Thompson Sampling for Dynamic Pricing

Ravi Ganti*, Matyas Sustik, Quoc Tran, and Brian Seaman

Walmart Labs, San Bruno, CA, USA

February 12, 2018

Abstract

In this paper we apply active learning algorithms for dynamic pricing in a prominent e-commerce website. Dynamic pricing involves changing the price of items on a regular basis, and uses the feedback from the pricing decisions to update prices of the items. Most popular approaches to dynamic pricing use a passive learning approach, where the algorithm uses historical data to learn various parameters of the pricing problem, and uses the updated parameters to generate a new set of prices. We show that one can use active learning algorithms such as Thompson sampling to more efficiently learn the underlying parameters in a pricing problem. We apply our algorithms to a real e-commerce system and show that the algorithms indeed improve revenue compared to pricing algorithms that use passive learning.

1 Introduction

The Internet has enabled many industries to go online. e-commerce which entails selling of goods both physical and digital, via the Internet, is one of the vibrant industries in modern times. The presence of gigantic all-purpose e-commerce retailers such as Walmart.com, Amazon.com, Jet.com, Target.com, and the more specialized and niche e-commerce retailers such as Moosejaw, Modcloth.com, Shoebuy.com, Wine.com, Bonobos.com has shifted commerce from offline medium to online medium. While the biggest brick-and-mortar stores are constrained by shelf space and tend to stock only the 100K most popular items, the largest of e-commerce retailers are less constrained by shelf space and tend to carry more than 400 million items. e-commerce retailers have also benefited from the frictionless, digital economy, and the explosion of online activity in general. It is now not hard to study customer behaviour online such as what items a customer viewed on a website, what items are popular in general, what items a customer bought in the past, and use this personalized information along with information from other sources to come up with actionable insights to entice and retain customers.

*rmahapatruni@walmartlabs.com

One of the most important problem any retailer has to solve is, how to price items correctly. Setting the prices too high or too low can cause bad customer experience and dramatically effect the bottom-line of the retailer. The size of the catalog carried by a brick-and-mortar store is very small, and because of the difficulty in changing prices, pricing of an items is an easy problem and can be solved by carrying out elementary data analysis in excel. However, the presence of a large collection of products on an e-commerce website, the ability to collect large amounts of user behaviour data easily, and also the relative ease with which price changes can be made on a large scale, makes the problem of pricing both challenging and rich. In traditional brick-and-mortar stores price of items tend to remain unchanged. However, such static pricing policies do not work in an e-commerce setting. Static pricing does not simulate the demand/price curve, nor does it allow a firm to respond to competitor price changes. Moreover static pricing is unable to fully exploit the power of computation, data, and the ease of implementing price changes in an online world, As a result, dynamic pricing is the pricing policy of choice in an e-commerce setting.

In this paper we study algorithms for dynamic pricing. Dynamic pricing in an e-commerce setting is a hard problem because of the scale of the problem and uncertain user behaviour. It is not clear how one can use the past collected data to come up with better prices. A typical approach to dynamic pricing consists of performing the following four steps: (i) Collect historical data about the price of an item and the demand of item at different price points, (ii) Posit a statistical model for the demand as a function of price, and estimate the model parameters using historical data, (iii) Using the learned demand function, optimize some metric of interest (e.g. revenue) to get the new optimal price, (iv) apply the obtained optimal price to the item for the next few days and repeat the above. We classify these approaches to dynamic pricing as passive learning approaches. Such approaches are myopic and try to optimize for the metric in the short term without trying to make an active effort to learn the demand function. In fact such approaches have shown to lead to incomplete learning (Harrison et al., 2012), and have poor performance, and lose revenue on the long run. In this paper we show how one can avoid this problem. Our contributions are as follows.

1. *We propose and implement a simple, scalable, active learning algorithm for dynamic pricing.* While passive learning based dynamic pricing approaches treat the pricing problem as a vanilla optimization problem of optimizing the cumulative metric of interest over a finite horizon T, we introduce an algorithm called MAX-REV-TS that views dynamic pricing problem as an optimization problem under uncertainty.

2. We begin by explaining the architecture of a dynamic pricing system that is currently in place. Our DP system makes precise parametric assumptions on the demand function and maximizes a metric of interest daily under constraints. To make our discourse concrete we shall use the revenue function as the metric of interest in this paper [1]. The parameters of the demand function are updated regularly. Since this DP system is based on passive learning, we shall call this system as MAX-REV-PASSIVE .

[1] We shall use the words revenue and reward interchangeably.

3. We then introduce a new DP system, MAX-REV-TS which is very similar to MAX-REV-PASSIVE , but one which uses a multi-armed bandit (MAB) algorithm called Thompson sampling (Thompson, 1933). MAX-REV-TS uses the same parametric formula for the demand function as MAX-REV-PASSIVE and proceeds to maximize revenue over a fixed horizon. TS operates as follows: A Bayesian prior is put on the unknown parameters of the demand function model and using the Bayes rule the posterior is derived. We use a conjugate prior-likelihood pair to obtain posterior updates in closed form. TS then samples a parameter from the posterior distribution and derive a new set of prices by maximizing the reward function for the next time step using the sampled parameters. This process is repeated until the end of the horizon.

4. The main focus of this paper is provide an implementation of MAX-REV-TS in a major e-commerce system. We discuss various practical issues that we had to take care of when applying TS in a real production environment. We show results of our implementation in a 5 week period and compare it to the MAX-REV-PASSIVE algorithm. The results show that MAX-REV-TS shows a statistically significant improvement over passive learning algorithms for dynamic pricing under appropriate conditions. To the best of our knowledge we are the first to demonstrate active DP algorithms in a real e-commerce production system. Furthermore, while typical applications of MAB algorithms in production systems have dealt with the case when there are only finite number of arms Agarwal et al. (2016); Shah et al. (2017), our paper is the first to demonstrate practical application of a more involved bandit optimization problem which has infinite arms.

2 Related Work

DP has been an active area of research for more than two decades and has seen considerable amount of attention in operations research (OR), management sciences (MS), marketing, economics and computer sciences. A lot of work in dynamic pricing was traditionally carried out in OR/MS, and economics communities. It was pioneered by airline industry, where prices are adjusted over time via capacity control, and dynamic programming solutions are used to decide the fares Talluri & Van Ryzin (2006); Phillips (2005). DP has been considered under various settings such as pricing with and without inventory constraints, pricing with single or multiple items, finite or infinite horizon DP, DP with known or unknown demand function. It is not possible to give a thorough literature survey of dynamic pricing. We refer the interested reader to the excellent survey of den Boer (2015) for a comprehensive survey on DP. In this section we give a limited survey of DP.

In economics, the early works of Rothschild (1974); Aghion et al. (1991) considered the problem of DP with unknown demand model. In these papers, the dynamic pricing problem is cast as a decision making problem in an infinite horizon, with discounted rewards. A Bayesian setting is used and a solution to the dynamic pricing problem is given via Bayesian dynamic programming. These papers focused on inves-

tigating if following the optimal policy obtained via dynamic programming was helpful in recovering the unknown demand information. While, such dynamic programming based policies might lead to an optimal pricing scheme, the approaches tend to be intractable in practice.

