu_{0t}+\sum_{j\in[n]}\nu_{jt}}\big)\right]. \tag{45}$$

From the above expression, it is clear that the ranking can significantly impact the retailer's revenue in the second stage: since only hooked customers proceed to purchase a product, selecting a suitable ranking increases the probability that the customer makes a purchase. At the same time, maximizing the probability that an incoming customer gets hooked *does not imply* revenue maximization because - conditional on getting hooked - the revenue generated is not equal for all customer types. We note that for a fixed customer profile $(\boldsymbol{p}_t, k_t, \boldsymbol{\nu}_t) \sim \mathcal{D}$, the expected revenue conditional on getting hooked, $\sum_{i\in[n]} r_i \exp(\frac{\nu_{it}}{\nu_{0t}+\sum_{j\in[n]}\nu_{jt}})$ is a constant that depends only on the type parameters. Therefore, the revenue maximization problem is equivalent to a weighted version of OnAR where the weight for a given customer type is its expected revenue.

We next present the Threshold Acceptance Algorithm for Revenue Maximization (Algorithm 4) as an adaptation of the Threshold Acceptance Algorithm (Algorithm 3) presented in Section 4.2. There are two key differences between Algorithms 3 and 4. First, Algorithm 4 defines Δ_{ir} to be the marginal revenue from customers who get hooked - as opposed to the marginal fraction of customers who get hooked - by first clicking on product i at rank r. Second, to empirically estimate Δ_{ir} in Algorithm 4, we display a certain ranking to $L = r_{\max}^2 \frac{n^2}{\epsilon^2}\log(\frac{n}{\sqrt{\delta}})$ customers, as opposed to $L = \frac{n^2}{\epsilon^2}\log(\frac{n}{\sqrt{\delta}})$ customers in Algorithm 3; the increased number

Algorithm 4: Threshold Acceptance Algorithm for Revenue Maximization

Input: Parameters $\epsilon, \delta, \alpha > 0$, Initial and Final Thresholds: $\tau^{\max} \geq \tau^{\min} > 0$;

Initialize null ranking $\tilde{\pi}(i) = \emptyset$ for all $1 \leq i \leq n$, and initialize $\tau = \tau^{\max}$;

Initialize unranked products $U = [n]$, $CurrentRank = 1$;

while $\tau \geq \tau^{\min}$ *and* $CurrentRank \leq n$ **do**

  for $i \in U$ **do**

    Define $\tilde{\pi}_i^r$ for $r = CurrentRank$ such that;

      a) $\tilde{\pi}_i^r$ is identical to $\tilde{\pi}$ in positions $1, ..., r-1$;

      b) $\tilde{\pi}_i^r(i) = r$;

      c) Products in $U \setminus \{i\}$ are arbitrarily assigned to positions $j > r$;

    Display ranking $\tilde{\pi}_i^r$ to $L = \frac{n^2 r_{\max}^2}{\epsilon^2} \log(\frac{n}{\sqrt{\delta}})$ customers, where $r_{\max} = \max_{i \in [n]} r_i$;

    Let S_{ir} be the total revenue from customers whose first click is product i;

    Define $\Delta_{ir} = \frac{S_{ir}}{L}$;

    if $\Delta_{ir} \geq \tau$ **then**

      Set $\tilde{\pi}(i) \leftarrow CurrentRank$;

      $U \leftarrow U \setminus i$, $CurrentRank \leftarrow CurrentRank + 1$;

    end

  end

  $\tau \leftarrow \frac{\tau}{1+\alpha}$;

end

of sampled customers for each candidate ranking - and the length of the overall learning phase - stems from the fact that the (random) revenue exhibits greater variance over a larger range, i.e., $R_t(\pi) \in [0, r_{\max}]$.

The following result shows that Algorithm 4 achieves a similar approximation guarantee as Algorithm 3 for the revenue maximization objective.

THEOREM 5. *The Threshold Acceptance Algorithm for Revenue Maximization (Algorithm 4) with* $\tau^{\max} = r_{\max} = \max_{i\in[n]} r_i$ *and* $\tau^{\min} = \frac{\epsilon}{n}$ *returns a ranking* $\tilde{\pi}$ *such that with probability* $1-\delta$

$$\mathbb{E}[R_t(\tilde{\pi})] \geq \frac{1}{2+\alpha}\mathbb{E}[R_t(\pi^*)] - \epsilon,$$

where π^* *is the optimal solution to the offline revenue maximization problem. For any* $\alpha > 0$*, the length of the learning phase is* $O\left(r_{\max}^2 \frac{n^3}{\epsilon^2} \log_{1+\alpha}(\frac{nr_{\max}}{\epsilon}) \log(\frac{n}{\sqrt{\delta}})\right)$.

We present the proof in Appendix A and note that all other algorithms and corresponding performance guarantees for both the online and offline settings can similarly be extended to the revenue maximization objective. We note that instead of focusing on revenue, one could alternatively maximize the total number of clicks by formulating an analogous two-stage model where hooked customers click on multiple products across the assortment in the second stage. Given the similarities between the two models, it is possible to adapt our algorithms and analysis to derive comparable results for this objective of maximizing total clicks.

Appendix C: Sensitivity Analysis for Simulations

In this appendix, we perform sensitivity analysis on the default parameter set (recommended by Wayfair) that we used to generate customer preference vectors from observed clickstream data, as well as on the number of customers to offer each ranking during the learning phase (i.e. sample size, L); all results in this appendix pertain to the Threshold Acceptance Algorithm as implemented in Section 5. First, we varied the percent of customers with non-zero preferences between 70-90% (specifically, {70%, 80%, 90%} - recall that the main results presented are for 80%). Figure 5 presents our sensitivity analysis on the percent of customers with non-zero preferences. The analysis illustrates that our results are robust to this parameter choice, and in most cases, that our algorithm performs marginally better as the percent of customers with non-zero preferences increases. Intuitively, when there are more customers with non-zero preferences, our algorithm has more opportunity to hook additional customers compared to Wayfair's ranking where the number of hooked customers is fixed. In reality, it is impossible to know the percent of customers with non-zero preferences, and our results show that over a reasonable and wide parameter range, our algorithm still attains significant performance improvement over Wayfair's ranking.

Second, we varied the percent of customers who view all n products between 2.5-10% (specifically, {2.5%, 5%, 7.5%, 10%} - recall that the main results presented are for 5%). When the percent of customers who view all products was 2.5% or 10%, we were unable to find a power law distribution to generate customers

