# Learning with Abandonment

**Sven Schmit** [1] **Ramesh Johari** [2]

## Abstract

Consider a platform that wants to learn a personalized policy for each user, but the platform faces the risk of a user abandoning the platform if they are dissatisfied with the actions of the platform. For example, a platform is interested in personalizing the number of newsletters it sends, but faces the risk that the user unsubscribes forever. We propose a general thresholded learning model for scenarios like this, and discuss the structure of optimal policies. We describe salient features of optimal personalization algorithms and how feedback the platform receives impacts the results. Furthermore, we investigate how the platform can efficiently learn the heterogeneity across users by interacting with a population and provide performance guarantees.

## 1. Introduction

Machine learning algorithms are increasingly intermediating interactions between platforms and their users. As a result, users' interaction with the algorithms will impact optimal learning strategies; we investigate this consequence in our work. In the setting we consider, a platform wants to personalize service to each user. The distinctive feature in this work is that the platform faces the risk of a user abandoning the platform if they are dissatisfied with the actions of the platform. Algorithms designed by the platform thus need to be careful to avoid losing users.

There are many examples of such settings. In the near future, smart energy meters will be able to throttle consumers' energy consumption to increase efficiency of the power grid during peak demand, e.g., by raising or lowering the level of air conditioning. This can lead to cost savings for both utility companies and consumers. However, if the utility company is too aggressive in its throttling of energy, a user might abandon the program. Due to heterogeneity in housing, appliances and preferences of customers, it is important that utility companies learn personalized strategies for each consumer.

Content creators (e.g., news sites, blogs, etc.) face a similar problem with e-mail dissemination. There is value in sending more e-mails, but each e-mail also risks the recipient unsubscribing, taking away any opportunity of the creator to interact with the user in the future. Yet another example is that of mobile app notifications. These can be used to improve user engagement and experience. However if the platform sends too many notifications, an upset user might turn off notifications from the application.

In all of the above scenarios, we face a decision problem where "more is better;" however, there is a threshold beyond which the user abandons and no further rewards are gained. This work focuses on developing insight into the structure of optimal learning strategies in such settings. We are particularly interested in understanding when such strategies take on a "simple" structure, as we elaborate below.

In Section 2, we introduce a benchmark model of learning with abandonment. In the initial model we consider, a platform interacts with a single user over time. The user has a *threshold* $\theta$ drawn from a distribution $F$, and at each time $t = 0, 1, 2, \ldots$ the platform chooses an action $x_t$. If $x_t$ ever exceeds $\theta$, the user abandons; otherwise, the user stays, and the platform earns some reward dependent on $x_t$.

We first consider the case where the distribution $F$ and the reward function are known (say, from prior estimation),[1] and the challenge is finding an optimal strategy for a given new user. We consider the problem of maximizing expected discounted reward. Intuitively, we might expect that the optimal policy is increasing and depends on the discount factor: in particular, we might try to serve the user at increasing levels of $x_t$ as long as we see they did not abandon. Surprisingly, our main result shows this is not the case: that in fact, the *static* policy of maximizing one-step reward is optimal for this problem. Essentially, because the user abandons if the threshold is ever crossed, there is no

[1] Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA, USA [2] Management Science & Engineering, Stanford University, Stanford, CA, USA.. Correspondence to: Sven Schmit <schmit@stanford.edu>.

*Proceedings of the $35^{th}$ International Conference on Machine Learning*, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

[1] We also use F to denote the CDF.

value to trying to actively learn the threshold.

In Section 3, we consider how to adapt our results when $F$ and/or the reward function are unknown. In this case, the platform can learn over multiple user arrivals. We relate the problem to one of learning an unknown demand curve, and suggest an approach to efficiently learning the threshold distribution $F$ and the reward function.

Finally in Section 4, we consider a more general model with "soft" abandonment: after a negative experience, users may not abandon entirely, but continue with the platform with some probability. We characterize the structure of an optimal policy to maximize expected discounted reward on a per-user basis; in particular, we find that the policy adaptively experiments until it has sufficient confidence, and then commits to a static action. We empirically investigate the structure of the optimal policy as well.

**Related work** The abandonment setting we consider is the unique and novel feature of this work. Independently from this work, Lu et al. (2017) model the abandonment problem using only two actions; the safe action and the risky action. This naturally leads to rather different results. There are some similarities with the mechanism design literature, though there the focus is on strategic behavior by agents (Rothschild, 1974; Myerson, 1981; Farias & Van Roy, 2010; Pavan et al., 2014; Lobel & Paes Leme, 2017). As in this work, the revenue management literature considers agents with heuristic behaviour, but the main focus is on dealing with a finite inventory (Gallego & Van Ryzin, 1994).