In the OR/MS community most work on dynamic pricing has focused on the case of known demands, but with other constraints such as inventory constraints. The problem of DP under unknown demand constraints has only gained traction in this community since mid 2000s. Aviv & Pazgal (2005) were one of the first to look at the problem of dynamic pricing with unknown demand function in this community. They assume that the demand function comes from a parametric family of models with unknown parameter values. They then use a Bayesian approach, where a prior distribution is used on the unknown parameters, and the posteriors are obtained via the Bayes theorem. The problem is modeled as an infinite horizon, discounted revenue maximization problem. They then introduce certainty-equivalence heuristic which is an approximate dynamic programming solution to obtain a DP solution. Araman & Caldentey (2009); Farias & Van Roy (2010) used the same setup as in Aviv & Pazgal (2005) but introduced different approximate dynamic programming heuristics to provide a solution to the dynamic pricing problem. While Bayesian, approaches such as the ones mentioned above are attractive as they let you model the parameter uncertainties explicitly, they often lead to intractabilities, such as not being able to compute the posterior in closed form. Non-Bayesian approaches offer computational benefits over Bayesian methods. In the non-Bayesian context Carvalho & Puterman (2005a,b) investigate non-myopic policies for dynamic pricing under unknown demand function. They proposed a variant of one-step lookahead policies which instead of maximizing the revenue at the next step, maximizes revenue for the next two steps. They applied this semi-myopic policy to a binomial demand distribution with logit expectation and compared the performance of this policy with other myopic policies. They showed via simulations that one-step lookahead policies could significantly outperform myopic policies. Various other semi-myopic policies have also been investigated in Carvalho & Puterman (2005b).

Semi-myopic policies such as one-step lookahead policies in the non-Bayesian context, and approximate Bayesian dynamic programming based solutions provide tractable dynamic pricing policies. However, these approaches lack theoretical guarantees. A lot of recent work in dynamic pricing has been to obtain suboptimal, yet tractable dynamic pricing policies with provable guarantees. Besbes & Zeevi (2009) take a non-Bayesian approach to dynamic pricing. They use exploration-exploitation style algorithms where in the first phase (exploration), multiple prices are tried out and demand function estimated at these multiple price points, and in the second phase the optimal price is chosen by solving a revenue maximization problem. Harrison et al. (2012) considered the problem of DP under the assumption that the true model is one of two given parametric models. They then showed that myopic policies that try to maximize the immediate revenue can fail and myopic Bayesian (and also non-Bayesian policies) can lead to convergence to an incorrect model and hence what they term as incomplete learning. Broder & Rusmevichientong (2012); den Boer & Zwart (2013); Keskin & Zeevi (2014) look at various parametric models for the demand function and derive exploration-exploitation based policies using maximum likelihood based formulations. Our work is very similar to the above exploration-exploitation style DP

approaches that maximize cumulative revenue over a finite horizon.

Finally, we would like to mention the very recent work of Ferreira et al. (2016) who study the problem of DP of multiple items under finite horizon, with inventory constraints, and unknown demand function. They provide a Thompson sampling based solution to maximize the revenue for all the items under the given constraints. The proposed algorithms show promising performance on small, synthetic datasets.

Thompson sampling has been used to tackle the problem of exploration-exploitation in other domains such as computational advertising, feed ranking, reinforcement learning. An exhaustive list is out-of-scope of this paper and we refer the interested reader to the excellent tutorial of Russo et al. (2017).

Notation. Vectors and matrices are denoted by bold letters. Given a vector $\boldsymbol{x}_t$, $x_{i,t}$ denotes the i^{th} dimension of $\boldsymbol{x}_t$.

3 Architecture of a dynamic pricing system

Any commercial dynamic pricing system needs to figure out how to model the demand function and then how to use the modeled demand function to calculate optimal prices to be applied every day. The DP system that we introduce in this paper has different components which all implement statistical learning and convex optimization algorithms to generate prices daily. We shall now look at each of these components in brief.

Demand modeling. A standard approach to demand modeling in pricing problems is to assume that the demand function comes from some parametric family, and then estimate the parameters of the underlying function using statistical techniques. Lot of parametric models have been investigated in pricing and revenue management literature such as linear models, log-linear models, constant elasticity models and logit model Talluri & Van Ryzin (2006). Parametric modeling has been particularly popular because it allows one to build simple, explainable models and also allows for simple pricing algorithms. However, commonly used models, both parametric and non-parametric, in demand modeling tend to assume that the underlying demand function is stationary. In reality the demand function is not stationary and one needs to model the dynamic nature of the demand function in real world pricing systems. In order to do this we adapt commonly used demand functions to account for possible changes in the underlying demand function. As previously mentioned a commonly used demand function is the constant elasticity model, which models the demand of item i as

$$d_i(p_i) = f_i \left(\frac{p_i}{p_{0,i}} \right)^{\gamma_{\star,i}} \tag{1}$$

Here, $d_i(p_i)$ is the demand of item i at price $p_{0,i}$, f_i is the baseline demand at price $p_{0,i}$ for item i and $\gamma_{\star,i} < -1$ is the price elasticity of item i. In order to account for possible changes in the underlying demand function we instead model the demand function, at time t for item i, using the equation

$$d_{i,t}(p_i) = f_{i,t} \left(\frac{p_i}{p_{i,t-1}} \right)^{\gamma_{\star,i}} \tag{2}$$

Here $f_{i,t}$ is the demand forecast for item i on day t if the price is $p_{i,t-1}$, and $\gamma_{\star,i}$ is the price elasticity of item i. Our DP engine has two dedicated components namely demand forecasting component, and elasticity estimation component that estimate $f_{i,t}$, $\gamma_{\star,i}$ every day.

Demand forecasting component. The demand forecasting module at time t, for item i, takes the observed demand for the item i in the past at different price points $((p_{i,1}, d_{i,1}), \ldots (p_{i,t-1}, d_{i,t-1}))$ and estimates $f_{i,t}$. The demand forecasting module uses multiple models for forecasting, such as time-series models ARMA, ARIMA, EWMA, AR(k) along with non-time series models such as logistic regression and combines these models to get a new model. The weights using in the combination of the models depends on the past performance of individual models. For the purpose of this paper it is enough to treat this as a black-box and one can assume that the demand forecasts are available for each item on each day.

Price elasticity component. The price elasticity component estimates the elasticity parameter $\gamma_{\star,i}$, for each item i. $\gamma_{\star,i}$ captures how the demand of item i changes with its price. If $p_{i,t}$ is close to $p_{i,t-1}$ then one can approximate the above exponential form by the following linear equation

$$d_{i,t}(p_i) \approx f_{i,t} + (p_i - p_{i,t-1}) \frac{f_{i,t}\gamma_{\star,i}}{p_{i,t-1}}. \tag{3}$$

From the above linear approximation, elasticity estimation can be reduced to a linear regression problem which can be solved using either ordinary least squares (OLS) approach or via robust linear squares(RLS). We use RLS for elasticity estimation. Furthermore, instead of looking at all of the past data for an item, one can focus only on the most recent data (say 1 month or 2 months) to estimate elasticities. Estimating elasticities is generally much more harder than demand forecast. This is because if an item has only k distinct price points in the last few months then one needs to estimate elasticity for this item using only k data points. It is not uncommon to see k being less than 5 for a lot of items. Hence, elasticity estimation is plagued by data sparsity issue. In such cases one can mitigate the problem to some extent by using multi-task learning techniques Caruana (1998) to group items with similarly elasticities and estimate the elasticity of these items together.

