Learning to Diversify for E-commerce Search with Multi-Armed Bandit

Anjan Goswami
agoswami@ucdavis.edu
University of California, Davis

Chengxiang Zhai
czhai@illinois.edu
University of Illinois, Urbana-Champaign

Prasant Mohapatra
pmohapatra@ucdavis.edu
University of California, Davis

ABSTRACT

Search is central to e-commerce platforms. Diversification of search results is essential to cater to the diverse preferences of the customers. One of the primary metrics of e-commerce businesses is revenue. On the other hand, the prices of the products shown influence customer preferences. Hence, diversifying e-commerce search results requires learning the diverse price preferences of the customers and simultaneously maximizing the revenue without hurting the relevance of the results. In this paper, we introduce the learning to diversify problem for e-commerce search. We also show that diversification improves the median customer lifetime value (CLV), which is a critical long-term business metric for an e-commerce business. We design three algorithms for the task. The first two algorithms are modifications of algorithms that are in the past developed in the context of the diversification problem in web search. The third algorithm is a novel approximate knapsack based semi-bandit algorithm. We derive the regret and pay-off bounds of all these algorithms and conduct experiments with synthetic data and simulation to validate and compare the algorithms. We compute revenue, median CLV, and purchase based mean reciprocal rank (PMRR) under various scenarios such as with changing user preferences with time in our simulation to compare the performances of these algorithms. We show that our proposed third algorithm is more practical and efficient compared to the first two algorithms and can produce higher revenue, maintain a better median CLV and PMRR.

ACM Reference Format:
Anjan Goswami, Chengxiang Zhai, and Prasant Mohapatra. 2019. Learning to Diversify for E-commerce Search with Multi-Armed Bandit. In *Proceedings of SIGIR (ECOM'19).* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Search has become a central functionality of e-commerce sites. Typically, large e-commerce platforms have several competing products relevant to a query. For example, Amazon returns more than 1000 products for the query "car seat", about 300 products for the query "smart watch", and all the products up to almost the last page of results are relevant to the query. Many products shown in first few pages have great reviews and it is clear that these products have been regularly purchased many times from Amazon. This scenario illustrates that there are often too many choices for users on a large e-commerce web site. However, it has been well researched that that user clicks drop dramatically after the first page [30, 39] in web and e-commerce search. Consequently, a user often does not browse through all the relevant products returned by the search query and can abandon the search if there are only a few or no relevant products found in top results. This search abandonment is known to get reduced by showing the diverse result set to the users in web search [15]. Search abandonment hurts e-commerce platforms even more because the business model depends on actual purchases instead of an ad-clicks. Hence, the diversification of ranking is a critical problem for e-commerce sites. One of the primary business metrics for an e-commerce business is revenue [17] generated from the sales. The attempt for diversification can hurt the revenue unless we explicitly formulate the diversification problem that maximizes the revenue. Moreover, e-commerce businesses use customer lifetime value or CLV [20, 23] as a critical long term metric. CLV is the revenue generated by customers in their lifetime with the site. A successful e-commerce site intends to increase the pool of high CLV customers. Intuitively, by selecting proper diversification of results, an e-commerce site can cater to the preferences of a larger pool of customers and can improve the median CLV of the business. The aspects of ensuring maximization of revenue and not hurting the relevance simultaneously while learning to diversify for e-commerce search require a unique formulation. In this paper, we address this problem of diversification of e-commerce search results. Additionally, we show that our formulation also improves the median CLV. We have made three contributions in this paper:

(1) We define the learning to diversify problem for e-commerce search considering maximization of revenue and keeping the loss in relevance within a bound. Additionally, we show that such design of learning to diversify problem also improves median CLV for an e-commerce business.

(2) We present three multi-armed bandit based algorithms for this and derive the regret and pay-off bounds for them. Our third algorithm is a novel approximate knapsack based semi-bandit optimization algorithm and we show that the algorithm performs well for our problem.

(3) We also present a simulation-based evaluation strategy and conduct experiments with synthetic data to show that under most of the scenarios such as changing customer preferences and under the assumption of position bias our semi-bandit algorithm can maintain a right balance of revenue, median CLV, and mean reciprocal rank [16] based on purchases.

ECOM'19, July 2019, Paris, France

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 BACKGROUND

Diversification of search result is one of the options for managing the uncertainties and ambiguities in understanding the user's information need from search queries [13]. A search ranking function, optimized for relevance, is not designed to minimize the possibilities of redundancies on the search results [5]. The diversification problem has been discussed as one of the most important future research directions in learning to rank [12]. Researchers address this problem by using variants of two broad approaches: (1) The first approach defines a measure of similarity based on the contents of the documents and designs a ranking function that can use this measure to generate a diversified search result while not hurting the relevance as much as possible. (2) The second approach uses a multi-armed bandit based online learning algorithm to learn the diverse user preferences and optimize the ranking based on that.

The second approach does not require defining any similarity measures, and instead, it learns from the data. Hence, it is easier to realize in practice. In this paper, our algorithms are based on the second approach because it is much harder to map any product similarity measures to user preferences than to learn it from the data. Moreover, user preferences can change over time, and a machine learning based approach can better adapt to the changes.

One of the most influential papers on the first approach is by Carbonell et al. [11] where the authors introduce the concept of maximal marginal relevance to maximize selecting documents that are different from the already chosen documents while reducing the loss in overall relevance.

In another paper, Zhai [44] provides an algorithm for optimizing search results by diversification using risk minimization principles that use correlation among search result as a similarity measure. Agrawal et al. [2] propose a greedy algorithm that reorders the top k search results to jointly maximize the probability of showing diverse documents for a query and minimize the exposure of low-quality documents to the users. The authors also provide a generalization of the classic ranking metrics such as discounted cumulative gain (DCG), mean average precision (MAP), etc. to account for diversification in the ranking. Some researchers [13] pose this problem slightly differently for reducing ambiguity in search queries. Santos et al. [37] provide another similar algorithm that uses additional information by either reformulating queries or using queries from related search to diversify the search results. There are also several papers written on diversification of results of recommender systems using a variant of content based similarity measures [42].