It may seem that our problem is closely related to many problems in reinforcement learning (Sutton & Barto, 1998) due to the dynamic structure of our problem. However, there are important differences. Our focus is on personalization; viewed through the lens of reinforcement learning, this corresponds to having only a single episode to learn, which is independent of other episodes (users). On the other hand, in reinforcement learning the focus is on learning an optimal policy using multiple episodes where information carries over between episodes. These differences present novel challenges in the abandonment setting, and necessitate use of the structure present in this setting.

Also related is work on safe reinforcement learning, where catastrophic states need to be completely avoided (Moldovan & Abbeel, 2012; Berkenkamp et al., 2017). In such a setting, unlike in ours, the learner usually has access to additional information, for example a safe region is given. In this work, there is no hard constraint on avoiding catastrophic states, they simply lead to less reward.

## 2. Threshold Model

In this section, we formalize the problem of finding a personalized policy for a single user without additional feedback.

### 2.1. Formal setup and notation

We consider a setting where heterogeneous users interact with a platform at discrete time steps indexed by $t$, and focus on the problem of finding a personalized policy for a single user. The user is characterized by sequence of hidden thresholds $\{\theta_t\}_{t=0}^{\infty}$ jointly drawn from a known distribution that models the heterogeneity across users. At every time $t$, the platform selects an action $x_t \in \mathbf{X} \subset \mathbb{R}_+$ from a given closed set $\mathbf{X}$. Based on the chosen action $x_t$, the platform obtains the random reward $R_t(x_t) \geq 0$. The expected reward of action $x$ is given by $r(x) = \mathbb{E}(R_t(x)) < B$, which we assume to be stationary, bounded, and known to the platform.[2] While not required for our results, we expect $r$ to be increasing. When the action exceeds the threshold at time $t$, the process stops. More formally, let $T$ be the stopping time that denotes the first time the $x_t$ exceeds the threshold $\theta_t$:

$$T = \min\{t : x_t > \theta_t\}. \tag{1}$$

The goal is to find a sequence of actions $\{x_t\}_{t=0}^{\infty}$ that maximizes:

$$\mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t R_t(x_t)\right], \tag{2}$$

where $\gamma \in (0,1)$ denotes the discount factor. We note that this expectation is well defined even if $T = \infty$, since $\gamma < 1$. We focus here on the discounted expected reward criterion. An alternative approach is to consider maximizing the average reward on a finite horizon; considering this problem remains an interesting direction for future work.

### 2.2. Optimal policies

Without imposing further restrictions on the structure of the stochastic threshold process, the solution is intractable. Thus, we first consider two extreme cases: (1) the threshold is sampled at the start and then remains fixed across time; and (2) the thresholds are independent across time. Thereafter, we look at the robustness of the results when we deviate from these extreme scenarios.

**Fixed threshold** We first consider a case where the threshold is sampled at the beginning of the horizon, but then remains fixed. In other words, for all $t$, $\theta_t = \theta \sim F$. Intuitively, we might expect that the platform tries to gradually learn this threshold, by starting with small $x_t$ and in-

[2] In Section 3 we discuss the case when both $F$ and $r$ are unknown.

creasing it as long as the user does not abandon. In fact, we find something quite different: our main result is that the optimal policy is a constant policy.

**Theorem 1.** *Suppose that for all t, $\theta_t = \theta$ for $\theta \sim F$, and that the function $p : x \to r(x)(1 - F(x))$ has a unique optimum $x^* \in \mathbf{X}$. Then, the optimal policy is $x_t = x^*$ for all t.*

All proofs can be found in the supplemental material, but we sketch an argument why there exists a constant policy that is optimal. Consider a policy that is increasing and suppose it is optimal.[3] Then there exists a time $t$ such that $x_t = y < x_{t+1} = z$. Compare these two actions with the policy that would use action $z$ at both time periods. First suppose $\theta < y$; then the user abandons under either alternative and so the outcome is identical. Now consider $\theta \geq y$; then by the optimality of the first policy, given knowledge that $\theta \geq y$, it is optimal to play $z$. But that means the constant policy that selects $z$ at both time periods is at least as good as the optimal policy.

In the appendix, we provide another proof of the result using value iteration. This proof also characterizes the optimal policy and optimal value exactly (as in the proposition). Remarkably, the optimal policy is independent of the discount factor $\gamma$.

**Independent thresholds** For completeness, we also note here the other extreme case: suppose the thresholds $\theta_t$ are drawn independently from the same distribution $F$ at each $t$. Then since there is no correlation between time steps, it follows immediately that the optimal policy is a constant policy, with a simple form.