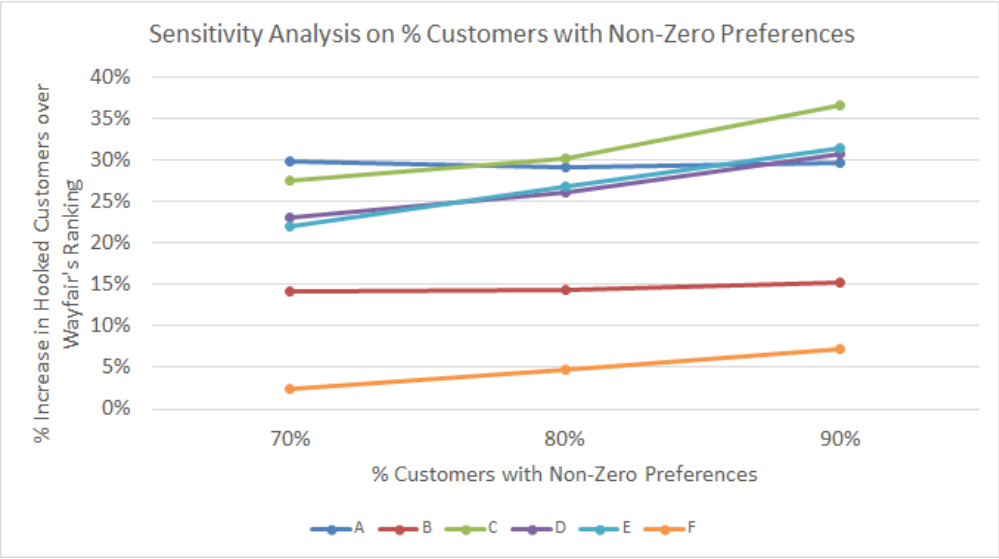

Figure 5 Sensitivity Analysis on Percent of Customers with Non-Zero Preferences

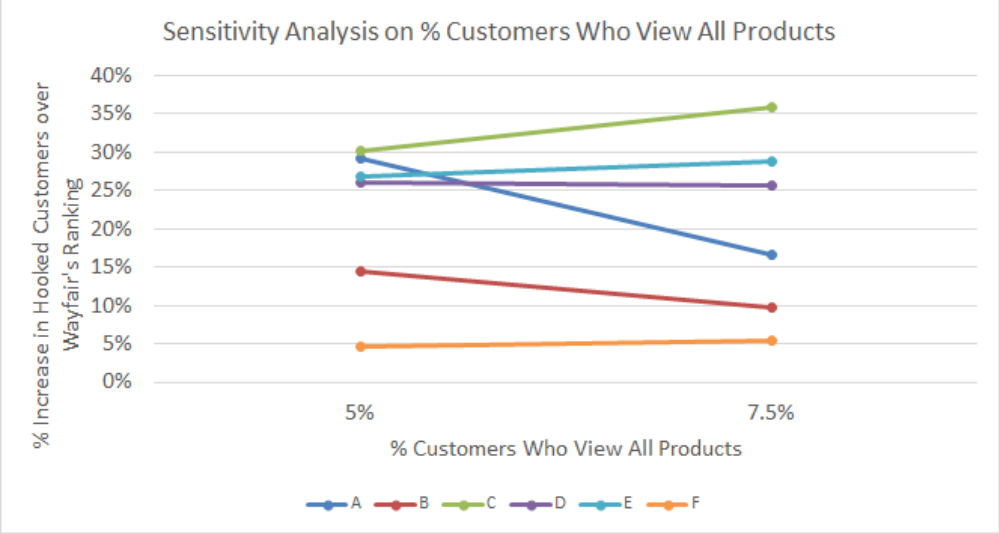

Figure 6 Sensitivity Analysis on Percent of Customers Who View All Products

that recovered the actual clickstream data for some of the events, i.e. we could not find matches between actual and simulated clicks as we did in Figure 2; thus, these parameter values are likely unrealistic. Figure 6 presents our sensitivity analysis on the percent of customers who view all products. The analysis illustrates that our results are robust to this parameter choice. In reality, it is impossible to know the percent of customers who view all products, and our results show that over a reasonable parameter range, our algorithm still attains significant performance improvement over Wayfair's ranking.

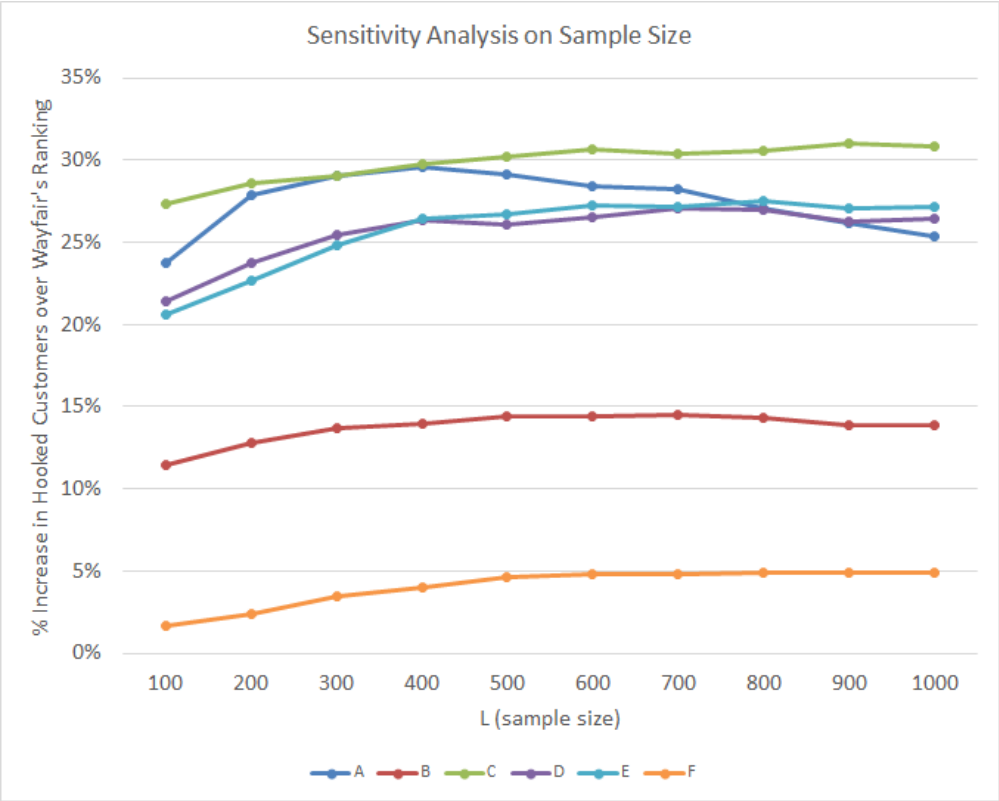

Figure 7 Sensitivity Analysis on Sample Size, L

Third, we varied the number of customers for which to offer each rank (L) during the learning phase such that $L \in [100, 1000]$, allowing us to evaluate the tradeoff between faster vs. more accurate learning. Figure 7 illustrates that when the sample size is small, there is insufficient learning and the algorithm's performance suffers. As the sample size increases up to approximately 400 or 500 customers, the algorithm is able to sufficiently learn the (approximate) best ranking. For the smallest event (A), we see evidence that as the sample size grows too large, the algorithm is unable to learn fast enough to reap the benefits of more precise learning. Interestingly, we see that for all but the smallest event, the performance of our algorithm is remarkably constant for a wide range of sample sizes. This is a desirable artifact since Wayfair can simply choose a single sample size for the algorithm's implementation across all events - likely $L = 400$ or 500 - without having to estimate the event size T *a priori*.

Appendix D: Inferring Customer Behavior from Click Data

In Section 5 of the main paper, we presented a new methodology for using the clickstream data to learn the distribution $\mathcal{D}$ of customer preferences and first impression windows. This methodology relied on the customer behavior model introduced in Section 2 (e.g. customers who are not hooked within their first impression window exit the event) and additional assumptions, e.g., hooked customers go on to view the entire assortment. Putting aside the data, we then generated customers from the inferred distribution and simulated their behavior on the ranking that Wayfair had used for the original event. Assuming that the customers follow the model in Section 2, we then compared the fraction of customers who click on each product in the simulated models to that of the true data, taking into account the entire dataset. The close fit between the simulated and actual distributions that we see in Figure 2 indicates that our model is a good predictor of customer behavior on actual events.

In this Appendix, we further examine the predictive power of our customer behavior model as well as others proposed in the literature. First, we present an *out-of-sample fit*, wherein we learn the customer preference distribution $\mathcal{D}$ using a fraction (70%) of the clickstream data and compare the click profile of customers generated from this distribution to the true clicks from the remaining 30% of the customers (holdout data). This serves as a contrast to the methodology leading to Figure 2 in Section 5.2, which could be interpreted as an in-sample fit since all of the customer data was utilized to learn the distribution $\mathcal{D}$ and the simulated clicks were compared to the true clicks arising from the entire dataset. Our goal in Section 5.2 was to re-create customer profiles in a manner that is consistent with Wayfair's actual events to generate the most realistic customers to test our algorithms; thus we opted to utilize all of the data in doing so. Here in this Appendix, our goal is to show the effectiveness of our customer profile inference process and thus we use a 70/30 train/test split to evaluate its accuracy on a holdout sample.