The most important paper on the second approach is probably the one written by Radlinski et al. [36] where authors come up with an online algorithm based on classic multi-armed bandit (MAB) paradigm that can learn the user preferences. The authors also provide a baseline greedy algorithm to compare with their proposed multi-armed bandit based algorithm. Two of our algorithms are direct modifications of the algorithms presented in that paper. The MAB algorithm in that paper requires one MAB per rank position, and each MAB can have as many arms as the number of items. This strategy increases the requirements of , and also there are overlapping arms for the MABs in various positions which are not optimal since one needs to discard the already selected arms for the MABs. Several papers that use MAB framework or online learning to diversify, appear in the domain of news content optimization problems [1] where diversity is considered to be the essential factor. One of the interesting papers by Yue et al [43] uses linear submodular bandits based paradigm to learn the user preferences for diverse ranking. The use of sub-modular bandits guarantees the existence of a greedy approximation algorithm. In some other papers, researchers use a linear or nonlinear model to simultaneously learning to rank and diversify [14, 18, 28]. The main problem with such algorithms is that the computation of variance that is used to update the rewards in MAB framework becomes more complex. It is also generally much harder to evaluate the effectiveness of an online algorithm compared to traditional batch learning models. Hence, using a ranking function for both the learning to rank and diversify may not be practical for realization. Although, it is possible to use an online algorithm for diversification on top of traditional learning to rank (LTR) algorithms to reduce the complexity of the engineering system. In e-commerce, the challenge is the need to account for revenue maximization, which requires formulating a different learning problem.

We show the improvement in median CLV with the diversification of results in e-commerce. We have not found this relationship in any other papers. However, extensive literature is available on CLV [9, 29] for the interested readers. We omit to provide any survey of CLV modeling literature since our work is not related to any of those.

3 PROBLEM SETTING

We address the problem of diversifying the e-commerce search results based on the perspective of the e-commerce firm and also from the perspective of the users. The firm intends to maximize the revenue, increase the CLV. The customers desire to find relevant products that they may be interested in purchasing. Hence, we define the learning to diversify problem for e-commerce along with maximization of revenue and maintaining a relevance threshold. We now create a few notations to explain the problem. Let's consider a query q and a corresponding set of n relevant items $\mathcal{D}_i = \{d_1, d_2, \cdots, d_n\}$ whose prices are given by $\{\rho_1, \rho_2, \cdots, \rho_n\}$. We also assume that each product has a relevance score with respect to the query and these scores are given by $\{s_1, s_2, \cdots, s_n\}$. The relevance score is correlated to user clicks but the degree of correlation can vary for various queries and the product corpora. Let us denote a set of m users by $U = \{u_1, \cdots, u_m$. We also consider a simple user behavior model where we assume that a user browses the items one after another from the top and then either purchase an item and leaves the site or leaves the site without any purchase after browsing at most the top k ($k << n$) items that are shown to him or her. We also assume that we study iterations (search sessions) $\{1, \cdots, T\}$ and the top k items at an iteration t for a query q can be given by $(b_1^t, \cdots, b_k^t)$. The corresponding products can be given by $(d(b_1^t), \cdots, d(b_n^t))$. Note that, in our scenario, we do not have specific user information, and we only have the query. In an e-commerce site, this is a general scenario when customers start browsing for a product, and typically they log into the site for purchase or sometimes can use a guest account for a purchase. Our goal is to select the best top k items from the n items for the

query q such that the result set caters to diverse preferences of the users. If the user finds a relevant product in the top k, then he or she can decide to purchase or not purchase that product. Let us use two indicator variables, $z_{jt} = \{0, 1\}$ and $x_{jt} = \{0, 1\}$. The first one denotes if the product d_j is selected in top k results and is shown to a customer and the second one is to denote whether or not the product d_j is purchased in iteration t for the given query. The revenue generated from the product d_j for the query in an iteration t can be computed as $r_t = \sum_{j=0}^{j=n} x_{jt}\rho_j$. The total revenue can then be given by $R_T = \sum_{t=1}^{t=T} r_t$. We can also similarly define an user level revenue expression R_{u_i} to denote the total revenue obtained by an user in T iterations. Then, we also have $R_T = \sum_{i=1}^{i=m} R_{u_i}$. We also define a cumulative sum of relevance scores of top k products to keep a bound on relevance and denote it by $\mathcal{S} = \{s_1, \cdots, s_k\}$. Given the above set up learning to diversify problem in e-commerce can be mapped to maximizing $\forall i R_{u_i}$. However, this is not the same as maximizing R_T. It is also hard to estimate R_{u_i} without estimating the purchase probabilities of a specific user. Learning to diversify algorithm, on the other hand, can optimize the search result more for a larger pool of users and improve the median of the CLV. However, the optimization problem then requires to maximize the revenue while aiming to learn the diverse preferences of the customers.

Hence one possible solution is simultaneously learning to diversify and maximize the total revenue while maintaining a reasonable value for relevance. This approach also requires us to use a threshold for the relevance of the top k products.

Note that using such a relevance threshold is not really a new concept and has been used before in literature on diverse ranking [11] and in general for limiting recall set in search [22]. It is not hard in practice to establish such thresholds for a query and is often used in industry [41] for various purposes.

4 LEARNING ALGORITHMS

We now describe three algorithms for the problem. The first two algorithms are modifications of algorithms proposed in Radlinski et al. [36] paper. The third algorithm is a new knapsack based semi-bandit algorithm that we develop for this problem. We use the first two algorithms as baselines for learning to diversify problem for e-commerce search. The third problem is a novel algorithm that we are introducing in this paper.

4.1 Revenue Ranked Explore and Commit algorithm (RREC)

This algorithm is similar to the "ranked explore and commit algorithm" (REC) described in the paper by Radlinski et al. [36]. We intend to maximize the revenue, which is real-valued instead of click-through rate, which is a binary input. It requires a minor change. The algorithm iteratively shows each of the n items at each rank x times. It then records the purchases for these nx iterations for every product at every rank position. We can then estimate the probability of purchase of every product at every rank. We then require to multiply the estimated probability of purchase by the price of the product to determine the generated revenue. The revenue is real-valued and can be an arbitrary number. Hence, we need to normalize the price between 0 and 1 for all the products to estimate a normalized value of the expected revenue. After we complete nx iterations, the algorithm then presents the products in the order of decreasing the expected revenue. Our implementation of the algorithm is shown in 1.

This first greedy algorithm maximizes the revenue generated after nx iterations if the user preferences are unchanged. The main problem of this algorithm is that It assumes that the preferences of the users' do not change with time. The algorithm may achieve excellent performance from the revenue metic's perspective in specific scenarios when the customer preferences do not change particularly after the nx iterations once it has a reasonable estimation of the purchase probabilities. However, since there is a need for showing every product at every position, the regret of this algorithm can be very poor, and consequently, the revenue generated in initial nx iteration can be arbitrarily bad. Hence, this algorithm is quite impractical for actual implementation.

Moreover, in e-commerce, it is particularly unwise to take any risk of making customers unhappy. Even the controlled experiments require to be conducted very carefully often with a budget [21]. We present this algorithm to provide a comparison with a simple greedy algorithm that always maximizes the revenue after a sufficiently long enough iteration as a baseline.