**Proposition 2.** *If*

$$x^* \in \arg\max_{x \in \mathbf{X}} \frac{r(x)(1 - F(x))}{1 - \gamma(1 - F(x))} \tag{3}$$

*is the unique optimum, then the optimal policy under the independent threshold assumption is $x_t = x^*$ for all t.*

**Robustness** So far, we have considered two extreme threshold models and have shown that constant policies, albeit different ones, are optimal. In this section we look at the robustness of those results by understanding what happens when we interpolate between the two sides by considering an additive noise threshold model. Here, the threshold at time $t$ consists of a fixed term and a noisy term: $\theta_t = \theta + \varepsilon_t$, where $\theta \sim F$ is drawn once, and the noise terms are drawn at every time $t$, independent of $\theta$. In general, the optimal policy in this model is increasing and intractable because the posterior over $\theta$ now depends on all previous actions. However, there exists constant policies that are close to optimal in case the noise terms are either small or large, reflecting our preceding results in the extreme cases.

First consider the case where the noise terms are *small*. In particular, suppose the error distribution has an arbitrary distribution over a small interval $[-y, y]$. Note in particular that we do not assume the noise is independent across time.

**Proposition 3.** *Suppose $\varepsilon_t \in [-y, y]$ and the reward function r is L-Lipschitz. Then there exists a constant policy with value $V_c$ such that*

$$V^* - V_c \leq \frac{2yL}{1 - \gamma} \tag{4}$$

*where $V^*$ is the value of the optimal policy for the noise model.*

This result follows from comparing the most beneficial and detrimental scenarios; $\varepsilon_t = y$ and $\varepsilon_t = -y$ for all $t$, respectively, and noting that in both cases the optimal policies are constant policies, because thresholds are simply shifted. We can then show that the optimal policy for the worst scenario achieves the gap above compared to the optimal policy in the best case. The details can be found in the appendix.

Similarly, when the noise level is sufficiently *large* with respect to the threshold distribution $F$, and independent across time, there also exists a constant policy that is close to optimal. The intuition behind this is as follows. First, if the noise level is large, the platform receives only little information at each step, and thus cannot efficiently update the posterior on $\theta$. Furthermore, the high variance in the thresholds also reduces the expected lifetime of any policy. Combined, these two factors make learning ineffective.

We formalize this by comparing a constant policy to an oracle policy that knows $\theta$ but not the noise terms $\varepsilon_t$. Let $G$ be the CDF of the noise distribution $\varepsilon_t$ with $\bar{G}$ denoting its complement: $\bar{G}(y) = 1 - G(y)$. Then we note that for a given threshold $\theta$, the probability of survival is $\bar{G}(x - \theta)$, and thus the expected value for the constant policy $x_t = x$ for all $t$ is

$$\frac{\bar{G}(x - \theta) r(x)}{1 - \gamma \bar{G}(x - \theta)}. \tag{5}$$

Define the optimal constant policy given knowledge of the fixed part of the threshold, $\theta$ by $x(\theta)$:

$$x(\theta) = \arg\max_{x} \frac{\bar{G}(x - \theta) r(x)}{1 - \gamma \bar{G}(x - \theta)}. \tag{6}$$

We can furthermore define the value of policy $x_t = x(\theta)$ when the threshold is $\theta'$ by $v(\theta, \theta')$:

$$v(\theta, \theta') = \frac{\bar{G}(x(\theta) - \theta') r(x(\theta))}{1 - \gamma \bar{G}(x(\theta) - \theta')}. \tag{7}$$

[3] It is clear that the optimal policy cannot be decreasing.

We note that $v$ is non-decreasing in $\theta'$. We assume that $v$ is $L_v$-Lipschitz:

$$|v(\theta,\theta') - v(s,\theta')| \leq L_v|\theta - s| \tag{8}$$

for all $\theta$ and $s$. Note that noise distributions $G$ that have high variance lead to a smaller Lipschitz constant.

To state our result in this case, we define an $\eta$-cover, which is a simple notion of the spread of a distribution.

**Definition 1.** *An interval $(l, u)$ provides an $\eta$-cover for distribution $F$ if $F(u) - F(l) > \eta$.*

In other words, with probability as least $1 - \eta$, a random variable drawn from distribution $F$ lies in the interval $(l, u)$.

**Proposition 4.** *Recall that we assume $r$ is bounded by $B$, and $\mathbf{X}$ is a continuous and connected space. Suppose $v$ defined above is $L_v$-Lipschitz, and there exists an $\eta$-cover for threshold distribution $F_\theta$ with width $w = u-l$. Then the constant policy $x_t = \frac{l+u}{2}$ with expected value $V_\theta$ satisfies*

$$V^* - V_\theta \leq V_o - V_\theta \leq \frac{L_v w}{2} + 2\frac{\eta B}{1-\gamma}. \tag{9}$$

The shape of $v$, and in particular its Lipschitz constant $L_v$ depend on the threshold distribution $F$ and reward function $r$. As the noise distribution $G$ "widens", $L_v$ decreases. As a result, the bound above is most relevant when the variance of $G$ is substantial relative to spread of $F$.