Second, we investigate whether some of the other models from the literature (see Sections 1.1 and 2.1) could explain the observed click data for the six Wayfair events. We consider the cascade model (Kveton et al. 2015a) - single customer type with regards to intrinsic click probabilities, uniform first impression window k, no bias - and position-based model (Lagrée et al. 2016) - single customer type with regards to intrinsic click probabilities, variable first impression window k_t, no bias. We devise a simple methodology for deriving model parameters that best explain the observed data for each of these models and show the resulting fit would lead to large errors. We also rule out frameworks where customer clicks follow a choice

model (e.g., consider-then-choose models) by providing evidence that the products in Wayfair's events tend to be diverse rather than substitutable.

Taken together, the results in this appendix strongly suggest that while our framework is able to suitably explain customer clicks even using out-of-sample data, other models studied in the literature do not fully account for all facets of the observed customer behavior.

D.1. Out-of-Sample Model Fit

Since the methodology used to derive the out-of-sample model comparison is very similar to that used to obtain the in-sample plot in Section 5.2, we simply highlight the key differences and the main parameter choices. Recall that $\mathcal{T}$ denotes the entire click data for any one of the events. We infer the model parameters and compare against the out-of-sample data as follows:

1. **Data Split**: Randomly split $\mathcal{T}$ into training data $\mathcal{T}^{\text{train}}$ and test data $\mathcal{T}^{\text{test}}$ such that $|\mathcal{T}^{\text{train}}| = 0.7|\mathcal{T}|$ and $|\mathcal{T}^{\text{test}}| = 0.3|\mathcal{T}|$.

2. **Conditional Inference Process**: Assume that the first impression windows follow a power-law distribution, i.e., $Pr(k_t = r) = 0.95\frac{ar^{-b}}{\sum_{l=1}^{n-1} al^{-b}}$ $\forall r = 1, ..., n-1$ and that $Pr(k_t = n) = 0.05$. For any given distribution of first impression windows $\mathcal{D}_k$, learn the distribution $\mathcal{D}_p$ of interest probabilities using the exact same procedure outlined in Section 5.2 but using only the data in $\mathcal{T}^{\text{train}}$ for the inference process.

3. **Minimizing Error**: Optimize for $\mathcal{D}_p$ and $\mathcal{D}_k$ in order to minimize the error in the percentage of customers hooked - upon viewing Wayfair's ranking - between the training data and the inferred distribution. The fraction of customers hooked due to the inferred distribution can be estimated by sampling $|\mathcal{T}^{\text{train}}|$ customers with preferences drawn from $\mathcal{D}_p$ and first impression windows from $\mathcal{D}_k$, and identifying the subset of the sampled customers who would get hooked when exposed to Wayfair's original ranking.

4. **Compare against Test Data**: Sample $|\mathcal{T}^{\text{test}}|$ customers from $\mathcal{D}_p$, $\mathcal{D}_k$, simulate their response when exposed to Wayfair's Ranking and calculate the fraction of simulated users who click on each individual product in the assortment. Repeat Step 4 one hundred times to help eliminate noise in individual draws from the inferred distributions $\mathcal{D}_p, \mathcal{D}_k$. Compare to the fraction of customers in $\mathcal{T}^{\text{test}}$ who click on each product.

Figure 8 compares the click probability for each product in the test data versus those of the simulated customers (averaged over 100 simulations) for each of the six events. For the vast majority of products across the six events, the simulated and actual click probabilities are very similar. This highlights the effectiveness

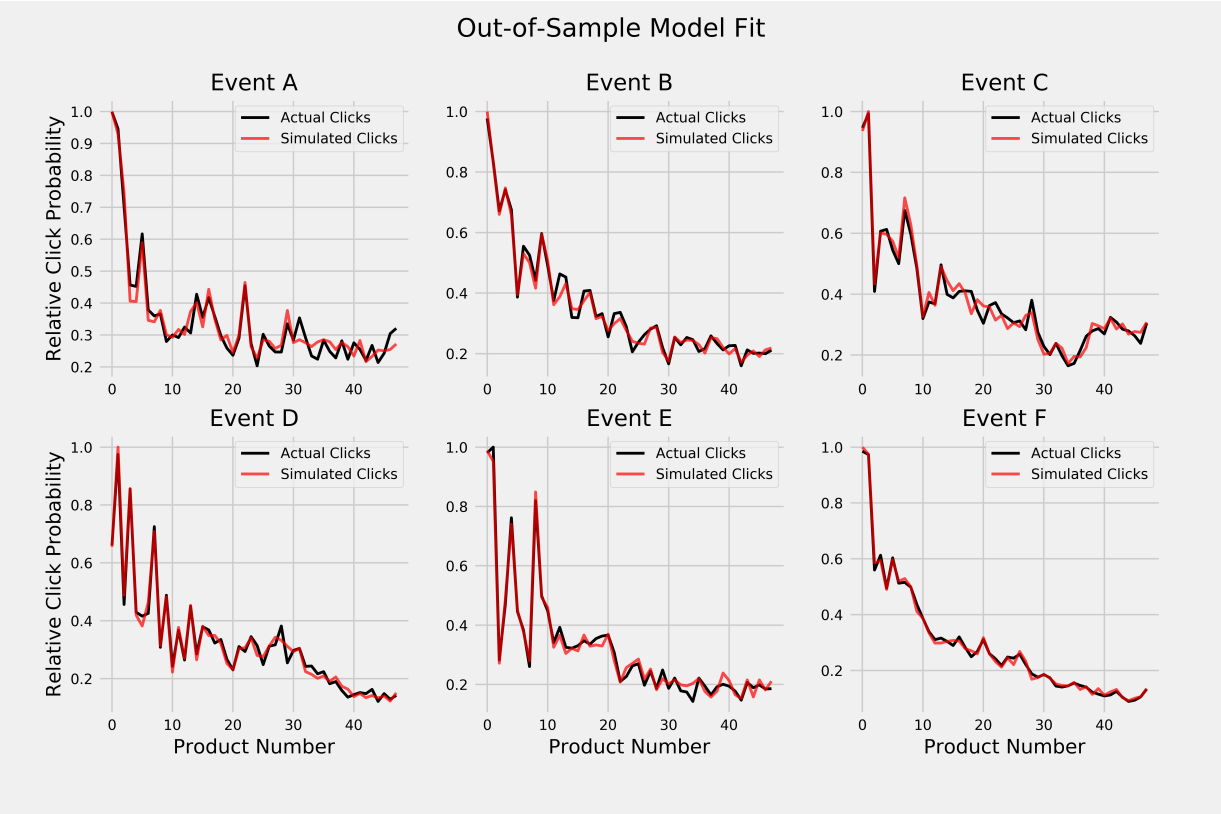

Figure 8 Actual and simulated relative click probabilities for six events, where "actual clicks" are based on the test data and "simulated clicks" are from a model whose parameters were inferred using the training data. The click probabilities are normalized in each event for confidentiality so that the maximum click probability of any product is one.

of our customer profile inference process used in Section 5.2 and provides further evidence that our model proposed in Section 2 is a good predictor of customer behavior on actual events. Finally, we observe from Figure 8 that our learned models incur marginally smaller error for events D, E, and F - i.e., when compared to events A, B, and C, the simulated customer profiles for these events are a better fit to the true click probabilities. This is fundamentally an artefact of the size of the test dataset since the events are labeled in increasing order of their population (event A has the highest error and smallest test data, and so on).

D.2. Fitting Data to Other Models

In the following subsections, we investigate whether the cascade model, position-based models, and choice models could explain the observed click data for the six Wayfair events.