Algorithm 1 Revenue Ranked Explore and Commit algorithm (RREC)

input: Items $(d_1, d_2, \cdots, d_n)$, parameters: ϵ, δ, k.
$x \leftarrow \lceil 2k^2/\epsilon^2 \log(2k/\delta) \rceil$
$(b_1, b_2, \cdots, b_k) \leftarrow k$ arbitrary items.
for $i = 1, \cdots k$ **do** ▷ at each rank $\forall j i_j = 0, p_j = 0, r_j = 0$
  for $c = 1, \cdots, x$ **do** ▷ Loop x times
    for do$j = 1, \cdots, n$ ▷ over every item d_j
      $b_c \leftarrow d_j$
      display $b_1, \cdots, b_k$ to the user
      $i_j = i_j + 1$
      if user purchases on b_c **then**
        $p_j = p_j + 1$
      end if
    end for
  end for
  for do$j = 1, \cdots, n$
    $pr_j = p_j/(i_j + \beta)$ ▷ $\beta \geq 1$ is a constant to avoid division by zero
    $mr_j = pr \times \rho_j \times Z$ ▷ Z is a normalization constant for the prices for a query
  end for$j^* \leftarrow \text{argmax}_j\, mr_j$ ▷ Commit to best document at this rank
$b_i \leftarrow d_{j*}$

end for

4.2 Revenue Ranked Bandits Algorithm (RRBA)

In this algorithm, we modify Radlinski et al. [36]'s "ranked bandit algorithm" (RBA) for learning to diversify and maximize the revenue coming from the users. We provide a schematic of the modified algorithm in 2. It uses k bandits $MAB_1, \cdots, MAB_k$ for k

rank positions for a query. Each bandit is assumed to have n arms $(a_1, \cdots, a_n)$ corresponding to the n products in the recall set for the query. If an user purchases a product d_j from rank position r at an iteration t, we update the score for the arm a_j of MAB_r as follows:

$$v_{rj} = p_{rj}/i_{rj} \times \rho_j \times Z + \alpha\sqrt{2\ln t/i_{rj}}$$

where p_{rj} is the purchase count and the i_{rj} is the impression count of the product d_j at rank position r at that iteration, α is a constant, and Z is a normalization factor for the price of the products. The algorithm considers the same set of products as arms for all the MABs at all positions. Consequently, once a product is selected by the $k-1$ MAB, the same product cannot be selected by the k-th MAB. Hence, except the MAB at the very first position, all other MABs may not select their best arms. This phenomenon makes the algorithm performing poorly for choosing the optimal top k products. Moreover, the authors mention in the paper that the analysis of the regret for the non-binary case is non-trivial and the greedy algorithm on which RBA is based can obtain a pay-off bound that is a factor of $(k-\epsilon)$ below optimal for any ϵ. This algorithm can learn the preferences even if those are changing and does not require to estimate the purchase probability of every product as similar to the RREC. However, another problem with this algorithm is that the number of MABs and the amount of bookkeeping required to run this in practice. Typically, any such MAB algorithms in practice cannot replace the learning to rank algorithms, as mentioned in the paper by Radlinski et al. [36]. The most practical way of implementing any bandit algorithms or optimizations can be to use it as the topmost layer of a multi-layer ranking architecture where the set of products ranked by the LTR algorithm in a layer below can be handed over to the MAB algorithm.

4.3 Knapsack based bandit algorithm (KPBA)

KPBA is a novel algorithm that we propose in this paper. To overcome the problem of overlapping arms in k MABs for RRBA algorithm, we here consider a semi-bandit algorithm [6] that uses a single bandit with n arms and selects k best arms at every iteration.

It processes feedback for k arms at each iteration. Furthermore, to guarantee better relevance, we introduce a relevance threshold while maximizing the revenue and learning the diverse preferences. This optimization problem turns out to be similar to the well known exact k-item Knapsack problem or E-kKP [25]. Note that the revenue expression is based on the UCB score similar to our formulation of RRBA. This algorithm does not require as much bookkeeping as the RRBA since it does not need to maintain k MABs. It also does not have the limitation of not being able to select an optimal arm. Furthermore, It keeps a relevance bound, and hence, it can have a better performance for relevance based on any information retrieval based ranking metrics such as mean reciprocal rank (MRR). It is also a bandit paradigm, thus it can learn changing preferences of customers.

We now discuss the KPBA formulation below:

Suppose, an item d_j is purchased at an iteration t in this algorithm, then the normalized revenue generated in that iteration can be written as $r_{jt} = \rho_j \times Z$. The algorithm has to keep track of n UCB scores for the n products. The UCB score for a product d_j after iteration t can be written as $v_{jt}^{UCB} = r_{jt} + \alpha\sqrt{2\ln t/i_{jt}}$. Note, i_{jt} is the impression count of item d_j at iteration t. At each iteration $\{1, 2, \cdots, T\}$, each product for the top k position can then be chosen from a knapsack based optimization framework as follows:

$$\max_{1,2,\cdots,T} \quad \sum_{j=1}^{k} v_{jT}^{UCB}$$
$$\text{subject to} \quad \sum_{j=1}^{k} s_j \geq B \tag{1}$$

Here B is a threshold for the cumulative sum of relevance scores. We update the v_j^{UCB} after each iteration. In order to solve this problem, we define the problem 1 as a binary integer programming (BIP) problem [19]. We define $\hat{s_i} = 1 - s_i$ and $\hat{B} = C - B$, where C is another constant. Problem 1 can then be rewritten as:

$$\max_{x_{1t},\cdots,x_{nt}} \quad \sum_{i=1}^{n} x_{it} \times v_{it}^{UCB}$$
$$\text{subject to} \quad \sum_{i=1}^{n} x_{ij} \times \hat{s_i} \leq \hat{B}$$
$$\sum_{i=1}^{n} x_{ij} = k \tag{2}$$

Problem 2 is known as E-kKP. This problem is NP-hard in terms of the number of arms n. A brute force solution for this problem can be found in $O(n^k)$ time where k is the number of selected arms.