To summarize, our results show that in the extreme cases where the thresholds are drawn independently, or drawn once, there exists a constant policy that is optimal. Further, the class of constant policies is robust when the joint distribution over the thresholds is close to either of these scenarios.

## 3. Learning Thresholds

Thus far, we have assumed that the heterogeneity across the population and the mean reward function are known to the platform, and we have focused on personalization for a single user. It is natural to ask what the platform should do when it lacks such knowledge, and in this section we show how the platform can learn an optimal policy efficiently across the population. We study this problem within the context of the fixed threshold model described above, as it naturally lends itself to development of algorithms that learn about population-level heterogeneity. In particular, we give theoretical performance guarantees on UCB type (Auer et al., 2002) algorithms, and compare performance to an explore-exploit benchmark.

**Learning setting** We focus our attention on the fixed threshold model, and consider a setting where $n$ users arrive sequentially, each with a fixed threshold $\theta_u$ ($u = 1, \ldots, n$) drawn from unknown distribution $F$ with support on $[0, 1]$. To emphasize the role of learning from users over time, we consider a stylized setting where the platform interacts with one user at a time, deciding on all the actions and observing the outcomes for this user, before the next user arrives. Inspired by our preceding analysis, we consider a proposed algorithm that uses a constant policy for each user. Furthermore, we assume that the rewards $R_t(x)$ are bounded between 0 and $M$, but otherwise drawn from an arbitrary distribution that depends on $x$.

**Regret with respect to an oracle** We measure the performance of learning algorithms against the oracle that has full knowledge about the threshold distribution $F$ and the reward function $r$, but no access to realizations of random variables. As discussed in Section 2, the optimal policy for the oracle is thus to play constant policy $x^* = \max_{x\in[0,1]} r(x)(1 - F(x))$. We define regret as

$$\begin{aligned}\text{regret}_n(A) &= nr(x^*)(1 - F(x^*)) \\ &\quad - (1-\gamma)\sum_{u=1}^{n}\mathbb{E}\left[\sum_{t=0}^{T_u-1}\gamma^t r(x_{u,t})\right]\end{aligned} \tag{10}$$

which we note is normalized on a per-user basis with respect to the discount factor $\gamma$.

### 3.1. UCB strategy

We propose a UCB algorithm (Auer et al., 2002) on a suitably discretized space, and prove an upper bound on its regret in terms of the number of users. This approach is based on earlier work by (Kleinberg & Leighton, 2003)[Section 3] for learning demand curves. Before presenting the details, we introduce the UCB algorithm for the standard multi-armed bandit problem (Bubeck et al., 2012).

In the standard setting, there are $K$ arms, each with its own mean $\mu_i$. At each time $t$, UCB($\alpha$) selects the arm with largest index $B_{i,t}$

$$B_{i,t} = \bar{X}_{i,n_i(t)} + \sigma\sqrt{\frac{2\alpha\log t}{n_i(t)}} \tag{11}$$

where $n_i(t)$ is the number of pulls of arm $i$ at time $t$. We assume $B_{i,t} = \infty$ if $n_i(t) = 0$. The following lemma bounds the regret of the UCB index policy.

**Lemma 5** (Theorem 2.1 (Bubeck et al., 2012))**.** *Suppose rewards for each arm $i$ are independent across multiple pulls, $\sigma$-sub-Gaussian and have mean $\mu_i$. Define $\Delta_i = \max_j \mu_j - \mu_i$. Then, UCB($\alpha$) attains regret bound*

$$\text{regret}_n(UCB) \leq \sum_{i:\Delta_i>0}\frac{8\alpha\sigma^2}{\Delta_i}\log n + \frac{\alpha}{\alpha-2}. \tag{12}$$

Kleinberg & Leighton (2003) adapt the above result to the problem of demand curve learning. We follow their approach: First discretize the action space and then use the standard UCB approach to find an approximately optimal action. For each user, the algorithm selects a constant action $x_u$ and either receives reward $R_u = 0$ if $x_u > \theta_u$ or $R_u = \sum_{t=0}^{\infty} \gamma^t R_t(x_u)$.

We need to impose the following additional assumption on the function $p(x) = r(x)(1 - F(x))$.

**Assumption 1** (Lemma 3.11 in Leighton and Kleinberg)**.** *There exists constants $c_1$ and $c_2$ such that*

$$c_1(x^* - x)^2 < p(x^*) - p(x) < c_2(x^* - x)^2 \tag{13}$$

*for all $x \in [0, 1]$.*

For example, if $p$ is strongly concave, this assumption is satisfied. Then, we prove the following learning result.