Optimization component. The last component is an optimization engine that outputs prices of different items. For the sake of convenience, it is natural to put multiple items into a single basket, and impose various constraints that the prices the items in the basket could take. These constraints are typically set by business needs, and include constraints such as minimum and maximum prices the items in the basket can have, bounds on how much the price of an item in the basket could change each day, and some basket-wide level constraints such as minimum margin on the basket. If this item is sold by other e-commerce retailers then it is natural to include the prices of the other e-commerce retailers in the constraint set. These constraints can change from time to time, but at each point in time are known to the optimization engine. For the purpose of this paper it is not necessary for us to focus on the exact geometry of the constraint set, but it suffices to assume that the constraint set is always a feasible, convex set, and "simple" enough to enable optimization. The revenue of an item i in a basket $\mathcal{B}$ can be

Algorithm 1: Architecture of the proposed dynamic pricing engine.

Input: A basket $\mathcal{B}$, and time period T over which we intend to maximize cumulative revenue

for $t \leftarrow 1$ ***to*** T **do**

1. For each item $i \in \mathcal{B}$ calculate their demand forecasts using the demand forecaster.
2. For each item $i \in \mathcal{B}$ calculate their price elasticities $\gamma_{\star,i}$.
3. Solve the MAX-REV optimization problem, shown in Equation (7) to obtain new prices $\mathbf{p}_t$.
4. Apply these prices and observe the revenue obtained R_t.

end

estimated as

$$Rev_{i,t}(p_{i,t}) = p_{i,t} \times d_{i,t}(p_{i,t}) \tag{4}$$

$$\approx p_{i,t}(f_{i,t} + (p_{i,t} - p_{i,t-1})\,\frac{f_{i,t}\gamma_{\star,i}}{p_{i,t-1}}) \tag{5}$$

$$= \frac{p_{i,t}^2 f_{i,t}\gamma_{\star,i}}{p_{i,t-1}} - p_{i,t}f_{i,t}\gamma_{\star,i} + p_{i,t}f_{i,t} \tag{6}$$

where the first line corresponds to the definition of revenue, and the second line comes from the linear approximation of the demand function as shown in equation (3). Let, $\mathbf{p} = [p_1, p_2, \ldots, p_{|\mathcal{B}|}]$ be a vector of prices of items in the basket $\mathcal{B}$. The optimization engine solves the following optimization problem using estimates for the quantities $\gamma_{\star,i}, f_{i,t}$ obtained from the elasticity estimation and demand forecasting module respectively.

$$\mathbf{p}_t = \arg\max_{\mathbf{p}} \sum_{i \in \mathcal{B}} \frac{p_i^2 f_{i,t}\gamma_{\star,i}}{p_{i,t-1}} - p_i f_{i,t}\gamma_{\star,i} + p_i f_{i,t} \tag{7}$$

$$\text{subject to: } \mathbf{p} = [p_1, p_2, \ldots, p_{|\mathcal{B}|}] \in \mathcal{C}_t$$

Since price elasticities are negative, the above optimization problem is maximizing a concave function subject to convex constraints. Hence, it is a convex optimization problem that can be solved using standard techniques Bertsekas & Scientific (2015). We call the optimization problem in Equation (7) as MAX-REV optimization problem as it tries to maximize the revenue of a basket under constraints. When MAX-REV is used along with passive elasticity estimation techniques (for example using OLS or RLS as mentioned above) we call the resulting algorithm as MAX-REV-PASSIVE . The entire architecture can be summarized in Figure (1).

4 Towards active algorithms for dynamic pricing

The MAX-REV-PASSIVE algorithm provides a dynamic pricing algorithm that estimates demand forecasts and elasticity passively, and uses these estimates to solve a MAX-REV optimization problem. As mentioned in the introduction, dynamic pricing algorithms that use passive learning are incomplete learning algorithms and do not maximize revenue in the long run. From a practical standpoint, estimating demand forecasts and elasticity is not easy. Price elasticity of an item reflects how the demand changes when the price of an item changes. If the price of an item does not change often (a typical case in e-commerce) then it becomes very hard to estimate the price elasticity, which can in turn lead to fewer price changes, and sub-optimal price changes. Moreover, due to the poor estimation of elasticities, there is high uncertainty in elasticity estimates. One approach to factor into account the inaccuracies in elasticities is to model the MAX-REV problem as a robust optimization problem Ben-Tal et al. (2009). In a robust optimization formulation of MAX-REV , one considers the elasticity parameter as an uncertain quantity and tries to maximize revenue w.r.t. the worst possible value that the elasticity can take. For example, the robust MAX-REV problem would be

$$\mathbf{p}_t = \arg\max_{\mathbf{p}} \min_{\gamma_\star \in \mathcal{G}} \sum_{i \in \mathcal{B}} \frac{p_i^2 f_{i,t} \gamma_{\star,i}}{p_{i,t-1}} - p_i f_{i,t} \gamma_{\star,i} + p_i f_{i,t}$$
$$\text{subject to: } \mathbf{p} = [p_1, p_2, \ldots, p_{|\mathcal{B}|}] \in \mathcal{C}_t$$

where the set $\mathcal{G} \subset \boldsymbol{R}^{|\mathcal{B}|}$ is a set of possible values for $\gamma_\star$. The robust optimization formulation replaces a point estimate for the vector $\gamma_\star$ with one of infinite possible values, constrained to be in the set $\mathcal{G}$. The problem with this approach is that this formulation can be very pessimistic, and there is no clear way to incorporate feedback to update the set $\mathcal{G}$ as we gather more data.

An alternative approach to handle uncertainties in the parameters is to consider the MAX-REV problem as an optimization problem under uncertainty. This allows us to systematically model the uncertainty in our parameters, and use the feedback from our previous pricing actions to update parameter uncertainties.

4.1 Decision making under uncertainty

Decision making under uncertainty is a broad area, which as the name suggest, focuses on how to make decisions to maximize some objective, when there is uncertainty in estimates related to the objective function. This broad class of problems has found various applications such as in search, online advertising, online recommendations, route planning. For example in search it is not clear what the click-through-rate (CTR) of certain documents are because these documents have never been shown to users or has been shown very few times. Solutions to such problems involves exploration, which means that we make potentially sub-optimal decisions as long as the feedback from these decisions can improve our estimates. While making sub-optimal decisions (exploration) can help gather more information it can also hamper our experience, and therefore one needs to be careful when trying to explore. This problem is known as