Algorithm 2 Revenue Ranked Bandits Algorithm (RRBA)

Initialize: $MAB_1(n), MAB_2(n), \cdots, MAB_k(n)$ ▷ Initialize the MABs

for $t = 1, \cdots, T$ **do** ▷ for T iterations
  for $r = 1, \cdots, k$ **do** ▷ for every position r
$\hat{b}_r \leftarrow$ selectarm$(MAB)_r$
    if $\hat{b}_r \in (b_1^t, \cdots, b_{r-1}^t)$ **then** ▷ replace repeats
$b_r^t \leftarrow$ Arbitrary document from D
    else
$b_r^t \leftarrow \hat{b}_r$
    end if
  end for
  display $(b_1^t, \cdots, b_k^t)$ to users; record purchases.
  for $r = 1, \cdots, k$ **do** ▷ Do all updates
    $i_{d(b_r^t)} = i_{d(b_r^t)} + 1$ ▷ Assume that the products
    if user purchases b_r^t, and $\hat{b}_r \leftarrow b_r^t$ **then**
      $p_{d(b_r^t)} = p_{d(b_r^t)} + 1$
    end if
    $pr_{d(b_r^t)} = p_{d(b_r^t)}/i_{d(b_r^t)}$
    $mr_{d(b_r^t)} = pr_{d(b_r^t)} \times \rho_j \times Z$
    $var_{d(b_r^t)} = \alpha\sqrt{2\ln t/i_{d(b_r^t)}}$
    $sc_{d(b_r^t)} = mr_{d(b_r^t)} + var_{d(b_r^t)}$
    Update MAB_r, arm $= b_r^t$, reward $= sc_{d(b_r^t)}$
  end for
end for

However, there is a $\frac{1}{2}$-approximation algorithm named $\mathsf{H}^{\frac{1}{2}}$ which has been proposed in a paper by Caprara et al. [10]. The authors use a well-known LP relaxation of Knapsack problem [32] and use the fact that any basic feasible solution of a linear relaxation of BIP contains at most two fractional weights. In case, there are no fractional weights, then the optimal basic feasible solution of LP relaxation problem is an optimal solution for the BIP problem. However, if there are one or two weights out of all n weights have fractional values for the basic feasible solution, then we can select one out of the two fractional, and it can still guarantee a $\frac{1}{2}$-approximate solution for E-kKP.

The authors provide an analysis of the algorithm and have shown that it runs in $O(n)$ time. Readers can get the details of the algorithm $\mathsf{H}^{\frac{1}{2}}$, and it's analysis in the paper by Caprara et al. [10].

We have shown a schematic of our proposed algorithm in refkpba where we use the $\mathsf{H}^{\frac{1}{2}}$ algorithm as a subroutine.

Algorithm 3 Knapsack based bandit algorithm (KPBA)

Initialize: $SemiMAB(n)$ ▷ Initialize the MABs
for $t = 1, \cdots, T$ **do** ▷ for T iterations
  $\{b_1, \cdots, b_k\} \leftarrow \mathsf{H}^{\frac{1}{2}}(\mathcal{D})$ ▷ The details of this algorithm is in the paper [10]
  display $(b_1^t, \cdots, b_k^t)$ to users; record purchases.
  for $l = 1, \cdots, k$ **do** ▷ Update the impression counts of products, corresponding to b_1^t to b_k^t
    $i_l = i_l + 1$ ▷ $l = 1, \cdots, k$ are now indices of k products
    $I_l = 0$ ▷ I_l is a boolean indicator variable.
    if user purchases b_l, and $\hat{b_r} \leftarrow b_r$ **then** ▷ Update UCB scores $I_l = 1$
    end if
    for $l = 1, \cdots, k$ **do**
      $v_l = I_l \times p_l / i_l \times \rho_l \times Z + \alpha\sqrt{2 \ln t / i_l}$
    end for
  end for
end for

5 THEORETICAL ANALYSIS

5.1 The offline optimization problem

It is straightforward to see that the problem of finding set of k optimal products from n products with binary reward is equivalent to the maximum coverage problem [36]. The optimal greedy approximation solution [27, 35] for this problem is a $(1 - \frac{1}{e})$-approximation algorithm.

5.2 RREC

We modify Radlinski etal.'s [36] REC algorithm to work with real-valued rewards. This algorithm serves as a baseline in our paper. However, Radlinski et al.'s [36]' paper mentions that the regret for REC can be extended to the case when the rewards are not binary. They show that REC can achieve a payoff $(1 - \frac{1}{e} - \epsilon)OPT - O(k^3 \times n/\epsilon \ln(k/\delta))$ with at least probability $(1 - \delta)$. In our case, the expression will be similar since we use the same algorithm with real reward.

5.3 RRBA

Radlinski et al. [36] have provided the results on regret and payoff for RBA algorithm using EXP3 [8] and binary rewards. The combined payoff is shown in the paper as $(1 - \frac{1}{e})OPT - O(k\sqrt{nT \log n}$. Authors commented that the case of real reward can be $(k - \epsilon)$ worst below optimal, for any $\epsilon \geq 0$. We can use the same combined payoff for RRBA using the real reward.

5.4 KPBA

Our problem is conceptually similar to dynamic assortment selection problem [38]. However, assortment selection models often assume that the purchase of the product depends on the selection of a specific set of products. Many papers in this area use the concept of the utility of a set of products and use a choice model for modeling the purchase behavior of the users [33]. However, we assume that users explore each product individually and independently, and users are not interested in going beyond the top k products if they do not find the desired product. The analysis of our algorithm can be similar to the analysis of the algorithms developed for dynamic assortment selection problems. Specifically, we are going to use the proof for regret bound from one such paper by Agrawal et al. [3]. The authors use a choice model for purchase and keep showing one assortment of k products for a particular time until a no purchase event happens. In our case, the loss in revenue at each iteration t is expressed as the following:

$$((1 - \frac{1}{e})OPT - \frac{1}{2}\sum_{j=1}^{j=n} z_{jt} v_{jt}^{UCB})$$

The second term is coming from the half approximation exact k knapsack algorithm that we have used. Intuitively, since v_{jt}^{UCB} follows the properties of UCB algorithm, hence each product here is akin to be bounded by the regret bound given by the UCB algorithm [7]. Note that, the number of times an item shown to the customers on an average is bounded by $\frac{tk}{n}$ for t iterations. Let's also use $\hat{r_{jt}}$ to denote the expected normalized revenue of an item d_j in an iteration t. Now, we can write the following:

$$\sum_{j=1}^{j=n} z_{jt} v_{jt}^{UCB} \geq \sum_{j=1}^{j=n} z_{jt} \hat{r_{jt}}$$

$$\sum_{j=1}^{j=n} (z_{jt} v_{jt}^{UCB} - z_{jt} \hat{r_{jt}}) \leq \sum_{j=1}^{j=n} \alpha\sqrt{(2 \ln t / i_{jt})}$$

$$\sum_{j=1}^{j=n} (z_{jt} v_{jt}^{UCB} - z_{jt} \hat{r_{jt}}) \leq O(\sqrt{(n \ln t/(tk))})$$

$$\sum_{j=1}^{j=n} (z_{jt} v_{jt}^{UCB} - z_{jt} \hat{r_{jt}}) \leq O(\sqrt{(nt \ln t)})$$

Note that, we use the following expected average:

$$\sum_{t=1}^{t=T} \sqrt{\frac{1}{t}} \leq \sqrt{T}$$

We can also use a different bound from UCB algorithm using $\lg n$) instead of $\lg t$).