D.2.1. Cascade Models: We analyze whether the popular cascade model studied in the literature (e.g., Kveton et al. (2015a) and citations therein) suitably captures customer behavior in the six Wayfair events.

The cascade model is a special case of the more general framework proposed in Section 2, wherein there is only a single customer type with a fixed but unknown set of click probabilities for each item, a known first impression window k identical for all customers, and $f_{it}(\chi) = 0$ for all i, t and $\chi \subseteq [n]$; to be specific, every customer t has the same vector of click probabilities over the products $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ and views the exact same number of products $k \in [1, 48]$ during her first inspection of the event. As with our model, if the customer clicks on at least one of the first k products, we assume that they are hooked; otherwise they exit the event. Our goal in this section is to fit the clickstream data from the six Wayfair events to a cascade model by identifying the parameters $(\boldsymbol{p}, k)$ that best explain the data. Following this, we evaluate the fit quality by looking at two metrics of interest when customers are exposed to Wayfair's actual ranking - (i) the fraction of customers hooked, and (ii) the fraction of customers who click on each product.

We begin by pointing out that there are two particular approaches in which one could fit a cascade model to our dataset. First, the clickstream data shows us that a non-trivial fraction of the customers' first click corresponds to each of the top-48 ranked products. In the face of this observation, it seems reasonable to deduce that the only possible value of the first impression window k that *fully* explains the data is $k = 48$. For the second approach, since fewer customers click on products in later ranks, one could alternatively assume that the first impression window $k < 48$, and any clicks on later products stem from hooked customers; note that this approach cannot fully explain the data since some customers first click on a product after rank k for any $k < 48$, i.e., their first impression window must have been more than k. For each of these two approaches, we develop an inference methodology and test whether a cascade model suitably predicts the observed customer behavior in our clickstream data.

Cascade model with $k = 48$: Fixing the first impression window to $k = 48$, our goal here is to identify the click probabilities for each product, $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$, that best explain the clickstream data. Since all customers look at all of the products in the assortment when $k = 48$, we can infer that for any given product, its click probability should match its empirical click probability in the data. Formally, for product i, the most likely value of its click probability is given by:

$$p_i \triangleq \frac{\text{Number of customers who click on product } i}{\text{Total number of customers } (|\mathcal{T}|)}. \tag{46}$$

In simple terms, this is the only value of p_i that ensures that the (expected) fraction of customers who click on product i in our simulated instances matches the true fraction observed in the data. Note that this definition leverages the cascade model's assumption that customers' clicks across products are independent.

Fit Accuracy: Cascade Model, $k = 48$	
Event ID	**Error in %Hooked**
A	58.0%
B	59.1%
C	56.6%
D	48.7%
E	52.3%
F	74.4%

Table 3 Error in percentage of hooked customers when customers are generated based on a cascade model with $k = 48$ inferred from the click data. Error is computed as $\frac{\%\text{Hooked in Simulations} - \text{Actual } \%\text{Hooked}}{\text{Actual } \%\text{Hooked}}$.

To evaluate the fit accuracy of this model for each of the six events, we generate $|\mathcal{T}|$ customers using the inferred click probabilities from (46), and we examine the fraction of generated customers that would be hooked by Wayfair's ranking vs. the actual fraction of customers hooked; results are presented in Table 3. For each of the six events, the simulated customers generated using the inferred click probabilities become hooked at a significantly higher rate than the actual customers in the event - in fact, over 50% more customers are hooked in the simulations. This observation provides a strong case for customers having first impression windows smaller than the length of the entire assortment, but also heterogeneous window sizes to account for those whose first click occurs at a later rank. Overall, the large error rate in the percentage of hooked customers leads us to conclude that cascade models where customers view the entire assortment do not accurately reflect customer behavior in our dataset.

Cascade model with best-fit k ($k \in [1, 48]$): Although the cascade model with $k = 48$ provided a near-perfect fit for the fraction of customers who click on each product, it led to very large errors in terms of the percent of customers hooked. Thus, we next seek to jointly learn both the click probabilities $\boldsymbol{p}$ and first impression window k so that the fraction of hooked customers in our simulations matches the fraction hooked in the dataset. Our inference process consists of two steps. First, we iterate over all values of the first impression window k in the range $[1, 48]$. Second, upon fixing a value of k for each event, we compute the vector $\boldsymbol{p}^{(k)}$ that achieves the best possible fit for the observed click distribution across products, conditional on the first impression window being k. Finally, for each of the forty-eight $(\boldsymbol{p}^{(k)}, k)$ pairs that we inferred for each event, we simulate $|\mathcal{T}|$ customers with these parameters and estimate the fraction who would have been hooked by Wayfair's ranking. We then select the parameters $(\boldsymbol{p}^{(k)}, k)$ for each event such that the fraction hooked is closest to the value of this metric in the actual data.

Finally, we touch upon our approach for setting the probabilities $\boldsymbol{p}^{(k)}$ for a fixed value of k. Since all customers view the first k ranked products, we can choose the click probabilities for these products exactly as we did for the $k = 48$ case. That is, for all products i such that their position under Wayfair's ranking is smaller than or equal to k, its probability $p_i^{(k)} \triangleq \frac{\text{Number of customers who click on product } i}{\text{Total number of customers}(|\mathcal{T}|)}$. For any product i whose position in Wayfair's ranking is strictly greater than k, its click probability is simply its empirical click probability among hooked customers, i.e. those who clicked on a product at or before rank k and thus browsed product i:

$$p_i^{(k)} \triangleq \frac{\text{Number of hooked customers who click on product } i}{\text{Total number of hooked customers}}$$

The above formula also implicitly assumes that once a customer is hooked, they go on to view all 48 products, consistent with our simplifying assumption in Section 5.

Table 4 presents the value of the first impression window k that minimizes the error in the percentage of hooked customers, along with a relative measure of this error, for each of the six events under consideration. Given that we tune the first impression window k to minimize the error in the percentage of hooked customers, it is not surprising that this model achieves a close fit with the original data for $k < 48$. That said, when we look at the fraction of simulated customers who click on each product i and compare with the actual fraction of customers who click on each product, we see that for every product i that is ranked beyond the customers' first impression window k, the fraction of customers who click on this product in our simulations is significantly (by $10-50\%$) smaller than the true clicks received by this product (Figure 9). This poor fit in the click profile beyond the first impression window is consistent across all six events. As we argued earlier, this mismatch stems from the heterogeneity in preferences, and the fact that many customers do have large first impression windows (e.g., $k > 40$), which are not captured by the best-fit cascade model.

To conclude, we presented two methods for fitting our data to a cascade model. When we assume $k = 48$ in order to attain per-product click probabilities that match the original customers, the generated customers are hooked at a significantly higher rate compared to the actual data. While lowering the first impression window resolves this issue, the resulting click probabilities for products beyond the first impression window no longer match their true values. Compared to this, our model in Section 2 achieves a close-fit, both in terms of the fraction of customers hooked, and the individual click probabilities across all 48 products. Thus we conclude that our customer behavior model is a better fit for the actual data than the cascade model.

Fit Accuracy: Cascade Model with Best-Fit k		
Event ID	**Error in %Hooked**	**First Impression Window (k)**
A	0.9%	23
B	1.1%	19
C	1.4%	23
D	1.0%	24
E	0.7%	21
F	1.0%	13

Table 4 Error in percentage of hooked customers when customers are generated based on a cascade model with best-fit k inferred from the click data. The first impression window is chosen to minimize error, which is computed as $\left|\frac{\text{\%Hooked in Simulations}-\text{Actual \%Hooked}}{\text{Actual \%Hooked}}\right|$.