exploration-exploitation trade-off. Passive learning algorithms such as MAX-REV-PASSIVE shown in Figure (1) are focused purely on exploitation and do not explore enough and hence lose out on revenue in the long run. Various algorithms and models have been devised to handle the exploration-exploitation trade-off. The best known model is the multi-armed bandit model (MAB), which can be described as follows: There are k arms, and choosing an arm gives an i.i.d. reward from a fixed unknown probability distribution that depends on the arm. Choosing an arm i gives no information about any other arm $j \neq i$. An agent is tasked with obtaining the maximum possible reward in T rounds, where in each round she chooses one of the k arms and obtains an i.i.d. reward associated with the arm distribution. A simple algorithm is to pull each arm once (or may be a few times each), calculate the average reward of each arm and then for the rest of the time choose the arm which gave the maximum reward. This simple solution is sub-optimal as it is too greedy and suffers the same problem as the MAX-REV-PASSIVE algorithm. Another simple solution is to choose the arm with the best reward each time with probability $1 - \epsilon$, but with probability ϵ choose a random arm. When $\epsilon = 0$, this reduces to a greedy algorithm, when $\epsilon = 1$ this becomes a purely explorative algorithm. However, with the right choice of ϵ, it is possible to do well. Such algorithms are known as ϵ-greedy algorithms. The performance of these algorithms are measured in terms of regret, which tells us how well the algorithm does in comparison to an oracle algorithm that knows the best arm a priori. If the average regret of the algorithm converges to 0, then the algorithm is said to be consistent. While ϵ-greedy is consistent for the right value of ϵ, better algorithms exist, two of which we cover here in brief. A generalization of the multi-armed bandit problem is the bandit optimization problem where one tries to solve an optimization problem with certain parameters being unknown. For example, in bandit linear optimization problem one tries to solve the optimization problem $\max_{\boldsymbol{x} \in \mathcal{C}} \boldsymbol{x}^\top \boldsymbol{\theta}_\star$, where $\mathcal{C}$ is a known polytope and $\boldsymbol{\theta}_\star$ is an unknown vector. The bandit game proceeds in rounds where in each round the learner proposes a $\boldsymbol{x}$, and gets to see a noisy evaluation of the objective function $\boldsymbol{x}^\top \boldsymbol{\theta}_\star$. For an extensive survey of various bandit algorithms and bandit models we refer the interested reader to the survey of Bubeck et al. (2012).

The Upper Confidence Bound (UCB) algorithm Auer et al. (2002b,a) constructs confidence bounds on the reward of each arm, and chooses the arm which has the largest upper confidence bound. The UCB algorithms and its variants are optimal, consistent algorithms. Implementing UCB requires one to accurately estimate confidence bounds, which might not be easy in complex decision making problems.

An alternative to the UCB algorithm is the Thompson Sampling algorithm (TS) Thompson (1933); Gopalan et al. (2014); Russo et al. (2017). TS is a Bayesian algorithm, which places a prior distribution on the reward of the arms, and this distribution gets updated, using the Bayes rule, as one gathers feedback from previous actions. The TS algorithm is based on probability matching, i.e. an arm is chosen with probability equal to the probability of the arm being optimal. Like UCB, TS is also a consistent algorithm, but can be easier to implement than the UCB algorithm, even in complex decision making problems. This is because TS avoids explicitly constructing confidence intervals around the uncertain parameters. These properties of TS make it an attractive choice of an explore-exploit algorithm for real-world applications.

4.2 Dynamic pricing as a bandit optimization problem and Thompson sampling

In order to apply TS to our problem of maximizing revenue, we shall reformulate our MAX-REV problem as a bandit optimization problem. This is shown in game (2). Our bandit formulation explicitly models the uncertainty in the true elasticity values, but assumes that there is no uncertainty in our demand forecasts. That is, we assume that the demand forecasting module is perfect and spits out accurate forecasts. This assumption is motivated by the fact that demand forecasts are relatively easier to estimate when compared to price elasticities, as they do not suffer from data sparsity problem. For this reason, our bandit formulation of DP focuses on modeling uncertainties in price elasticities.

Game 2: MAX-REV problem as a bandit optimization problem

Input: A basket, $\mathcal{B}$, containing B items, and the time period T over which we intend to maximize revenue.

Oracle chooses a vector $\gamma_\star \preceq 0$, but does not reveal it to the player;
for $t \leftarrow 1$ ***to*** T **do**
    The player plays a price vector $\mathbf{p}_t$ for the basket $\mathcal{B}$;
    The player obtains a stochastic reward (revenue) R_t, such that $\boldsymbol{E}[R_t] = Rev_t(\mathbf{p}_t; \gamma_\star, f_{1,t}, \dots f_{B,t})$ where Rev_t is given by Equation (10) and the demand forecast used in Equation (10) are assumed to have been calculated without any uncertainty using the demand forecast component of our dynamic pricing system;
end

TS begins by putting in a prior distribution over the unknown parameters. Since, $\gamma_\star$ is unknown to us, we shall put a prior distribution over $\gamma_\star$. Since we observe the revenue corresponding to out pricing decisions each day, we put a likelihood model on the observed revenue. Let, R_t be the observed revenue of a basket [2]. Our probabilistic model is given by the following equations

$$\Pi_0(\gamma_\star) = N(\boldsymbol{\mu}_0, \Sigma_0). \tag{8}$$

$$l(R_t; Rev_t, \gamma_\star) = N(R_t; Rev_t, \sigma^2). \tag{9}$$

$$Rev_t(\mathbf{p}; \gamma_\star, \boldsymbol{f}_t) = \sum_{i \in \mathcal{B}} \frac{p_i^2 f_{i,t} \gamma_{\star,i}}{p_{i,t-1}} - p_i f_{i,t} \gamma_{\star,i} + p_i f_{i,t}. \tag{10}$$

$$\Pi_t(\gamma_\star) \propto \Pi_{t-1}(\gamma_\star) N(R_t; Rev_t, \sigma^2). \tag{11}$$

Equation (11) provides us a way to obtain the posterior on the parameter $\gamma_\star$ using Bayes update. In order to make this equation concrete we need to see how Rev_t depends on

[2] We define the revenue of a basket as the sum of revenues of all items.

$\gamma_\star$. Let,

$$\boldsymbol{\theta}_{ti} = \frac{p_i^2 f_{i,t}}{p_{i,t-1}} - p_i f_{i,t}. \tag{12}$$

$$\bar{R}_t = p_i f_{i,t}. \tag{13}$$

It is then easy to see that $Rev_t = \gamma_\star^\top \boldsymbol{\theta}_t + \bar{R}_t$.

$$\begin{aligned} N(R_t; Rev_t, \sigma^2) &= N(R_t; \gamma_\star^\top \boldsymbol{\theta}_t + \bar{R}_t, \sigma^2) \\ &\propto \exp\left(-\frac{1}{\sigma^2}(R_t - \bar{R}_t - \gamma_\star^\top \boldsymbol{\theta}_t)^2\right) \\ &\propto \exp\left(-(\gamma_\star - \beta_t)^\top \boldsymbol{M}_t^{-1}(\gamma_\star - \beta_t)\right). \end{aligned}$$

where,

$$\boldsymbol{M}_t^{-1}\boldsymbol{\beta}_t = \frac{R_t - \bar{R}_t}{\sigma^2}\boldsymbol{\theta}_t.$$

$$\boldsymbol{M}_t^{-1} = \frac{1}{\sigma^2}\boldsymbol{\theta}_t\boldsymbol{\theta}_t^\top + \lambda I.$$

$$\Pi_t(\gamma_\star) \propto N(\gamma_\star; \boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1}) N(\boldsymbol{\gamma}; \boldsymbol{\beta}_t, \boldsymbol{M}_t) \tag{14}$$

$$\propto N(\gamma_\star; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t), \tag{15}$$

where, $\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t$ are given by the following expressions

$$\boldsymbol{\mu}_t = (\boldsymbol{\Sigma}_{t-1}^{-1} + \boldsymbol{M}_t^{-1})^{-1}\left(\boldsymbol{\Sigma}_{t-1}^{-1}\boldsymbol{\mu}_{t-1} + \frac{R_t - \bar{R}_t}{\sigma^2}\boldsymbol{\theta}_t\right). \tag{16}$$

$$\boldsymbol{\Sigma}_t = (\boldsymbol{\Sigma}_{t-1}^{-1} + \boldsymbol{M}_t^{-1})^{-1}. \tag{17}$$