Our sketch of derivation is similar to the proofs in Agrawal et al.'s paper [3]. The bound can probably be made tighter using techniques used by Auer et al. [7] for proving the bounds of UCB algorithm. This shows that the regret here can be bounded by $O(\sqrt{(nT \lg T)})$. The payoff then can be also bounded by $((1-\frac{1}{e})OPT - O(\sqrt{(nT \lg n)})$ for KPBA, which is similar to Radlinski et al. [36]'s algorithm for real reward.

5.5 Comments from Analysis

From the above analysis, it is clear that RREC has several drawbacks as also identified by Radlinski et al. [36] in their paper. Moreover, RREC is impractical since it's regret in first nx iterations can be the worst [36]. We then expect this to also perform poorly in terms of revenue as well as median CLV. The analysis does not tell us anything about its performance in terms of relevance. The RRBA on the other hand, clearly can be concluded to perform better than RREC for revenue based on the analytical expression of regret. KPBA can be expected to be similar in terms of revenue and median CLV. However, KPBA can be more appealing compared to RRBA when we have real rewards. Moreover, since KPBA does not compromise relevance beyond a bound for achieving the diversification, hence, it can be expected to perform better for the users in terms of relevance metrics.

6 EVALUATION

In this study, we use the following metrics to evaluate the performance of our learning algorithms from the perspective of revenue, CLV, and relevance:

- **Average *Revenue* per Query: ARQ** This is the average revenue for all the queries in the experimental study and can be obtained by the expression: $\frac{\sum_{i=1}^{i=t} R_i}{n}$ where t is the total number of iterations in the simulation study and n is the number of queries.
- **Median *Customer Life Time Value*: MCV** Suppose each customer u_j spends $r_{u_j}^t$ for purchasing products in in t iterations. Then the customer value can be represented by the median of all customer spends.
- ***Mean Reciprocal Rank* of Purchases: PMRR** Reciprocal rank based on purchases is the reciprocal of the rank of the product that has been purchased for a given query in a search session. The mean reciprocal rank of purchases is the mean of the reciprocal ranks for all search sessions in a period of time.

It is clear from the analytical derivations in our section 3 that KPBA and RRBA both perform better compared to RREC in terms of ARQ. This also indicates that these two algorithms can exceed RREC in MCV. Moreover, because of the bounds in relevance for KPBA, it is expected that PMRR can be better than RRBA. However, it is not clear how the PMRR for KPBA can compare with RREC for which PMRR can be good particularly after nx iterations. It is also unclear the difference between KPBA and RRBA in terms of ARQ since the pay-off expressions are similar.

We did not have any historical search log data from a real e-commerce site. We thus evaluate our algorithms by generating some synthetic data and conducting a simulation study. We assume that different users have different price preferences and that mainly dictates their purchase behavior. In reality, user preference is a complex function of multiple factors associated with the products and other variables such as time in a year, the financial status of the person, etc. However, this assumption helps us keep our data generation and simulation simple but allows us to evaluate and compare our algorithms in a manner that reveals the characteristics of the proposed algorithms. In our simulation study, we model the biases of a real system. Typically, it is known that evaluation of bandit based algorithms in an online setting can be very hard [31] and offline counterfactual techniques [24] using historical logs have been recently a topic of research for such evaluations. We have taken the main ideas of using such evaluation techniques in this paper.

We aim to answer the following questions from the simulation study:

(1) How does KPBA compare with RRBA in terms of ARQ, MCV, and PMRR? In particular, we are interested in how the much better value of these metrics can be obtained if we use KPBA instead of using RRBA under scenarios such as without and with position bias and with changing customer preferences.

(2) We also intend to see if RREC performs better in PMRR compared to RRBA and how does that compare with KPBA with a reasonable assumption on relevance threshold.

We discuss the experiments after illustrating our data generation methodology.

6.1 Synthetic data generation

Our synthetic e-commerce data consists of N queries and M relevant products per query. We assign prices to the products randomly from a multimodal Gaussian distribution with $m = \{1, 8\}$ peaks with mean prices between \$10 to \$500. The purchase rates are generated from a similar distribution using mean peaks between 0.0 to 0.06. The maximum mean peak purchase rate is assigned to the cheapest mean price peak for 70% of the time and rest of the time that is assigned to any other mean price peaks. We additionally generate a relevance score that is linearly correlated to the purchase rates with a person correlation coefficient between 0.10 and 0.30 with a p-value less than 0.10. In this paper, we use $M = 200$. Note that in a real scenario, M can be a very large number, but typically a recall set for a search query can drop to a much smaller size because of the performance reasons and we anticipate to apply our algorithms on top of a multi-layer ranking architecture. Moreover, this makes running our experiments simpler and faster without losing generality. The figure 2 provides histograms of synthetically generated prices and also shows the relationship between synthetically generated relevance score and the product purchase rates for that query. The multimodal distribution of price and a weak linear correlation between relevance score and the purchase rate represent a common scenario for an e-commerce platform.

6.2 User's price preference model

We assign every user $u \in U$ to a preferred price cluster t_u using a Chinese Restaurant Process (CRP) [4] with a parameter θ. If $U = 20$, $\theta = 3.0$, then the average number of price clusters in CRP is 6.5. We then assign each product to one of these clusters so that cheaper

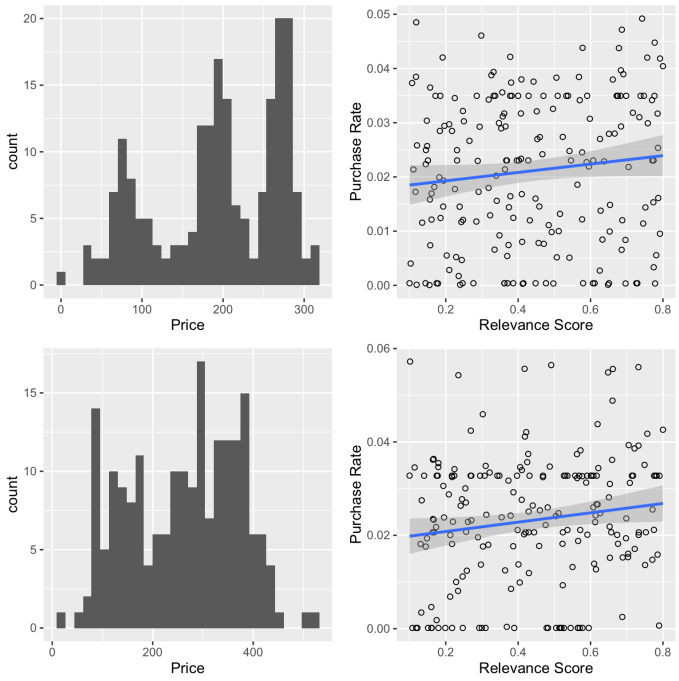

Figure 1: Price histograms and their corresponding relevance score and purchase rate. Note the plot shows correlation between the relevance score and purchase rate fitting a line.

products are assigned to a smaller price cluster number. We denote the price cluster for a product d_i as t_{d_i}.