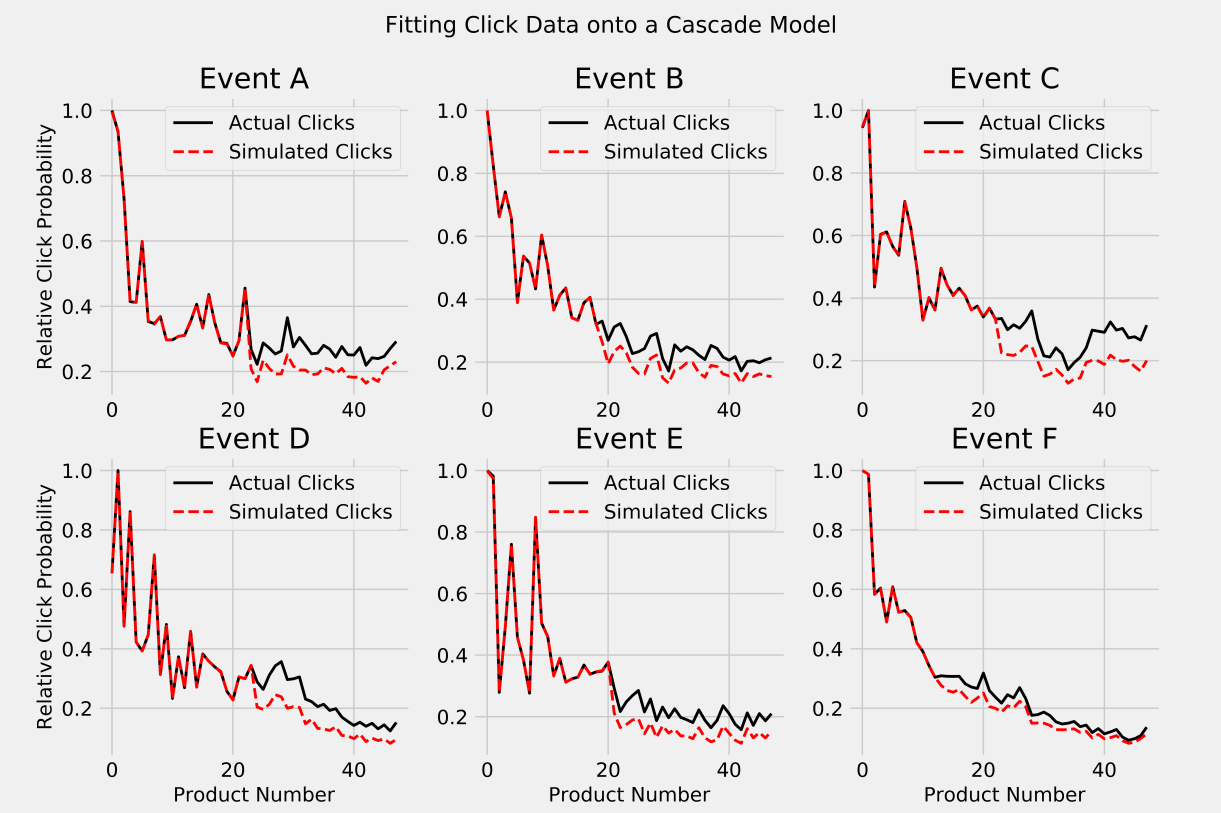

Figure 9 Actual and simulated relative click probabilities for all six events when the customers are generated from a best-fit cascade model with $k \in [1, 48]$. The click probabilities are normalized for confidentiality in each event so that the maximum click probability of any product is one.

D.2.2. Position-Based Models: Given that cascade models do not fully capture the heterogeneity in first impression windows, it is natural to consider whether a position-based model as in Lagrée et al. (2016) can fully predict the customer behavior in our data. Formally, position-based models are a special case of

our more general model in Section 2 - in the former, all customers have the same click probabilities, their first impression window is drawn from an independent distribution, and there are no bias functions. In this section, we devise a methodology for inferring the parameters of a position-based model, namely the click probabilities $\boldsymbol{p}$ and distribution $\mathcal{D}_k$ from which the first impression windows are drawn. Following this, we generate customers with these preferences and compare the percentage hooked and the click profile across the assortment to Wayfair's data, as we had done for the cascade model.

The methodology used to fit the data to a position-based model is somewhat similar to the more general inference framework that we developed in Section 5 but accounting for a single customer type (with regards to intrinsic click probabilities). Suppose that the products are labeled in the order of their position in Wayfair's static ranking π^{WF}, and so $\pi^{WF}(i) = i$ for all $i \in [1, 48]$. We begin with the assumption that the first impression windows are drawn from a power law distribution $\mathcal{D}_k$, i.e., $Pr(k_t = r) \propto ar^{-b}$ for all $r \in [1, 47]$ and tune parameters (a, b). In addition, we assume that a small fraction of the customers view the entire assortment (i.e., have $k_t = 48$). Both these assumptions ensure consistency with our simulations in the main paper. Similar to our methodology in Section 5, we begin by guessing the distribution $\mathcal{D}_k$ for the first impression windows - i.e., initial parameters (a, b). Since all customers view the first ranked product ($k_t \geq 1$ for all t), its click probability is equal to its empirical click probability, i.e., the fraction of customers in the data who click on the product at the first position. For every other product i, we set its click probability as follows:

$$p_i \triangleq \frac{\text{Number of customers who click on product } i \text{ and product 1}}{\text{Total number of customers who click on product 1}}. \tag{47}$$

The above formula for the click probabilities stems from the notion that in position-based models, customers' clicks on different products are independent of each other. Therefore, the fraction of customers who click on product i conditional upon clicking on the top-ranked product must equal the overall fraction of customers who would click on product i if they were to view it. Using data only from customers who click on the top-ranked product ensures that we are only considering customers who are guaranteed to have viewed product $i > 1$. This relies on the assumption that we made in Section 5, namely that hooked customers view all of the products in the assortment.

For the given set of parameters $(\boldsymbol{p}, \mathcal{D}_k)$, we then generate customers and compute the fraction that would be hooked if exposed to Wayfair's ranking. Finally, as in Section 5, we tune the parameters of the power law

distribution (a, b) to minimize the error (between the generated customers and dataset) in the fraction of hooked customers.

We now discuss the accuracy of the fit and the predictive power of position-based models for the same two metrics used previously. First, we note that our methodology is able to achieve a close fit to the original data in terms of the percentage of customers hooked. As is the case with our results in Section 5, this in itself is not particularly surprising given the degrees of freedom available to us in setting the power law parameters. Therefore, we focus mainly on the clicks generated by the simulated customers. Figure 10 compares the fraction of generated customers who click on each individual product to the true fraction in the underlying data. Looking at the plot, we see that the customers generated from a position-based model are significantly more likely to click on products in positions $[2, 10]$ than the actual customers. This large mismatch in the click probabilities is also consistent across all six events. On the other hand, for products that are ranked beyond position ten, the simulated clicks reasonably match the actual click probabilities. In light of our inference method (see Equation (47)), we argue that customers who click on the top-ranked product are much more likely to click on (say) the second or third ranked products than an average customer. Further, upon examining the clickstream data, we verify that this holds independent of the distribution used for the first impression windows. These findings provide a strong argument in favor of modeling multiple customer types, and against independence in clicks, a common assumption in much of the literature. Indeed, our model in Section 2 is able to accurately mirror the click probabilities across all products, as can be seen in Figure 2. Finally, Figure 10 gives the impression that more customers are hooked in our simulations than in the underlying dataset. This however, is not true; the additional clicks by the customers generated from the position-based model stems from the fact that these simulated customers tend to click on more products once they are hooked.