**Theorem 6.** *If $p$ satisfies the concavity condition (13), then UCB($\alpha$) on the discretized space with $K = O\left((n/\log n)^{1/4}\right)$ arms satisfies*

$$\text{regret}_n(\text{UCB}) \leq O\left(\sqrt{n \log n}\right) \tag{14}$$

*for all $\alpha > 2$.*

The proof consists of two parts, first we use Assumption 1 to bound the difference between the best action and the best arm in the discretized action space. Then we use Theorem 5 to show that the learning strategy has small regret compared to the best arm, again leveraging Assumption 1 to argue that most arms are pulled few times. Combined, these prove the result.

It is important to note that the algorithm requires prior knowledge of the number of users, $n$. In practice it is reasonable to assume that a platform is able to estimate this accurately, but otherwise the well-known doubling trick can be employed at a slight cost.

### 3.2. Lower bound

We now briefly discuss lower bounds on learning algorithms. If we restrict ourselves to algorithms that play a constant policy for each user, the lower bound by Kleinberg & Leighton (2003) applies immediately.

**Proposition 7** (Theorem 3.9 in (Kleinberg & Leighton, 2003))**.** *Any learning algorithm $A$ that plays a constant policy for each user, has regret at least*

$$\text{regret}_n(A) \geq \Omega(\sqrt{n}) \tag{15}$$

*for some threshold distribution.*

Thus, the discretized UCB strategy is near-optimal in the class of constant policies.

However, algorithms with dynamic policies for users can obtain more information on the user's threshold and therefore more easily estimate the empirical distribution function. Whether the $O(\sqrt{n})$ lower bound carries over to dynamic policies is an open problem.

### 3.3. Simulations

In this section, we empirically compare the performance of the discretized UCB against other policies. For our simulations, we also include the KL-UCB algorithm (Garivier & Cappé, 2011), and an explore-exploit strategy as a benchmark.

**KL-UCB** The KL-UCB algorithm by Garivier & Cappé (2011) is an improved variant of the standard UCB algorithm with tighter performance guarantees and better performance in practice. Using the same argument, we can show a similar regret bound for our setting.

**Proposition 8.** *If $p$ satisfies the concavity condition (13), then KL-UCB on the discretized space with $K = O\left((n/\log n)^{1/4}\right)$ arms satisfies*

$$\text{regret}_n(\text{KL-UCB}) \leq O\left(\sqrt{n \log n}\right) \tag{16}$$

**Explore-exploit strategy** Policies with dynamic actions for each user can gain more information about the threshold distribution. To understand possible gains from this additional information we consider a heuristic explore-exploit strategy that provides a benchmark for the UCB algorithms. The explore-exploit strategy first estimates an empirical distribution function, and then uses that to optimize a constant policy. For this algorithm, we assume that for zero reward, the learner can observe $\theta_u$ for a particular user, which mimics a strategy where the learner increases its action by $\varepsilon$ at each time period to learn the threshold $\theta_u$ of a particular user with arbitrary precision. Because it directly estimates the empirical distribution function and does not require discretization, it is better able to capture the structure of our model.

The explore-exploit strategy consists of two stages.

- First, obtain $m$ samples of $\theta_u$ to find an empirical estimate of $F$, which we denote by $\hat{F}_m$
- For the remaining users, play constant policy $x_u = \arg\max r(x)(1 - \hat{F}_m(x))$

Note that compared to the previous algorithm, we assume this learner has access to the reward function, and only the threshold distribution $F$ is unknown. If the signal-to-noise ratio in the stochastic rewards is large, this is not unrealistic: the platform, while exploring, is able to observe a large number of rewards and should therefore be able to estimate the reward function reasonably well.

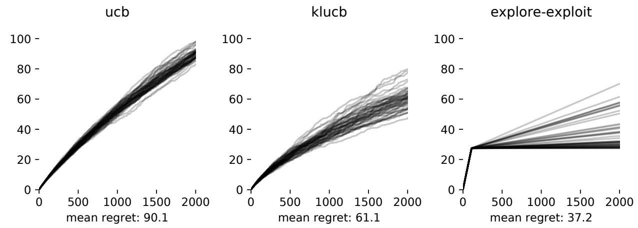

*Figure 1.* Cumulative regret plots for $r(x) = x$ and $F = U[0, 1]$.

**Setup** For simplicity, our simulations focus on a stylized setting, but we observed similar results under different scenarios.[4] We assume that the rewards are deterministic and follow the identity function $r(x) = x$, and the threshold distribution (unknown to the learning algorithm) is uniform on $[0, 1]$. For each algorithm, we run 50 repetitions for $n = 2000$ time steps, and plot all cumulative regret paths. For the discretized policies, we set $K \approx 2\left(\frac{n}{\log n}\right)^{1/4} = 10$. The explore-exploit strategy first observes $20 + 2\sqrt{n} = 110$ samples to estimate $F$, before committing to a fixed strategy.