6.3 User behavior model

In the simulation, we consider that a user u browses through the top k products shown one after another. The following equation provides the expression for the probability of a user u purchasing a product d_i that is located at ranking position j:

$$pr(u, d_i, j) = \begin{cases} c \times p_{d_i} \times \frac{1}{\log_2(j+1)} & \text{if } t_u = t_{d_i} \\ (1-c) \times p_{d_i} & \text{else} \end{cases}$$

We use $c = 0.7$. The factor $\frac{1}{\log_2(j+1)}$ is the position bias.

7 RESULTS

We conducted three experimental studies to evaluate the performance of these algorithms. For each iteration in the study, we use a query and a user uniform randomly. For each of the studies, we use three experiments with each algorithm with the following set-up: (1) $N = 1, |U| = 20, \theta = 3.0$, and iterations=1000, (2) $N = 10, |U| = 20, \theta = 10.0$, and iterations=50000, (3) $N = 10, |U| = 100, \theta = 10$, and iterations=50000

For each experiment, we reported the four metrics from an average of 100 runs of the simulator. We now describe the experiments in the sections below:

7.1 Comparison without Position Bias

In this experiment, we do not use the position bias for computing the probability of purchase. The results are summarized in table 1.

We observed that KPBA and RRBA generate significantly more ARQ and MCV compared to RREC. However, RREC does have better PMRR compared to RRBA. On the other hand, KPBA makes even more ARQ and MCV compared to RRBA, and its PMRR is close to RREC. We conduct experiments with a different number of users and iterations and observe a similar trend in the result. The second row for all three algorithms show the results with 10 queries and 50000 iterations. We see that the KPBA proves to be substantially better than both the algorithms for 50000 iterations except the PMRR is generally similar to RREC. The third row shows the results with 10 queries, 50000 iterations and 100 users who come through a CRP process with $\theta = 30.0$. We observe that KPBA again produces substantially better results for ARQ, MCV compared to the other two algorithms, and it also does very similar to RREC in PMRR metric.

7.2 Comparison with Position Bias

In this section, we repeat all the previous experiments with position bias in the user behavior by introducing a logarithmic decay $\frac{1}{\log k}$ of the probability of purchase based on the rank of the products, where k is the ranking position of the product in an iteration. We again observe a similar pattern that KPBA shows better performance compared to RRBA in all metrics. This result is expected since we have already found better PMRR when we run the experiments without position bias. The table 2 summarizes the experiments. The second and third row of each algorithm uses $\theta = 10$ for the CRP process. Note that, the results for KPBA is significantly better compared to RRBA when position bias is present. Thus, KPBA can be a more practical algorithm for realization.

Algo	Q	U	Iter	ARQ ($)	MCV ($)	PMRR
RREC	1	20	1000	140	7.0	0.41
	10	20	50000	3500	1486	0.54
	10	100	50000	5039	439	0.52
RRBA	1	20	1000	18567	870	0.26
	10	20	50000	138556	64483	0.29
	10	100	50000	144970	14254	0.29
KPBA	1	20	1000	25022	1085	0.41
	10	20	50000	156941	79621	0.52
	10	100	50000	174030	16105	0.52

Table 1: Comparison of KPBA, RRBA and RREC metrics without position Bias.

Algo	Q	U	Iter	ARQ ($)	MCV ($)	PMRR
RREC	1	20	1000	158	8	0.37
	10	20	50000	3500	1486	0.54
	10	100	50000	5039	439	0.52
RRBA	1	20	1000	16922	843	0.19
	10	20	50000	69221	34521	0.29
	10	100	50000	97534	9917	0.31
KPBA	1	20	1000	21497	1085	0.40
	10	20	50000	109086	53716	0.53
	10	100	50000	107091	10655	0.54

Table 2: Comparison of KPBA, RRBA and RREC with position Bias.

Algo	Q	U	Iter	ARQ ($)	MCV ($)	PMRR
RREC	1	20	1000	117	6	0.35
	10	20	50000	3100	1134	0.50
	10	100	50000	17111	1634	0.57
RRBA	1	20	1000	20288	1019	0.17
	10	20	50000	43654	25345	0.25
	10	100	50000	95281	9540	0.27
KPBA	1	20	1000	23201	1125	0.49
	10	20	50000	95754	42876	0.53
	10	100	50000	103324	10315	0.57

Table 3: Comparison of KPBA, RRBA and RREC with Changing Customer Preferences.

7.3 Comparison with Changing Customer Preferences

In this section, we experiment with changing customer preferences after every 500 iteration. We summarize the experiments in table 3. We again notice that the KPBA algorithm still performs better than RRBA in all three metrics, and it compares favorably in PMRR compared to RREC.

7.4 Comparison of convergences for the three algorithms

To show the growth of the four metrics under three above mentioned scenarios, we show the growth of two main metrics ARQ and MCV from our simulation study collecting these metrics for every 100-th iteration. The figure 2 uses 5 queries and 20 users with $\theta = 3.0$ and shows the comparison of the two metrics without position bias for all three algorithms. The figure 3 shows those two metrics for all three algorithms with position bias. The figure 4 shows three metrics with changing customer preference for every 500 iterations along with the position bias. It is very clear from the plots that the KPBA performs consistently better than RRBA and RREC in ARQ, and MCV metrics. Note, all the prices are in log scale on the convergence plots. The figure 4 shows that KPBA has a similarly good performance in PMRR metric compared to RREC.

7.5 Comments on experimental evaluation

We find that as expected from the analytical study, KPBA performs well in ARQ metric compared to RRBA. RREC performs worst in revenue metrics. We understand from our simulation studies that MCV metric is also significantly better for KPBA compared to RRBA and RREC. Moreover, the PMRR values in KPBA are similar or better compared to RREC and are far better compared to RRBA based on the result of the simulation and as discussed in our theoretical analysis. We also observe that KPBA continues to perform better with more iterations, the number of users, with position bias, and with changing user preferences.