In summary, even though position-based models allow for heterogeneity in first impression windows, our findings indicate that having only one customer type (with regards to intrinsic click probabilities) is not sufficient to capture the rich behavioral patterns exhibited by customers who view Wayfair's events. We believe that this finding would generalize to many other settings where the products are not substitutable and customers exhibit hedonic browsing behavior. It is worth noting that for both the cascade and position-based models, the optimal ranking is a popularity ranking, which is certainly not the case for Wayfair's events as can be gleaned from our results in Section 5.

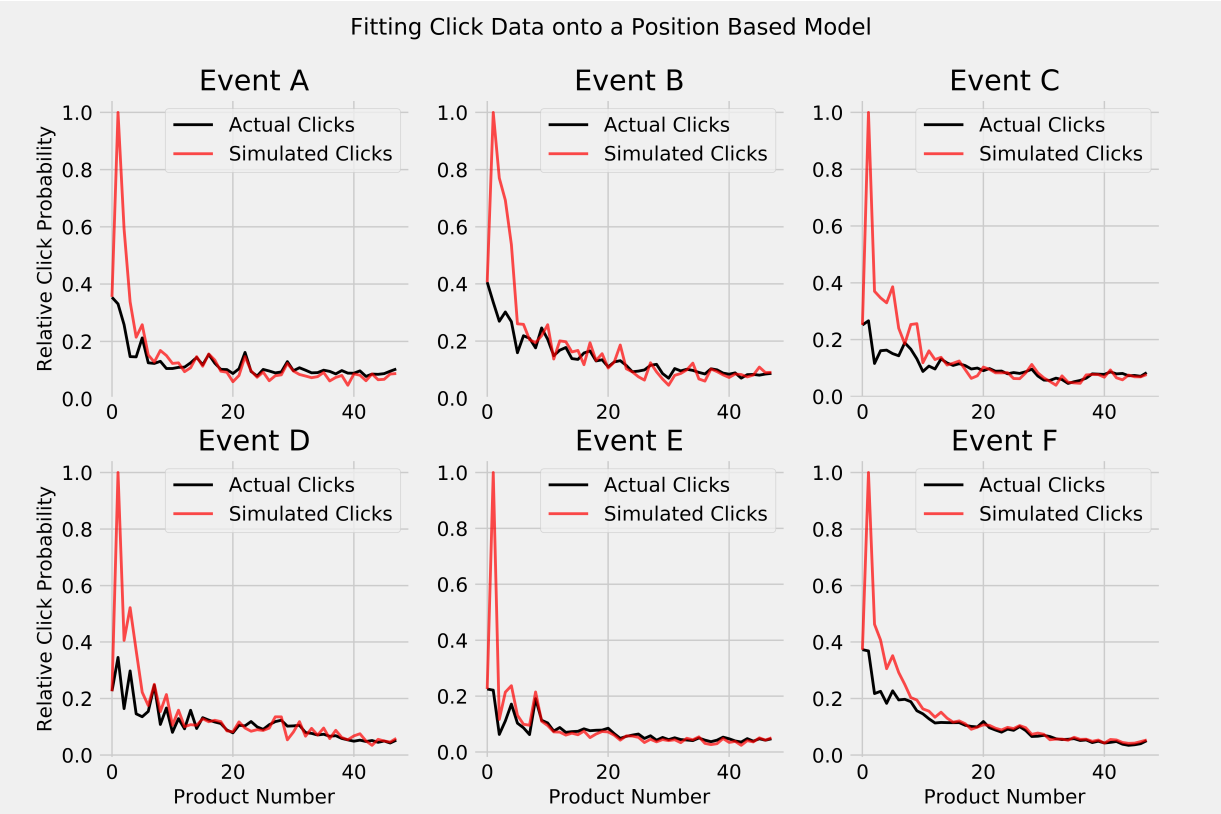

Figure 10 Actual and simulated relative click probabilities for all six events when the customers are generated from a best-fit position-based model with first impression windows drawn from a power law distribution. The click probabilities are normalized for confidentiality in each event so that the maximum click probability of any product is one.

D.2.3. Choice Models: Choice models are commonly used as customer behavior models in much of the operations management and marketing literatures, including many papers in the product ranking literature (Table 1). By construction, choice models assume that each customer chooses at most one product from a set of substitutable products offered by the retailer. Here, we illustrate that the assortment of products offered in each of Wayfair's six events cannot be considered substitutes, as one way to illustrate that choice models are not appropriate in this setting.

Table 5 displays the number of product categories that were offered in each event; each product category represents a set of products that Wayfair's merchandising team considers to be substitutable, at least from a functional perspective[22] (e.g., floor lamps and ceiling lights could constitute two separate product categories

[22] We note that the delineation of sets of substitutable products likely extends beyond simply a functional perspective, but nonetheless this is a reasonable proxy for understanding whether or not the products in an event are substitutable.

Event	A	B	C	D	E	F
Number of product categories offered	29	12	25	15	18	5

Table 5 Number of Product Categories Offered in Each Event

in one of the events). Given the large number of categories offered in most events, it is clear that products offered in an event cannot be considered substitutes and thus choice models would not accurately represent customer shopping behavior in our setting.

Appendix E: Additional Simulations: Finite Customer Types

In Section 5, we presented our main empirical results, where we evaluated the performance of our Simple Learning and Threshold Acceptance Algorithms on customer preferences inferred based on actual clickstream data from six distinct events. Despite the rather conservative assumptions made in that section, we observed that our proposed algorithms outperformed the default static ranking employed by Wayfair by moderate amounts (average improvement between 5 and 30%; refer to Figure 3). In this section, we adopt an alternative and less-conservative methodology where we assume that customers belong to a finite number of types and use the clickstream data to infer the types and the probability that an incoming customer belongs to each of these types. We then use the derived customer preference distribution to estimate the performance of our Threshold Acceptance Algorithm[23] in comparison to Wayfair's static ranking as well as the omniscient offline benchmark as defined earlier (the static ranking output by the Greedy Algorithm for OffAR, π^g).

We begin by highlighting a few of the restrictive assumptions from Section 5 below and outline our reasoning for relaxing these assumptions. Following this, we present the technical details surrounding the implementation. A comprehensive description of our partner Wayfair, customer behavior on the site, characteristics of the six stylistically-themed events that were chosen for these simulations, and the clickstream data can be found in Section 5.1, and we avoid repeating these details here.

In the process of learning customer preferences and first impression windows from the clickstream data, our implementation in Section 5 required the following behavioral assumptions. First, our inference was based on the premise that once a customer was hooked, she views all 48 products in the event. Consequently, each

[23] The performance of our Simple Learning Algorithm is very similar to that of the Threshold Acceptance Algorithm in Section 5, so we have omitted for brevity.

unique click vector was taken to represent a distinct customer type;[24] this resulted in each event catering to anywhere between 3500 and 14000 customer types. Second, the customer interest probabilities were assumed to be independent of their first impression windows. Third, for each customer t and product i, we had only integral click probabilities, i.e., $p_{it} \in \{0,1\}$. These assumptions are somewhat conservative as they tend to be inimical to the performance of our algorithm. For example, the premise that hooked customers view all 48 products severely limits the customer's response to any alternative ranking as the customer is assumed to not be interested in any product that she did not click on.

In the following simulations, we make a different set of assumptions regarding customer interests and behavior and similarly compare our algorithm to Wayfair's static ranking and offline benchmark:

- As in Section 5, we assume that $\boldsymbol{f}_t(\chi) = 0$ for all t.
- We suppose that there is a finite set $\mathcal{Z}$ of customer types, where each type $z \in \mathcal{Z}$ is specified by its parameters $(\boldsymbol{p}^{(z)}, \mathcal{D}_k^{(z)}, q^{(z)})$. Crucially, the distribution of customer first impression windows is not independent of product interest. This is a natural consideration in e-commerce domains, e.g., the demographics of customers may skew both their product preferences and their first impression windows. Finally, we assume that the average number of products per type that a customer has non-zero interest in is two. Mathematically, we have:

$$\frac{1}{|\mathcal{Z}|} \sum_{z \in \mathcal{Z}} \sum_{i \in [n]} \mathbb{1}\{p_i^{(z)} > 0\} = 2,$$

where $\mathbb{1}\{\}$ is the indicator variable that is one if the condition inside is true and zero otherwise. Note that some types may have only one product and others may have many.