**Results** The cumulative regret paths are shown in Figure 1. We observe that KL-UCB indeed performs better than the standard UCB algorithm.[5]

The explore-exploit strategy outperforms the UCB algorithms, which shows that better information on the distribution of $\theta_u$ and leveraging the structure of the problem can lead to gains over the more naive UCB algorithms. Thus there can be practical benefit to further algorithmic optimization.

## 4. Feedback

In this section, we consider a “softer” version of abandonment, where the platform receives some feedback before the user abandons. As example, consider optimizing the number of push notifications. When a user receives a notification, they may decide to open the app, or decide to turn off notifications. However, ignoring the notification is a likely third possible action. The platform can interpret this as a signal of dissatisfaction, and work to improve the policy.

In this section, we augment our model to capture such effects. While the solution to this updated model is intractable, we discuss interesting structure that the optimal policy exhibits: *partial learning*, and the *aggressiveness* of the optimal policy.

**Feedback model** To incorporate user feedback, we expand the model as follows. Suppose that whenever the current action $x_t$ exceeds the threshold (i.e., $x_t > \theta_t$), then with probability $q$ we receive no reward but the user remains, and with probability $1 - q$ the user abandons. Further, we assume that the platform at time $t$ both observes the reward $R(x_t)$, if rewarded, and an indicator $Z_t = \mathbb{I}_{x_t > \theta_t}$. This is equivalent to assuming that users have geometrically distributed *patience*; the number of times they allow the platform to cross their thresholds.

As before, the goal is to maximize expected discounted reward. Note that because the platform does not receive a reward when the threshold is crossed, the problem is nontrivial even when $q = 1$. We restrict our attention to the single threshold model, where $\theta$ is drawn once and then fixed for all time periods.

Figure 2 shows the numerically computed optimal policy when the threshold distribution is uniform on $[0, 1]$, the reward function is $r(x) = x$, the probability of abandonment $q = 0.5$ and $\gamma = 0.9$. Depending on whether or not a feedback signal is received, the optimal policy follows the green or the red line as we step through time from left to right.

We note that one can think of the optimal policy as a form of bisection, though it does not explore the entire domain of $F$. In particular it is conservative regarding users with large $\theta$. For example, consider a user with threshold $\theta = 0.9$. While the policy is initially increasing and thus partially personalizes to their threshold, $x_t$ does not converge to $0.9$, and in fact never comes close. We call this *partial learning*; in the next section, we demonstrate that this is a key feature of the optimal policy in general.

**Partial learning** Partial learning refers to the fact that the optimal policy does not fully reduce its uncertainty (the posterior) on $\theta$. Initially, the policy learns about the threshold using a bisection-type search. However, at some point (dependent on the user's threshold), further learning is too risky and the optimal policy switches to a constant policy. We note that this happens even when there is no risk of abandonment at all ($q = 1$), because at some point even the risk of losing a reward is not offset by potential gains in getting a more accurate posterior on $\theta$. Partial learning occurs under some regularity conditions on the threshold distribution that ensures the posterior does not collapse, and is Lipschitz as defined in the following paragraph.

Write $p(\cdot \mid l, u)$ for the posterior distribution over $\theta$ given

[4] Code to replicate the simulations is available at `https://github.com/schmit/learning-abandonment`.

[5] We also ran experiments with the MOSS algorithm (Audibert & Bubeck, 2009), which performs similarly to KL-UCB in our simulations. However, we do not have a theoretical regret bound for MOSS.

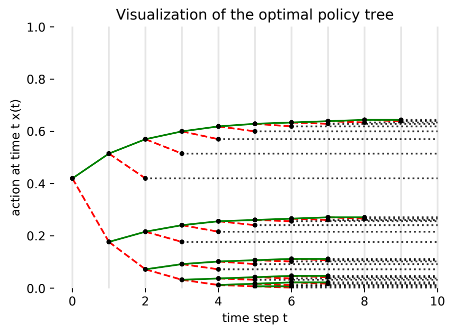

*Figure 2.* Visualization of optimal policy when discount factor $\gamma = 0.9$ in the $q = 0.5$ model. Follow the tree from left to right, where if $Z_t = 0$ (reward obtained) the next action follows from following the green line, and if $Z_t = 1$, the optimal action is given by the point following the dashed red line if the user has not abandoned.

lower bound $l$ and upper bound $u$ based on previous actions

$$p(y \mid l, u) = \mathbb{P}(l + y < \theta \mid l < \theta < u) = \frac{F(u) - F(l+y)}{F(u) - F(l)}. \quad (17)$$

We say the that the posterior distribution is non-degenerate if the following condition holds:

**Definition 2** (Non-degenerate posterior distribution )**.** *For all $\lambda > 0$, there exists a $\nu$ such that for all $l, u$ where $u - l < \nu$,*

$$p(\varepsilon \mid l, u) < 1 - \lambda\varepsilon \quad (18)$$

*for $0 < \varepsilon < \nu$.*

Thus, for sufficiently small intervals, the conditional probability decreases rapidly as we move away from the lower bound of the interval. Suppose $F$ is such that the posterior is non-degenerate and is Lipschitz in the following sense.