8 CONCLUSION

In this paper, we introduce the learning to diversify problem for e-commerce search. We show that in order to serve best for both the company and the customers, in e-commerce, it is required to construct a unique formulation of the learning to diversify problem where we intend to learn to diversify, and maximize the revenue simultaneously, and as well as ensure a value of relevance for the top k results. Our theoretical results show that the KPBA algorithm is expected to have better ARQ, and PMRR compared to RRBA. Our simulation studies show that KPBA also has better MCV compared to both RRBA and RREC, and it also gives a good performance in terms of PMRR compared to RREC. On the other hand, RRBA is quite bad from the customer's perspective for this problem since the PMRR is low in all scenarios based on our simulations. In essence, we can show that the KPBA can be an efficient and practical algorithm for diversifying e-commerce search results. We also show that e-commerce companies can improve the CLV by using a diverse ranking strategy. This connection between diversity in ranking and CLV can be worth exploring more in the future. KPBA can also potentially be further optimized by formulating a variable budget knapsack problem where we simultaneously also learn the optimal relevance threshold. It is also possible to find a better probabilistic approximation algorithm for such optimization problems [34]. We anticipate that this paper can motivate further research in the area of diverse ranking for e-commerce search and recommender systems.

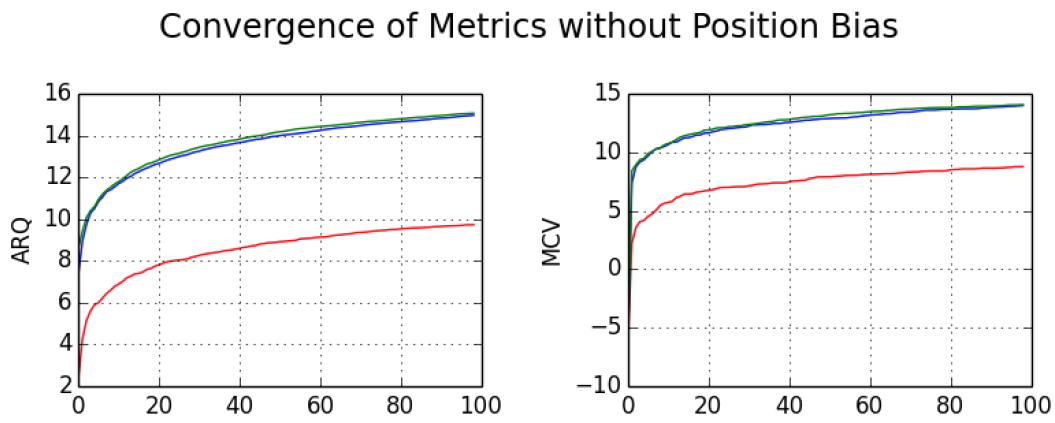

Figure 2: Note that the red curves represent RREC metrics, blue curves represent RRBA metrics and the green curves denote KPBA metrics. The revenue metric uses log scale.

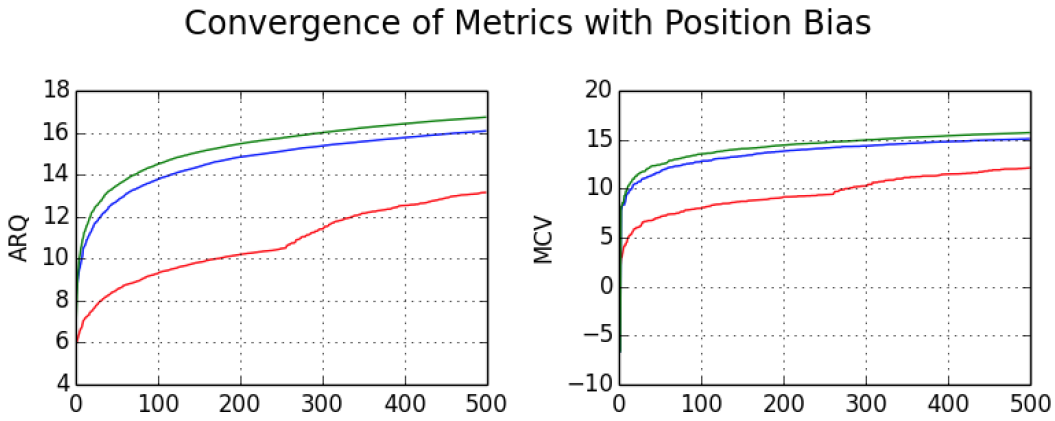

Figure 3: Note that the red curves represent RREC metrics, blue curves represent RRBA metrics and the green curves denote KPBA metrics. The revenue metric uses log scale.

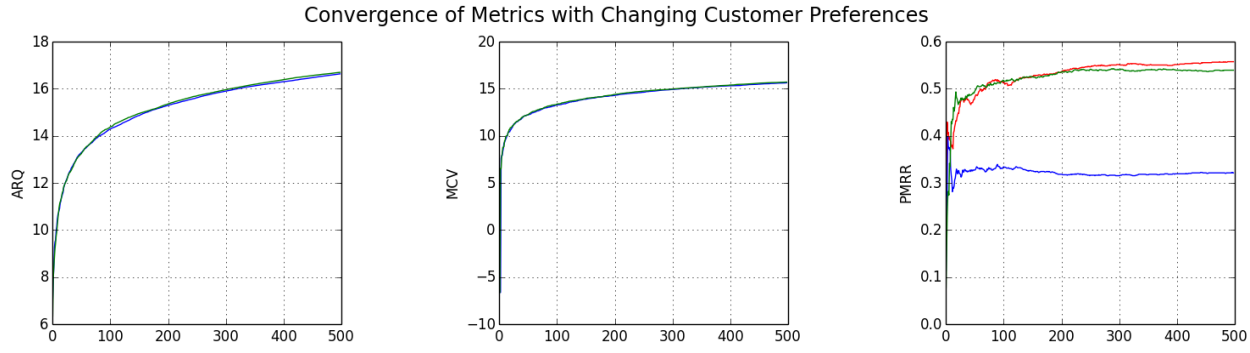

Figure 4: Note that the red curves represent RREC metrics, blue curves represent RRBA metrics and the green curves denote KPBA metrics. The revenue metric uses log scale.

REFERENCES

[1] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, Nitin Motgi, Seung-Taek Park, Raghu Ramakrishnan, Scott Roy, and Joe Zachariah. 2009. Online models for content optimization. In *Advances in Neural Information Processing Systems*. 17–24.

[2] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying search results. In *Proceedings of the second ACM international conference on web search and data mining*. ACM, 5–14.