- There exists a customer click probability parameter $p \in (0,1)$ such that $p_{it} \in \{0, p\}$ for all customers t and products $i \in [1,n]$. A simple interpretation for this parameter is that a customer t who is interested in a product i that falls within her first impression window only clicks on this product with some finite probability p and does not click on this product with probability $1-p$.
- Customer t only clicks on products displayed in ranks $[1, k_t]$, regardless of whether or not they are hooked. This assumption is in contrast with the one made in Section 5 that hooked customers browse and

[24] Formally, a preference distribution $\mathcal{D}$ can be partitioned into mutually exclusive customer types, where each type z is characterized by an interest probability vector $\boldsymbol{p}^{(z)}$, first impression window distribution $\mathcal{D}_k^{(z)}$ and a type probability $q^{(z)}$. An incoming customer t belongs to type z with probability $q^{(z)}$ and conditional upon her type being z, her click vector is $\boldsymbol{p}_t = \boldsymbol{p}^{(z)}$ and first impression window k_t is drawn independently from $\mathcal{D}_k^{(z)}$.

can click on products in the entire assortment. In reality, customer behavior and the performance of our algorithm may fall somewhere in-between.

E.1. Implementation

In Section 5, the primary difficulty in the inference process was to identify the (censored) interest probability vector $\boldsymbol{p}_t$ for customers who did not click on any product by assigning them to one of the click vectors corresponding to the hooked customers. Here, the challenge is two-fold: first, we need to infer a representative set of customer types including their preferences and first impression windows; second, we need to assign each of the customers to one of the types based on the observed click vector. Once again, the second part is compounded by the censored nature of the observations for all customers (hooked or otherwise).

We now outline the methodology used for constructing the empirical distribution $\hat{\mathcal{D}}$ of customer types based on the clickstream data. Where it is applicable, we use the same notation introduced in Section 5, e.g., $\boldsymbol{c}_t$ represents the click vector for customer t, $\tilde{\mathcal{T}}_c$ and $\tilde{\mathcal{T}}$ are the customers with non-zero clicks and non-zero preference vectors respectively, and so on. To reiterate, our goal is to use the set of customer click vectors $(\boldsymbol{c}_t)_{t\in\mathcal{T}}$ to construct the customer types that make up the distribution $\hat{\mathcal{D}}$, i.e., $(\boldsymbol{p}^{(z)}, \mathcal{D}_k^{(z)}, q^{(z)})_{z\in\mathcal{Z}}$.

1. *Fixing parameters a priori*: As we did in Section 5, we assume that the total number of customers with non-zero preferences $|\tilde{\mathcal{T}}|$ is fixed, as is the customer click probability parameter p and the number of types $|\mathcal{Z}|$. We will vary the parameter p within an appropriate range to recover the customer types that best fit the observed click data. The exact choice or range of the other parameters is discussed subsequently.

2. *Clustering to partition click vectors into types*: We identify a subset of the customers in $\tilde{\mathcal{T}}_c$ who clicked on two or more products ($\sum_{i\in[n]} c_{it} > 1$) and run a standard k-means++ (Arthur and Vassilvitskii 2007) algorithm to cluster these click vectors into $|\mathcal{Z}|$ partitions.

3. *Identifying representative set of interest vectors* $(\boldsymbol{p}^{(z)})_{z\in\mathcal{Z}}$: We extract the centroids of each cluster $(\boldsymbol{C}^{(z)})_{z\in\mathcal{Z}}$, which are fractional and tend to be overly dispersed as each centroid has non-zero entries corresponding to many products. Thus, we take each product $i\in[n]$ and assign it to the w clusters corresponding to the w largest values in $(C_i^{(z)})_{z\in\mathcal{Z}}$. Therefore, $p_i^{(z)} = p$ if z is one of the top-w clusters for product $i\in[n]$ as per the centroid, and $p_i^{(z)} = 0$ otherwise. Finally, the parameter w is chosen in order to satisfy our assumption that the average number of products that a customer type is interested in is two, i.e., $w = \frac{2|Z|}{48}$.

4. *Assigning customers (click vectors) to types:* Now that we have derived the interest vectors $\boldsymbol{p}^{(z)}$ for each type, we need to infer the first impression window distributions as well as type probabilities. As a first step,

we take each click vector belonging to the customers in $\tilde{\mathcal{T}}_c$ and assign it to one of the types using a maximum likelihood technique. Specifically, consider a click vector $\boldsymbol{c}_t$ and let r_t denote the position of the largest ranked product that the customer clicks on. For every type $z \in \mathcal{Z}$, customer $t \in \tilde{\mathcal{T}}_c$, and product $i \in [r_t]$, we define a probabilistic score $s_{it}^{(z)}$ as follows: (i) $s_{it}^{(z)} = p$ if $c_{it}, p_i^{(z)} > 0$; (ii) $s_{it}^{(z)} = 1-p$ if $p_i^{(z)} > 0 = c_{it}$; (iii) $s_{it}^{(z)} = \eta$ if $p_i^{(z)} = 0 < c_{it}$ and (iv) $s_{it}^{(z)} = 1$ if $p_i^{(z)} = c_{it} = 0$. In the above definition $\eta < \min\{p, 1-p\}$ is a penalty term corresponding to the case where a customer clicks on a product that a type is not interested in. The type $z(t)$ that a customer t is assigned to is the solution to the following expression that captures the likelihood of belonging to each type given the observed click vector $\boldsymbol{c}_t$:

$$z(t) = \arg\max_{z \in \mathcal{Z}} \prod_{i=1}^{r_t} s_{it}^{(z)}. \tag{48}$$

5. *Joint, iterated Bayesian inference of first impression windows and type probabilities*: We use an iterative approach that starts with an initial assumption (uniform distribution) on the first impression windows and type probabilities and jointly and iteratively updates both of these parameters until the process converges. Specifically, suppose that $(\hat{\mathcal{D}}_k^{(z)}, \hat{q}^{(z)})_{z \in \mathcal{Z}}$ are the current estimates for the corresponding distributions in a given iteration. We use the following two steps to update these parameters:

- For every $t \in \tilde{\mathcal{T}}_c$, we identify $z(t)$ as per (48). For the customers who did not click on any product but have non-zero preferences ($t \in \tilde{\mathcal{T}} \setminus \tilde{\mathcal{T}}_c$) we randomly assign them to a type $z(t) \in \mathcal{Z}$ using Bayes rule, which in turn is dependent on $(\hat{\mathcal{D}}_k^{(z)}, \hat{q}^{(z)})$:

$$Pr(z(t) = z | \boldsymbol{c}_t = 0) = \frac{\hat{q}^{(z)}}{\frac{|\tilde{\mathcal{T}} \setminus \tilde{\mathcal{T}}_c|}{\tilde{\mathcal{T}}}} \sum_{k \in [n]} Pr(\boldsymbol{c}_t = 0 | \boldsymbol{p}^{(z)}, k_t = k) Pr(k_t = k | \hat{\mathcal{D}}_k^{(z)}).$$

Once every customer t is assigned to a type $z(t)$, we can update $\hat{q}^{(z)}$ to be the fraction of customers assigned to type z.