The expression for $\boldsymbol{M}_t$ is calculated as follows

$$\boldsymbol{M}_t^{-1} = \frac{\boldsymbol{\theta}_t\boldsymbol{\theta}_t^\top}{\sigma^2} + \lambda I. \tag{18}$$

$$\boldsymbol{M}_t^{-1}\beta_t = \frac{(R_t - \bar{R}_t)\theta_t}{\sigma^2}. \tag{19}$$

$\lambda > 0$ is a small positive constant that has been added to the matrix $\boldsymbol{M}_t$ to make it invertible. Equation (8) puts a Gaussian prior on the elasticity vector $\gamma_\star$ and equation (9) puts a Gaussian likelihood on the observed revenue. One might, argue that since elasticity parameter is negative, and revenue is always positive a Gaussian prior and Gaussian likelihood are incorrect models. While this criticism is justified, we argue that using this combination of prior and likelihood leads to very simple posterior updates. Moreover, in our TS implementation, we perform rejection sampling on the elasticity parameter. We reject elasticity samples that are positive, which alleviates the problem of the prior having non-zero mass on elasticity values that are positive. One might put more appropriate distributions on both elasticity and revenue which reflect

the fact that these parameters take only a restricted set of values (e.g. a log-normal distribution might be more appropriate). However, with such distributions it is generally hard to obtain closed form updates and one has to resort to approximate sampling techniques Kawale et al. (2015); Gopalan et al. (2014); Osband & Van Roy (2015); Blei et al. (2017). Use of such techniques for scalable, active dynamic pricing will be investigated in future work. The pseudo-code for MAX-REV-TS is shown in algorithm (3).

Algorithm 3: MAX-REV-TS - ACTIVE LEARNING BASED DYNAMIC PRICING ALGORITHM

Input: A basket, $\mathcal{B}$, containing B items, and a time period T over which revenue needs to be maximized.

Oracle chooses a vector $\gamma_\star \preceq 0$, but does not reveal it to the player;
Initialize the prior $\Pi_0(\gamma_\star)$;
for $t \leftarrow 1$ ***to*** T **do**
    Keep sampling from $\boldsymbol{\gamma}_t \sim \Pi_{t-1}$ until all the components of $\boldsymbol{\gamma}_t$ are negative;
    Use the demand forecaster to obtain demand forecasts $f_{i,t}$ for all $i \in \mathcal{B}$;
    Solve the MAX-REV optimization problem shown in Equation (7), with $f_{i,t}, p_{i,t-1}$, and $\boldsymbol{\gamma}_t$ as an estimate for $\gamma_\star$, to get price vector $\mathbf{p}_t$;
    Apply the prices $\mathbf{p}_t$ to obtain reward R_t ;
    Perform the updates in (16) - (17) to get updated distribution $\Pi_t(\gamma_\star)$.
end

4.2.1 Choice of prior and scalability of updates

The update equations used by MAX-REV-TS algorithm needs a prior distribution, and an estimate for the noise variance σ^2. It is common to have prior estimates for elasticities using historical data. These prior estimates can be used as the mean $\boldsymbol{\mu}_0$ of the prior distribution. We set $\Sigma_0 = cI$, for some appropriate constant c. c should be large enough to perform exploration, but not too large enough to wash out values in $\boldsymbol{\mu}_0$. An estimate of σ can be obtained using historical data to calculate the sample standard deviation of the observed revenue of the basket. Finally, by keeping only a diagonal approximation of the covariance matrix and using Sherman-Morrison-Woodbury identity we can get away with $O(|\mathcal{B}|)$ runtime and storage complexity.

5 Experiments

We shall now demonstrate the performance of MAX-REV-TS and MAX-REV-PASSIVE on both synthetic and real-world datasets.

5.1 Results on a synthetic dataset

We generated synthetic datasets for testing the performance of MAX-REV-TS and MAX-REV-PASSIVE . In order to do that, we create a basket of 100 items, each

with elasticity in the interval $[-3, -1]$. This interval was chosen because in our real-world experiments we noticed that most items have elasticities in this range. At time t for each item, we generate a demand forecast for the item $f_{i,t}$ using an auto regressive process on the demands previously observed, and use the exponential formula $d_{i,t}(p_{i,t}) = \max(f_{i,t}\left(\frac{p_{i,t}}{p_{i,t-1}}\right)^{\gamma_{\star,i}} + \epsilon_t, 0)$ to generate the demand at the price $p_{i,t}$. All prices are constrained to belong to the set $\mathcal{C}_t = [10, 20]$. This is repeated for $T = 100$ rounds for both MAX-REV-TS and MAX-REV-PASSIVE . Notice that since MAX-REV-TS and MAX-REV-PASSIVE have different behaviours, the prices that they generate could be different and hence the datasets that they generate can be very different. Nevertheless, both the algorithms generate datasets using the same formula for the demand forecast, demand and elasticities, and have the same values for $f_{i,1}, p_{i,0}$. The data generating mechanism used by MAX-REV-TS and MAX-REV-PASSIVE is shown in Algorithm (4) We used the above data generating mechanism to generate synthetic

Algorithm 4: Data generating mechanism for synthetic experiments

Input: A basket, $\mathcal{B}$, containing $B = 50$ items, a small constant $c_0 > 0$, and an algorithm $\mathcal{A}$ (such as MAX-REV-TS or MAX-REV-PASSIVE)

Choose a vector $\gamma_\star \in [-3, -1]^{100}$, but do not reveal it to the algorithm.;

Set $\beta = 0.5$, $p_{i,0} = 12$ for all $i \in \mathcal{B}$, $f_{i,1} \in [0.5, 5]$ for each item.;

for $t \leftarrow 1$ ***to*** T **do**

  Calculate the demand forecast for each item i using the formula $f_{i,t} = c_0 + \sum_{\tau=t-1}^{0} \beta^{t-\tau} d_{i,\tau} + \epsilon_t$;

  Use $f_{i,t}, \gamma_{\star,i}$, and the price $p_{i,t}$ generated by algorithm $\mathcal{A}$, and the exponential formula $d_{i,t}(p_{i,t}) = \max\left(f_{i,t}\left(\frac{p_{i,t}}{p_{i,t-1}}\right)^{\gamma_{\star,i}} + \epsilon_t, 0\right)$ to generate the demand $d_i(p_{i,t})$ of the item i;

  Register $(f_{i,t}, p_{i,t-1}, d_{i,t})$ as a data point generated by the algorithm $\mathcal{A}$.

end

datasets for both MAX-REV-TS and MAX-REV-PASSIVE . The random vectors $\gamma_\star$ and $f_{i,1}$ are generated once and fixed for both the algorithms. This way we can guarantee that the starting point for both the algorithms is the same. The random noise added to demand forecast estimation and the demand are sampled independently from $N(0, 1)$ distribution. We are interested in calculating the total revenue of all items in the basket on each day. We ran 10 independent trials for each algorithm and report the average of the total revenue for both the algorithms in Figure (1). As one can see from this figure, MAX-REV-TS continues to learn and obtains increasing revenue as time progresses. In contrast, MAX-REV-PASSIVE learns very slowly and the average revenue stays almost the same with time.