**Assumption 2** (Lipschitz continuity of conditional distribution)**.** *There exists an $L' > 0$ such that for all intervals $[l, u]$ and all $0 < y < u - l$, we have*

$$p(y \mid l + \varepsilon, u) - p(y \mid l, u) \leq \varepsilon L'. \quad (19)$$

We can use this assumption to show that the value function corresponding to the dynamic program that models the feedback model is Lipschitz.

**Lemma 9** (Lipschitz continuity of value function)**.** *Consider a bounded action space $\mathbf{X}$. If $p$ is Lipschitz with Lipschitz constant $L_p$, and the reward function $r$ is bounded by $B$, there exists constant $L_V$ such that for all $l < u$*

$$V(l + \varepsilon, u) - V(l, u) \leq \varepsilon L_V. \quad (20)$$

Using these assumptions, we can then prove that the optimal policy exhibits partial learning, as stated in the following proposition.

**Proposition 10.** *Suppose $r$ is increasing, $L_r$-Lipschitz, non-zero on the interior of $\mathbf{X}$ and bounded by $B$. Furthermore, assume $p$ is non-degenerate and Lipschitz as defined above. For all $u \in Int(\mathbf{X})$ there exists an $\varepsilon(u) > 0$ such that for all $l$ where $u - l < \varepsilon(u)$, the optimal action in state $(l, u)$ is $l$, that is*

$$V(l, u) = \frac{r(l)}{1 - \gamma}. \quad (21)$$

*Furthermore, $\varepsilon(u)$ is non-decreasing in $u$.*

We prove this result by analyzing the value function of the corresponding dynamic program. The result shows that at some point, the potential gains from a better posterior for the threshold are not worth the risk of abandonment. This is especially true when we believe $\theta$ to be large. If, to the contrary, we belief the threshold is small, there is little to lose in experimentation. Note however that the result also holds for $q = 1$, where users never abandon. In this case the risk of crossing the threshold (and no reward for the current time step) outweighs (all) possible future gains. Naturally, if the probability of override is small (i.e. $q$ is small), the condition on $\lambda$ also weakens, leading to larger intervals of constant policies.

**Aggressive and conservative policies** Another salient feature of the structure of optimal policies in the feedback model is the aggressiveness of the policy. In particular, we say a policy is *aggressive* if the first action $x_0$ is larger than the optimal constant policy $x^*$ in the absence of feedback (corresponding to $q = 0$), and *conservative* if it is smaller. As noted before, when there is no feedback, there is no benefit to adapting to user thresholds. However, there is value in personalization when users give feedback.

Empirically, we find that when there is low risk of abandonment, i.e., $q \approx 1$, then the optimal policy is aggressive. In this case, the optimal policy can aggressively target high-value users because other users are unlikely to abandon immediately. Thus the policy can personalize to high-value users in later periods.

However, when the risk of abandonment is large ($q \approx 0$) and the discount factor is sufficiently close to one, the optimal policy is more conservative than the optimal constant policy when $q = 0$. In this case, the high risk of abandonment forces the policy to be careful: over a longer horizon the algorithm can extract value even from low value users, but it has to be careful not to lose them in the first few periods. That is, the long term value of a user with low threshold makes up for the loss in immediate reward

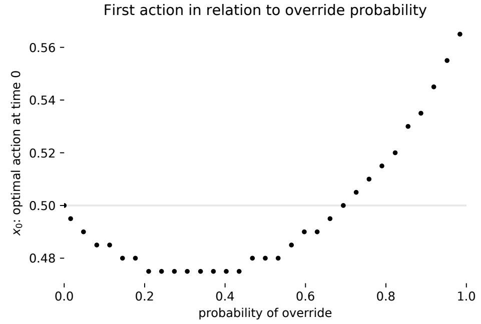

*Figure 3.* The relation between the override probability $q$ and the (approximate) optimal initial action $x_0$ when the discount factor $\gamma = 0.9$. The artifacts in the plot are due to the discretization error from numerical computations.

gained from aggressively targeting users with a high threshold. Figure 3 illustrates this effect. Here, we use deterministic rewards $r(x) = x$ and the threshold distribution is uniform $F = U[0, 1]$, but a similar effect is observed for other distributions and reward functions as well.