[3] Shipra Agrawal, Vashist Avadhanula, Vineet Goyal, and Assaf Zeevi. 2016. A Near-Optimal Exploration-Exploitation Approach for Assortment Selection. In *Proceedings of the 2016 ACM Conference on Economics and Computation*. ACM, 599–600.

[4] David J Aldous. 1985. Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIIIâĂŤ1983*. Springer, 1–198.

[5] Albert Angel and Nick Koudas. 2011. Efficient diversity-aware search. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 781–792.

[6] Jean-Yves Audibert, Sébastien Bubeck, and Gábor Lugosi. 2013. Regret in online combinatorial optimization. *Mathematics of Operations Research* 39, 1 (2013), 31–45.

[7] Peter Auer. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3, Nov (2002), 397–422.

[8] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. 2002. The nonstochastic multiarmed bandit problem. *SIAM journal on computing* 32, 1 (2002), 48–77.

[9] Hans H Bauer, Tomas Falk, and Maik Hammerschmidt. 2006. eTransQual: A transaction process-based approach for capturing service quality in online shopping. *Journal of Business Research* 59, 7 (2006), 866–875.

[10] Alberto Caprara, Hans Kellerer, and Ulrich Pferschy. 1998. Approximation Algorithms for Knapsack Problems with Cardinality Constraints. *European Journal of Operational Research* 123 (1998), 2000.

[11] Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 335–336.

[12] Olivier Chapelle, Yi Chang, and T-Y Liu. 2011. Future directions in learning to rank. In *Proceedings of the Learning to Rank Challenge*. 91–100.

[13] Harr Chen and David R Karger. 2006. Less is more: probabilistic models for retrieving fewer relevant documents. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 429–436.

[14] Wei Chu, Lihong Li, Lev Reyzin, and Robert E Schapire. 2011. Contextual Bandits with Linear Payoff Functions.. In *AISTATS*, Vol. 15. 208–214.

[15] Charles LA Clarke, Maheedhar Kolla, Gordon V Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 659–666.

[16] Nick Craswell. 2009. Mean reciprocal rank. In *Encyclopedia of Database Systems*. Springer, 1703–1703.

[17] Olayinka David-West. 2016. E-Commerce Management in Emerging Markets. *Encyclopedia of E-Commerce Development, Implementation, and Management* 1 (2016), 200.

[18] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. 2010. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems*. 586–594.

[19] Robert S Garfinkel and George L Nemhauser. 1972. *Integer programming*. Vol. 4. Wiley New York.

[20] Jennifer Gimson. 2017. Why Lifetime Value is the Most Important Metric in eCommerce. https://crealytics.com/blog/lifetime-value-important-metric-ecommerce/. (2017). [Online; accessed 27-January-2018].

[21] Anjan Goswami, Wei Han, Zhenrui Wang, and Angela Jiang. 2015. Controlled experiments for decision-making in e-Commerce search. In *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 1094–1102.

[22] David Hawking, Paul Thistlewaite, and Nick Craswell. 1997. Anu/acsys trec-6 experiments. In *TREC*. Citeseer, 275–290.

[23] Dipak Jain and Siddhartha S Singh. 2002. Customer lifetime value research in marketing: A review and future directions. *Journal of interactive marketing* 16, 2 (2002), 34–46.

[24] Thorsten Joachims and Adith Swaminathan. 2016. Counterfactual evaluation and learning for search, recommendation and ad placement. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 1199–1201.

[25] H. Kellerer, U. Pferschy., and D. Pisinger. 2003. *Knapsack Problems*. Springer.

[26] Hans Kellerer, Ulrich Pferschy, and David Pisinger. 2004. *Introduction to NP-Completeness of knapsack problems*. Springer.

[27] Samir Khuller, Anna Moss, and Joseph Seffi Naor. 1999. The budgeted maximum coverage problem. *Inform. Process. Lett.* 70, 1 (1999), 39–45.

[28] Junpei Komiyama, Junya Honda, Hisashi Kashima, and Hiroshi Nakagawa. 2015. Regret Lower Bound and Optimal Algorithm in Dueling Bandit Problem.. In *COLT*. 1141–1154.

[29] Kenneth C Laudon, Carol Guercio Traver, and Alfonso Vidal Romero Elizondo. 2007. *E-commerce*. Vol. 29. Pearson/Addison Wesley.

[30] Jessica Lee. 2013. No. 1 Position in Google Gets 33% of Search Traffic [Study]. https://searchenginewatch.com/sew/study/2276184/no-1-position-in-google-gets-33-of-search-traffic-study. (2013). [Online; accessed 27-January-2018].

[31] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 297–306.

[32] Silvano Martello and Paolo Toth. 1990. Knapsack problems: Algorithms and computer interpretations. *Hoboken, NJ: Wiley-Interscience* (1990).

[33] Daniel McFadden. 1980. Econometric models for probabilistic choice among products. *Journal of Business* (1980), S13–S29.

[34] S Muthukrishnan, Martin Pál, and Zoya Svitkina. 2007. Stochastic models for budget optimization in search-based advertising. In *International Workshop on Web and Internet Economics*. Springer, 131–142.

[35] George L Nemhauser and Laurence A Wolsey. 1978. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research* 3, 3 (1978), 177–188.

[36] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. 2008. Learning Diverse Rankings with Multi-Armed Bandits. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*.

[37] Rodrygo L.T. Santos, Craig Macdonald, and Iadh Ounis. 2010. Selectively Diversifying Web Search Results. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM '10)*. ACM, New York, NY, USA, 1179–1188.

[38] Denis Sauré and Assaf Zeevi. 2013. Optimal dynamic assortment planning with demand learning. *Manufacturing & Service Operations Management* 15, 3 (2013), 387–404.

[39] Daria Sorokina and Erick Cantu-Paz. 2016. Amazon Search: The Joy of Ranking Products. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. ACM, New York, NY, USA, 459–460.

[40] Unknown. 2013. The Moore?s law of e-commerce or the crucial importance of being first. https://www.mabaya.com/the-moores-law-of-e-commerce-or-the-crucial-importance-of-being-first/. (2013). [Online; accessed 27-January-2018].

[41] Unknown. 2017. Relevance Scores: Understanding and Customizing. https://docs.marklogic.com/guide/search-dev/relevance. (2017). [Online; accessed 27-January-2018].

[42] Cong Yu, Laks Lakshmanan, and Sihem Amer-Yahia. 2009. It takes variety to make a world: diversification in recommender systems. In *Proceedings of the 12th international conference on extending database technology: Advances in database technology*. ACM, 368–378.

[43] Yisong Yue and Carlos Guestrin. 2011. Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems*. 2483–2491.

[44] Cheng Xiang Zhai, William W Cohen, and John Lafferty. 2003. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 10–17.