5.2 Results on a real-world dataset

We now show our results on a real world e-commerce dataset. While applying the MAX-REV-TS algorithm in the case of synthetic experiments as shown in Section (5.1)

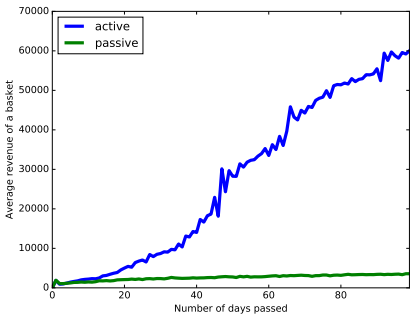

Figure 1: Plot of the average revenue of a basket (averaged over 10 trials) w.r.t. time for both MAX-REV-TS and MAX-REV-PASSIVE algorithms. MAX-REV-TS improves its estimates and better optimizes for the revenue of the basket when compared to MAX-REV-PASSIVE

is fairly straightforward, in a real world experiment with real e-commerce transactions there are many more complications that we need to handle carefully. We would like to discuss a few such issues that we encountered during our implementation.

1. In our discourse we considered the basket to be static. In reality basket changes from time to time. This is because items get added or removed from basket by merchants as per business needs.

2. On certain items, due to business constraints, the prices get fixed by business. For example, consumables such as diapers, cereal are high visibility items and their prices are almost always fixed. Items on which a merchant runs promotions also have their prices fixed. Items can go out-of-stock, unpublished from the site, or may be retired. We mark all such items and do not apply TS on those items.

3. Both MAX-REV-TS and MAX-REV-PASSIVE rely on the constant elasticity model to estimate the demand function. This model needs $f_{i,t}$ at each point of time. The demand forecaster estimates this quantity. For items that have not had sales in the recent past, the demand forecaster can predict $f_{i,t} = 0$. If $f_{i,t} = 0$, then our demand function becomes a 0 function, i.e. $d_{i,t}(p) = 0$, which makes the revenue function a constant and "shorts" our MAX-REV optimization problem and we lose information when $f_{i,t} = 0$. In general, pricing is much more harder when $f_{i,t} \approx 0$. For this reason, we do not apply TS on items with a very low demand forecast.

4. Feedback in a real-world e-commerce system is delayed. This is because when we apply new prices on items, one needs to wait for at least one day to see how the market responded to new prices, and what the revenue was at the new prices.

5.3 Experimental setup

We experiment on two separate, unrelated baskets. The first basket $\mathcal{B}_1$ has roughly 19000 items and the second basket, $\mathcal{B}_2$ has roughly 5000 items. Each of these baskets have well defined parameters such as margin goals that need to be achieved on the basket, upper and lower bounds on the price of the items. Each basket $\mathcal{B}$ can be partitioned into two disjoint sets, namely $\mathcal{B}_f, \mathcal{B}_{\text{MAX-REV}}$. The part $\mathcal{B}_f$ consists of items whose prices are already pre-fixed by various business constraints and hence they are not part of the MAX-REV optimization problem. $\mathcal{B}_{\text{MAX-REV}}$ is the subset of the basket on which we run the MAX-REV solver that generates a new set of prices. This subset consists of items that are published in-stock, with no active promotions. The set of items in $\mathcal{B}_{\text{MAX-REV}}$ can be further broken down into two disjoint part, namely $\mathcal{B}_p, \mathcal{B}_{TS}$. $\mathcal{B}_p \subset \mathcal{B}_{\text{MAX-REV}}$ is the set of items which are not eligible for TS, because of low demand forecast. We use a demand forecast threshold of 2. For items in $\mathcal{B}_p$ we use passive elasticity estimation methods such as OLS/RLS to calculate their elasticities. We apply the MAX-REV-TS algorithm on the set $\mathcal{B}_{TS}$. A caveat of our approach is that the MAX-REV optimization problem solved by MAX-REV-TS includes not just items in the partition $\mathcal{B}_{TS}$ but also items in the partition $\mathcal{B}_p$. Hence, our implementation is only an approximate MAX-REV-TS algorithm. One may avoid such a partition based approach and instead apply TS on all items in $\mathcal{B}_{\text{MAX-REV}}$. However, in our experience such implementations have very poor performance as they include items with low demand forecasts, for which our constant elasticity demand models are inaccurate. We ran MAX-REV-TS for about five weeks. On every third day, TS updates were performed to get an updated posterior distribution over their elasticities. The number of items that are on TS, on a particular day, in baskets $\mathcal{B}_1, \mathcal{B}_2$ combined is shown by the green bar in Figure (2). By design, TS encourages exploration of elasticity values for different items. This exploration in elasticity space also forces exploration of the price-demand curve of an item. Hence, we expect a good number of price changes among items that are put on TS. The blue bar in Figure (2) tells us among the items that were put on TS, for how many items did we observe a price change, compared to their prices on the previous day. As we can see from this figure, that on roughly $20\% - 30\%$ of the items there were price changes.

5.4 Revenue contributions before the start of TS and after TS was applied.

In Figure (3) we look at the revenue obtained by items that were put on TS each day. For items put on TS, each day, we also calculate what was the average revenue (and the standard deviation) corresponding to these items before the start of TS experiment. This average was calculated over a span of 30 days before the start of TS period. Since, MAX-REV-PASSIVE was applied before the start of TS period, this results provide a comparison of MAX-REV-TS with MAX-REV-PASSIVE . On basket $\mathcal{B}_1$, from a revenue point of view initially TS does not seem to help as it gets smaller revenue as compared to not applying TS. This is understandable, since initially the variance in our estimates of elasticity is large. However, about half-way through the experimental period the revenue contribution due to items on TS improves and is competent with the

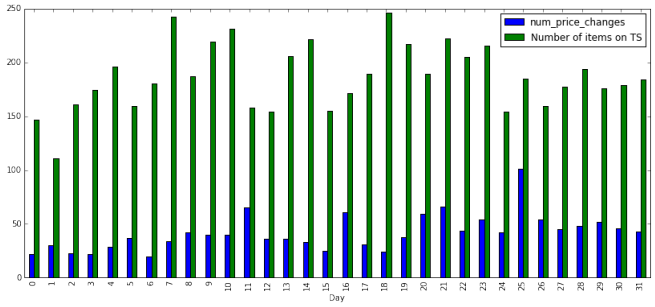

Figure 2: The total number of items in baskets $\mathcal{B}_1, \mathcal{B}_2$ which are on TS each day and the number of items whose price changed compared to the previous day.

average revenue before the start of TS period. On basket $\mathcal{B}_2$, the results suggest that TS consistently outperforms the revenue obtained before the start of TS period.

5.5 Statistical significance tests.

Figure (3) compares the revenue of items that were put on TS each day, with the average revenue of these same set of items before the start of TS. We now present quantitative results that looks at all items that were on TS at least some number (say k) of times. For these items we calculate δ_i, which is the difference between the average revenue of item i on days when it was on TS and the average revenue for a period of 30 days before the start of TS. For each value of k, let S_k be the number of items that were on TS at least k times during the TS period.