## 5. Conclusion

When machine learning algorithms are deployed in settings where they interact with people, it is important to understand how user behavior affects these algorithm. In this work, we propose a novel model for personalization that takes into account the risk that a dissatisfied user abandons the platform.

This leads to some unexpected results. We show that constant policies are optimal under the fixed threshold and independent threshold models. Under small perturbations of these models, constant policies are "robust" (i.e., perform well in the perturbed model), though in general finding an optimal policy becomes intractable.

Next, we consider a setting where a platform faces many users, but does not know the reward function nor population distribution over thresholds. Under suitable assumptions, UCB-type algorithms perform well, both theoretically by providing regret bounds and running simulations. An explore-exploit strategy demonstrates that dynamic policies could outperform constant policies by obtaining better information on the threshold distribution.

Feedback from users leads to more sophisticated learning strategies that exhibit partial learning; the optimal learning algorithm personalizes to a certain degree to each user. Also, we find that the optimal policy is more conservative when the probability of abandonment is high, and aggressive when that probability is low.

### 5.1. Further directions

There are several interesting directions of further research that are outside the scope of this work.

**Abandonment models** First, more sophisticated behaviour on user abandonment should be considered. This could take many forms, such as a total *patience budget* that gets depleted as the threshold is crossed. Another model is that of users playing a learning strategy themselves, comparing this platform to one or multiple outside options. In this scenario, the user and platform are simultaneously learning about each other.

**User information** Second, we have not considered additional user information in terms of covariates. In the notification example, user activity seems like an important signal of their preferences. Models that are able to incorporate such information and are able to infer the parameters from data are beyond the scope of this work but an important direction of further research.

**Lower bound for learning** Proposition 7 shows no algorithm that uses a constant policy for individual users can attain regret better than $\Omega(\sqrt{n})$. However, the explore-exploit strategy hints at the existence of dynamic policies that outperform the UCB type strategies we propose. Finding such policies is an open problem.

**Empirical analysis** This work focuses on theoretical understanding of the abandonment model, and thus ignores important aspects of a real world system. We believe there is a lot of potential to gain complementary insight from an empirical perspective using real-world systems with abandonment risk.

## 6. Acknowledgements

The authors would like to thank Andrzej Skrzypacz, Emma Brunskill, Ben Van Roy, Andreas Krause, Carlos Riquelme, Virag Shah and anonymous reviewers for their suggestions and feedback. This work was supported by the Stanford TomKat Center, and by the National Science Foundation under Grant No. CNS-1544548. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

## References

Audibert, J.-Y. and Bubeck, S. Minimax Policies for Adversarial and Stochastic Bandits. In *COLT*, pp. 217–226, 2009.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time

Analysis of the Multiarmed Bandit Problem. *Machine learning*, 47(2):235–256, 2002.

Berkenkamp, F., Turchetta, M., Schoellig, A. P., and Krause, A. Safe Model-based Reinforcement Learning with Stability Guarantees. In *NIPS*, 2017.

Bubeck, S., Cesa-Bianchi, N., et al. Regret Analysis of Stochastic and Nonstochastic Multi-Armed Bandit Problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

Farias, V. F. and Van Roy, B. Dynamic Pricing with a Prior on Market Response. *Operations Research*, 58(1):16–29, 2010.

Gallego, G. and Van Ryzin, G. Optimal Dynamic Pricing of Inventories with Stochastic Demand over Finite Horizons. *Management science*, 40(8):999–1020, 1994.

Garivier, A. and Cappé, O. The kl-ucb algorithm for bounded stochastic bandits and beyond. In *Proceedings of the 24th annual Conference On Learning Theory*, pp. 359–376, 2011.

Kleinberg, R. D. and Leighton, F. T. The Value of Knowing a Demand Curve: Bounds on Regret for Online Posted-Price Auctions. In *FOCS*, 2003.

Lobel, I. and Paes Leme, R. Dynamic Mechanism Design under Positive Commitment. 2017.

Lu, J., Kanoria, Y., and Lobel, I. Dynamic Decision Making under Customer Abandonment Risk. *MSOM*, 2017.

Moldovan, T. M. and Abbeel, P. Safe Exploration in Markov Decision Processes. *CoRR*, abs/1205.4810, 2012.

Myerson, R. B. Optimal Auction Design. *Mathematics of operations research*, 6(1):58–73, 1981.

Pavan, A., Segal, I., and Toikka, J. Dynamic Mechanism Design: A Myersonian Approach. *Econometrica*, 82(2): 601–653, 2014.

Rothschild, M. A Two-Armed Bandit Theory of Market Pricing. *Journal of Economic Theory*, 9(2):185–202, 1974.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*, volume 1. MIT press Cambridge, 1998.