- Given $z \in \mathcal{Z}$, for every customer $t \in \tilde{\mathcal{T}}$ such that $z(t) = z$, we compute the probability that the customer's first impression window is k given her type z. That is for all $k \geq r_t$, we can use the (conditional) distribution $\hat{\mathcal{D}}_k^{(z)}$ to infer the probability that her first impression window is $k_t = k$ subject to $k \geq r_t$, where r_t is the rank of the last product that the customer clicked on ($r_t = 0$ if $t \in \tilde{\mathcal{T}} \setminus \tilde{\mathcal{T}}_c$).

$$Pr(k_t = k | z(t) = z, k \geq r_t) = \frac{Pr(k_t = k | \hat{\mathcal{D}}_k^{(z)})}{\sum_{r=r_t}^{n} Pr(k_t = r | \hat{\mathcal{D}}_k^{(z)})}.$$

Finally, we can update the distribution $\hat{\mathcal{D}}_k^{(z)}$ by aggregating the probability that $Pr(k_t = k | z(t) = z)$ over all $k \in [n]$ and all t such that $z(t) = z$.

We terminate the iterative process when the root mean square errors of the values of both $\hat{\mathcal{D}}_k^{(z)}$ and $\hat{q}^{(z)}$ between successive rounds are smaller than a specified tolerance level

Parameter Choices First, we assume that 70% of the customers have non-zero preference vectors, i.e., $\frac{|\tilde{\mathcal{T}}|}{|\mathcal{T}|} = 0.7$. This reflects a conservative modeling choice that is at the lower end of the range of this parameter that we analyzed in Appendix C. By assuming that 30% of the customers cannot be hooked by any ranking, we seek to compensate for our other, less conservative assumptions. Second, for the rest of this section, we fix the number of customer types or clusters to be $|\mathcal{Z}| = 75$; we tested and found similar results for other values such as $|\mathcal{Z}| = \{25, 50, 100\}$ but present only $|\mathcal{Z}| = 75$ for brevity. This is in stark contrast to the experiments in Section 5 with thousands of types and only a handful of customers per type. Finally, we assume that the customer click probability $p \in [0.5, 0.75]$, which corresponds to an average click probability over all products of $\frac{2p}{48} \approx [0.02, 0.03]$ per Wayfair's recommendation.

E.2. Results

The implementation of the Threshold Acceptance Algorithm is identical to our earlier simulations and we refer the reader to Section 5.2.2 for a detailed description. As in Section 5, we conducted the following simulations 100 times for each event. First, we generated $|\tilde{\mathcal{T}}|$ customers from the inferred distribution $\hat{\mathcal{D}} = (\boldsymbol{p}^{(z)}, \hat{\mathcal{D}}_k^{(z)}, \hat{q}^{(z)})_{z \in \mathcal{Z}}$. We applied our Threshold Acceptance Algorithm for each customer profile that was generated assuming that customers arrive in a random order and then compared the algorithm's performance to that of Wayfair's static ranking (π^{WF}) and the offline benchmark static ranking. Owing to a judicious choice of the parameters mentioned earlier, the click behavior of customers on Wayfair's ranking was a close fit to the true click distribution observed in the data. For example, in all of our simulations, the percent of customers hooked by Wayfair's ranking was within $[0.98, 1.06]$ times the actual percentage of customers hooked.

Figure 11 compares the performance of our algorithm and π^{WF} to the static ranking output by the Greedy Algorithm for OffAR (π^g): specifically, (total number of hooked customers by our algorithm (or π^{WF})) / (total number of hooked customers by π^g). We see that our Threshold Acceptance Algorithm yields a significant improvement over Wayfair's static ranking. Averaged over 100 simulations, the number of customers hooked by our algorithm was at least a multiplicative factor of 1.5-times the number hooked by Wayfair's ranking for all six events. There are two main reasons why we see such a significant improvement over Wayfair's ranking. First, unlike in Section 5, here we allow product interests to be correlated with first impression windows. Our Threshold Acceptance Algorithm is able to exploit these dependencies better than Wayfair's static,

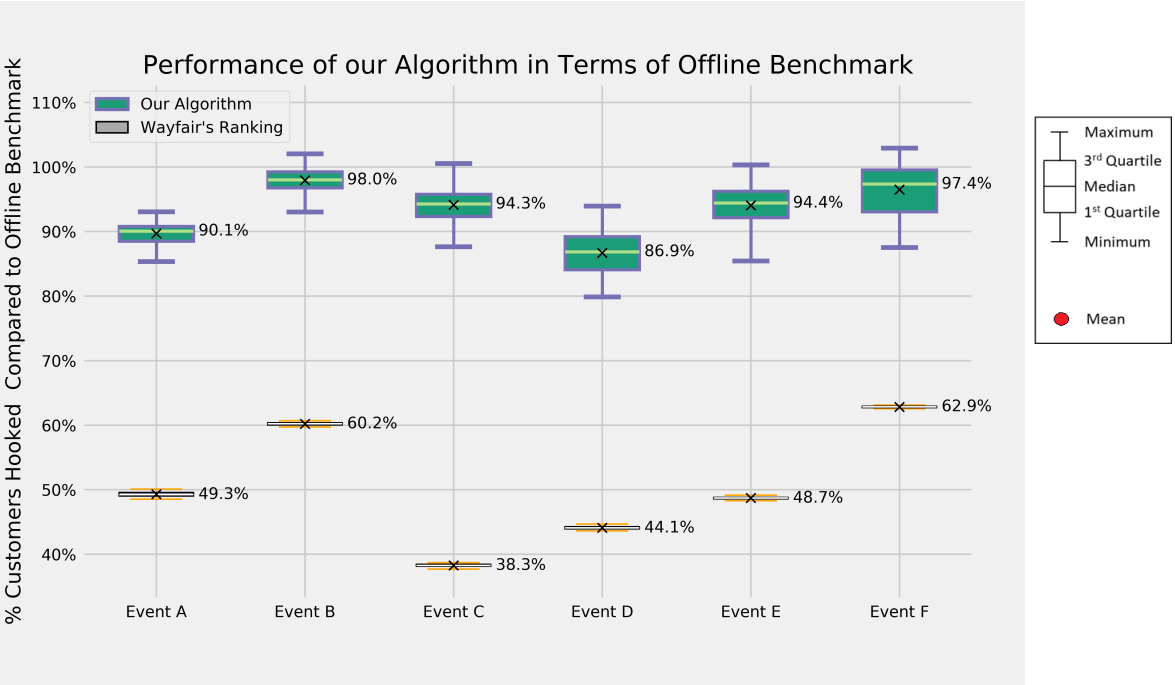

Figure 11 **Box & whisker plots presenting the percent of total customers hooked by π^g (the offline benchmark) that are hooked by our algorithm and π^{WF}**

popularity-based ranking. Second, our assumptions on browsing and clicking behavior for these simulations are less conservative than those in Section 5, and provide more opportunities for our algorithm to hook those customers who were not hooked by Wayfair's ranking. It is likely that the actual improvement in practice lies somewhere between these results and the more conservative ones in Section 5.

Another key finding is that our Threshold Acceptance Algorithm's performance is comparable to the offline benchmark: On average over the six events, the number of customers hooked by our algorithm is at least 87% of the amount hooked by π^g. We note that the variability in the performance of our Threshold Acceptance Algorithm is primarily due to the randomness that comes from sampling only $L = 500$ customers to estimate the marginal benefit of each product. A surprising by-product of this variability is that in a few instances, our algorithm actually outperforms the benchmark ranking, by up to 3%. Indeed, it is worth noting that since the benchmark ranking π^g is computed on the full distribution $\hat{\mathcal{D}}$, its performance can vary considerably on individual populations that are randomly sampled from $\hat{\mathcal{D}}$. Beyond randomness, this can also occur due to the fact that the Greedy Algorithm for OffAR is only a $\frac{1}{2}$-approximation to the true optimal ranking.[25]

[25] Recall that the true optimal ranking is NP-Hard to compute even for a special case of our problem

As a result, our algorithm may converge to a ranking better than π^g by making some locally suboptimal decisions.