For each k and for each basket we perform the following hypothesis test. H_0 : $\boldsymbol{E}[\delta_i] = 0$, H_1 : $\boldsymbol{E}[\delta_i] \neq 0$. We use Wald's test Wasserman (2013) to perform this hypothesis test and report significance at level $\alpha = 0.05$.

Table (1) suggests that for basket $\mathcal{B}_1$, barring $k = 30$, all the p-values of the Wald test are less than $\alpha = 0.05$. This means that except for $k = 30$ the null hypothesis can be rejected. Moreover the last column of the table shows that $\boldsymbol{E}[\delta_i] < 0$, which means that there is a statistically significant degradation in the revenue per item, for basket $\mathcal{B}_1$, when the revenue is calculated only over those days where TS was applied. In contrast to this negative result, Table (2) suggests that barring the case of $k = 1$, there is a statistically significant increase in revenue per item, for items in basket $\mathcal{B}_2$, due to the application of TS. In the above hypothesis testing experiments we reported results of the revenue difference per item when calculated only over the days when the items were eligible for TS. We now calculate a different metric which is similar in spirit to the one calculated in tables (1), (2), where instead of calculating the revenue of an item averaged only over days when the item was on TS, we now average the revenue of the item during the entire TS period. For each item i we calculate δ_i which is the difference in the average revenue of item i after the start of TS, and the average revenue of the item in the 30 day window before the start of TS. The null hypothesis for our test is that H_0 : $\boldsymbol{E}\delta_i = 0$ and the alternate hypothesis is H_1 : $\boldsymbol{E}\delta_i \neq 0$. We

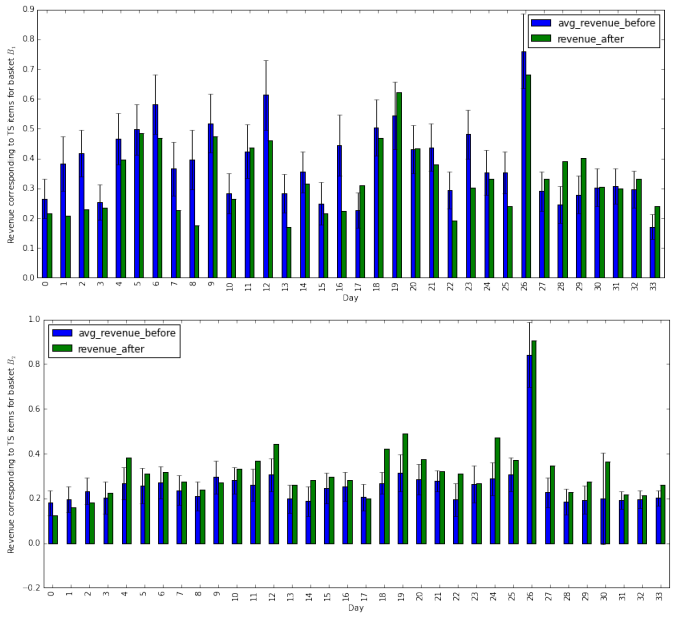

Figure 3: The green bar corresponds to the revenue contributed by items that were eligible for TS on that day. The blue bar corresponds to the revenue of these items, averaged over a period of 30 days before the start of the TS period. We also have shown the standard deviation of the revenue before the start of TS period. All quantities in this plot have been normalized.

k	S_k	p-value	mean of δ_i
5	82	0.039	-14.19
10	55	0.016	-13.35
15	26	0.061	-17.67
20	20	0.027	-25.04
25	15	0.023	-33.08
30	8	0.29	-21.58

Table 1: Wald's test results for basket $\mathcal{B}_1$ comparing the difference in revenues due to TS. Let δ_i be the difference between the average revenue of item i on the days this item was put on TS, and the average revenue of this item over a period of 30 days before the start of TS. k in the table filters out items which have been on TS for less than k days. The last column is the sample average of δ_i.

k	S_k	p-value	mean of δ_i
5	371	$2.37x10^{-6}$	1.92
10	197	$1.89x10^{-7}$	2.08
15	128	$2.13x10^{-6}$	2.68
20	91	0.000628	1.966
25	67	0.00109	2.21
30	44	0.0019	3.062

Table 2: Wald's test results for basket $\mathcal{B}_2$ with the same setup as in Table (1).

k	S_k	p-value	mean of δ_i
5	82	0.129	-9.65
10	55	0.328	-4.28
15	26	0.424	-5.45
20	20	0.25	-10.19
25	15	0.19	-14.27
30	8	0.52	-12.76

Table 3: Wald's test results for basket $\mathcal{B}_1$ comparing the difference in revenues due to TS. Let δ_i be the difference between the average revenue of item i during the TS period (irrespective of whether i was on TS or not on a certain day), and the average revenue of this item over a period of 30 days before the start of TS. k in the table filters out items which have been on TS for less than k days. The last column is the sample average of δ_i.

use Wald's test as our testing procedure and in tables (3), (4) report the p-value of the test and also the sample average of δ_i calculated over all qualifying items. As one can see from table (3), for basket $\mathcal{B}_1$ there is a degradation in the revenue per item, though not statistically significant. In contrast, for basket $\mathcal{B}_2$, as seen from Table (4), there is a statistically significant increase in the average revenue per item after the application of TS.

5.6 Discussion of our experimental results

An interesting observation from our experimental results is that while on basket $\mathcal{B}_1$ there is a decrease in revenue per item that is put on TS, this result is either not statistically significant at $\alpha = 0.05$ or its p-value is large. In contrast, for basket $\mathcal{B}_2$, the increase in revenue per-item is large and statistically significant, and the p-values are

k	S_k	p-value	mean of δ_i
5	371	$2.28x10^{-8}$	1.582
10	197	$3.13x10^{-7}$	1.649
15	128	$9.64x10^{-6}$	2.05
20	91	0.00099	1.64
25	67	0.00084	1.99
30	44	0.0013	2.74

Table 4: Wald's test results for basket $\mathcal{B}_2$ with the same setup as in Table (3).

Basket	$\lvert\mathcal{B}\rvert$	$\lvert\mathcal{B}_f\rvert$	$\lvert\mathcal{B}_p\rvert$	$\lvert\mathcal{B}_{TS}\rvert$	$\frac{\lvert\mathcal{B}_{TS}\rvert}{\lvert\mathcal{B}_{\text{MAX-REV}}\rvert}$
$\mathcal{B}_1$	19K	13K	6K	25	0.004
$\mathcal{B}_2$	5K	3.3K	1.5K	150	0.1

Table 5: Approximate size of the problem for different baskets. $|\mathcal{B}|$ is the size of the basket, $|\mathcal{B}_f|$ is the number of items whose prices are pre-fixed by business constraints, $|\mathcal{B}_p|$ is the number of items on which the MAX-REV optimization problem is applied but whose elasticities are generated using passive algorithms, and $|\mathcal{B}_{TS}|$ is the number of items on which the MAX-REV optimization problem is applied but whose elasticities are generated using TS.

small. One may ask, why is there a difference in performance of the MAX-REV-TS algorithm on these baskets? We present two reasons for this. As mentioned in subsection (5.3) our implementation applies TS only to a subset of the basket, $\mathcal{B}_{TS}$, and the MAX-REV-TS algorithm solves the MAX-REV problem over a larger subset $\mathcal{B}_{\text{MAX-REV}}$. Hence, our implementation of the MAX-REV-TS algorithm is only an approximation implementation. If $|\mathcal{B}_{TS}| << |\mathcal{B}_{\text{MAX-REV}}|$, then our implementation can be very crude and the TS updates will be very noisy. Hence, as the ratio $\frac{|\mathcal{B}_{TS}|}{|\mathcal{B}_{\text{MAX-REV}}|}$ decreases the updates of TS will become less accurate. This ratio can be seen as the singal-to-noise ratio of the problem. Secondly, even if TS updates were accurate enough, the MAX-REV solver, being optimized over $\mathcal{B}_{\text{MAX-REV}}$, might trade-off the revenue of $\mathcal{B}_p$ against $\mathcal{B}_{TS}$, if it lead to a larger revenue for $\mathcal{B}_{\text{MAX-REV}}$ under the provided constraints. Hence, it could be the case that even though we observe a degradation in revenue over the subset $\mathcal{B}_{TS}$, the overall revenue of items $\mathcal{B}_{\text{MAX-REV}}$ might not suffer. Table (5) shows the various sizes for baskets $\mathcal{B}_1, \mathcal{B}_2$. The ratio $\frac{|\mathcal{B}_{TS}|}{|\mathcal{B}_{\text{MAX-REV}}|}$ is 0.004 for basket $\mathcal{B}_1$, whereas it is 0.1 for basket $\mathcal{B}_2$, which explains the poor performance of TS on basket $\mathcal{B}_1$.

6 Conclusions and future work

In this paper we designed and implemented dynamic pricing algorithms that maximize a reward function under uncertainty. To the best of our knowledge this is the first real world study of active dynamic pricing algorithms in an e-commerce setting. We demonstrated the performance of our algorithms comparing against passive dynamic pricing algorithms that are most commonly used in practice. We see a statistically significant improvement in the per-item revenue when the number of items on which we apply TS is a large fraction of the basket of items. This ratio can be thought of as the SNR of the problem. On a basket, where TS led to a degradation of per-item revenue, we posit that this could be because the SNR was very low. Our future work includes investigations of various other reward functions such as volumes of sales, and experimental study on more baskets where the SNR ratio is large enough to enable reliable measurement of the performance of active dynamic pricing algorithms.

References

Agarwal, Alekh, Bird, Sarah, Cozowicz, Markus, Hoang, Luong, Langford, John, Lee, Stephen, Li, Jiaji, Melamed, Dan, Oshri, Gal, Ribas, Oswaldo, et al. A multiworld testing decision service. *arXiv preprint arXiv:1606.03966*, 2016.

Aghion, Philippe, Bolton, Patrick, Harris, Christopher, and Jullien, Bruno. Optimal learning by experimentation. *The review of economic studies*, 58(4):621–654, 1991.

Araman, Victor F and Caldentey, René. Dynamic pricing for nonperishable products with demand learning. *Operations Research*, 57(5):1169–1188, 2009.

Auer, Peter, Cesa-Bianchi, Nicolo, and Fischer, Paul. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002a.

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

Aviv, Yossi and Pazgal, Amit. A partially observed markov decision process for dynamic pricing. *Management Science*, 51(9):1400–1416, 2005.

Ben-Tal, Aharon, El Ghaoui, Laurent, and Nemirovski, Arkadi. *Robust optimization*. Princeton University Press, 2009.

Bertsekas, Dimitri P and Scientific, Athena. *Convex optimization algorithms*. Athena Scientific Belmont, 2015.

Besbes, Omar and Zeevi, Assaf. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57(6):1407–1420, 2009.

Blei, David M, Kucukelbir, Alp, and McAuliffe, Jon D. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, (just-accepted), 2017.

Broder, Josef and Rusmevichientong, Paat. Dynamic pricing under a general parametric choice model. *Operations Research*, 60(4):965–980, 2012.

Bubeck, Sébastien, Cesa-Bianchi, Nicolo, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

Caruana, Rich. Multitask learning. In *Learning to learn*, pp. 95–133. Springer, 1998.

Carvalho, Alexandre X and Puterman, Martin L. Learning and pricing in an internet environment with binomial demands. *Journal of Revenue and Pricing Management*, 3(4):320–336, 2005a.

Carvalho, Alexandre X and Puterman, Martin L. Dynamic optimization and learning: How should a manager set prices when the demand function is unknown? 2005b.

den Boer, Arnoud V. Dynamic pricing and learning: historical origins, current research, and new directions. *Surveys in operations research and management science*, 20(1): 1–18, 2015.

den Boer, Arnoud V and Zwart, Bert. Simultaneously learning and optimizing using controlled variance pricing. *Management science*, 60(3):770–783, 2013.

Farias, Vivek F and Van Roy, Benjamin. Dynamic pricing with a prior on market response. *Operations Research*, 58(1):16–29, 2010.

Ferreira, Kris Johnson, Simchi-Levi, David, and Wang, He. Online network revenue management using thompson sampling. *Harvard Business School Technology & Operations Mgt. Unit Working Paper. Available at SSRN: https://ssrn.com/abstract=2588730*, 2016.

Gopalan, Aditya, Mannor, Shie, and Mansour, Yishay. Thompson sampling for complex online problems. In *International Conference on Machine Learning*, pp. 100–108, 2014.

Harrison, J Michael, Keskin, N Bora, and Zeevi, Assaf. Bayesian dynamic pricing policies: Learning and earning under a binary prior distribution. *Management Science*, 58(3):570–586, 2012.

Kawale, Jaya, Bui, Hung H, Kveton, Branislav, Tran-Thanh, Long, and Chawla, Sanjay. Efficient thompson sampling for online matrix-factorization recommendation. In *Advances in Neural Information Processing Systems*, pp. 1297–1305, 2015.

Keskin, N Bora and Zeevi, Assaf. Dynamic pricing with an unknown demand model: Asymptotically optimal semi-myopic policies. *Operations Research*, 62(5):1142–1167, 2014.

Osband, Ian and Van Roy, Benjamin. Bootstrapped thompson sampling and deep exploration. *arXiv preprint arXiv:1507.00300*, 2015.

Phillips, Robert Lewis. *Pricing and revenue optimization*. Stanford University Press, 2005.

Rothschild, Michael. A two-armed bandit theory of market pricing. *Journal of Economic Theory*, 9(2):185–202, 1974.

Russo, Daniel, Van Roy, Benjamin, Kazerouni, Abbas, and Osband, Ian. A tutorial on thompson sampling. *arXiv preprint arXiv:1707.02038*, 2017.

Shah, Parikshit, Yang, Ming, Alle, Sachidanand, Ratnaparkhi, Adwait, Shahshahani, Ben, and Chandra, Rohit. A practical exploration system for search advertising. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1625–1631. ACM, 2017.

Talluri, Kalyan T and Van Ryzin, Garrett J. *The theory and practice of revenue management*, volume 68. Springer Science & Business Media, 2006.

Thompson, William R. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

Wasserman, Larry. *All of statistics: a concise course in statistical inference*. Springer Science & Business Media, 2013.