Modeling Attrition in Recommender Systems with Departing Bandits

Amit Tavor, Sagie Dekel

December 2024

Online Decision Making 00970249

Contents

Abstract

As recommender systems become increasingly integral to user engagement across various platforms, understanding the dynamics of user interactions has emerged as a critical area of research. The paper "Modeling Attrition in Recommender Systems with Departing Bandits" Ben-Porat et al. (2022) addresses the critical issue of user attrition in recommender systems, where dissatisfaction can lead to users leaving permanently. The authors introduce the "departing bandits" framework, which enhances traditional multi-armed bandit (MAB) models by incorporating user types and the probabilities of clicking and departing based on recommendations. This framework allows for the development of optimal policies that achieve sub-linear regret while adapting to varying user preferences. By modeling the problem as a partially observable Markov decision process (POMDP), the authors provide insights into effective exploration and exploitation strategies.

The project will encompass a summary of the key findings, a detailed examination of the proposed algorithms and their implications for real-world applications.

1 Paper Summary

1.1 Paper subject and field

"Modeling Attrition in Recommender Systems with Departing Bandits" introduces *departing bandits*, a novel extension of the traditional multi-armed bandit (MAB) framework. This model addresses scenarios where the length of interactions is influenced by the quality of decisions, capturing situations where dissatisfaction causes participants to terminate their engagement prematurely. The study provides a foundation for a new study field in dynamic and adaptive decision-making by incorporating the risk of user departure into reward optimization.

The research is situated within the fields of online learning and multi-armed bandits, with connections to reinforcement learning and sequential decision-making. It extends traditional MAB models by proposing k MAB problems for m hidden world states, capturing the complexity and dynamics of scenarios with policy-dependent user engagement. The framework leverages techniques such as regret analysis and partially observable Markov decision processes (POMDP's) to model and address the complexities of this setup. The paper includes theoretical guarantees and proposes efficient algorithms to minimize regret in both single and multi-type user scenarios.

Real World Applications: The departing bandits framework is highly relevant in real-world applications as it addresses scenarios where decision-making policies influence not only immediate outcomes but also the duration of interactions. By incorporating policy-dependent horizons, the framework enables the optimization of both short-term rewards and long-term engagement, which are critical factors in systems where participant satisfaction and retention directly impact overall performance. This approach provides a robust solution for managing the trade-off between exploration and exploitation in dynamic environments with uncertain and heterogeneous user behavior.

1.2 Related work

This study represents a significant advancement in the intersection of several research domains, building on established foundations while addressing critical gaps. By integrating concepts from multi-armed bandits (MABs), sequential decision-making, user behavior modeling and reinforcement learning (RL), it introduces novel perspectives that expand the applicability of these frameworks. Below, we provide an in-depth examination of how this work relates to previous studies and the gaps it addresses.

- **Multi-Armed Bandits and Contextual Bandits**: Multi-armed bandits (MAB) are a foundational framework for decision-making under uncertainty, where a decision-maker selects from a set of options to maximize cumulative rewards over time. Traditional MAB frameworks focus on balancing exploration and exploitation. Building on this, Contextual bandits enhance the MAB setup by introducing context-dependent rewards, where the decision-maker observes contextual information before making a choice, allowing for more personalized and situation-specific decisions. Departing bandits further advance these frameworks by introducing critical feature, the hidden state (user type that can be seen as context) is unknown to the algorithm. These

features make the framework more dynamic and realistic, addressing gaps in existing studies that consider either fixed (could be infinite) horizons or a single homogeneous user type but not both simultaneously.

- **POMDP and User Modeling**: The use of partially observable Markov decision processes (POMDP) has been instrumental in capturing hidden states in sequential decision-making. This study builds on these methods by integrating the partially observable nature of user preferences into a bandit framework. It combines the strengths of POMDPs in handling uncertainty with the regret minimization objectives of bandit algorithms. Additionally, the paper introduces belief updates for modeling user types, effectively combining POMDP methods with bandit exploration strategies. This integration not only enhances user modeling but also scales these approaches to streaming populations, a contribution that addresses limitations in single-user or small-scale POMDP applications.
- **Reinforcement Learning in Recommender Systems**: Departing bandits are closely related to reinforcement learning (RL) in recommender systems, particularly work addressing user departure. RL frameworks often consider scenarios where agent (user) leaves the system after a bounded number of interactions, as seen in existing studies on single-user types. Departing bandits generalize this to accommodate multiple user types with potentially unbounded interaction horizons. Additionally, unlike studies that rely on historical data to approximate value functions, departing bandits operate in an online learning setting, providing theoretical guarantees for regret minimization. This approach bridges theoretical advances with practical applicability, offering a robust solution for optimizing recommendations in dynamic, user-driven environments.

1.3 Modeling the Problem

The departing bandits framework models decision-making scenarios where the interaction length between a decision-maker and participants is not fixed but depends on the quality of the decisions made. This setup introduces the concept of policy-dependent horizons, where dissatisfied participants may leave prematurely, thereby affecting both short-term and long-term rewards. Below, we describe the components, structure, and the protocol defining this model.

1.3.1 Definitions and Notations

The departing bandits problem is defined by a tuple $\langle [M], [K], q, P, \Lambda \rangle$, which encapsulates the interaction dynamics and decision-making challenges in this framework. The problem involves a sequence of participants (users) interacting with a decision-making system. Each participant belongs to one of M user types, which define their preferences for different actions (arms). The key elements of the model are:

Definition 1.1 (User Types (M))**.** *A finite set of types, each representing a distinct preference profile.*

Definition 1.2 (Categories (K))**.** *A set of actions (arms) or categories, that the algorithm can recomend from.*

Definition 1.3 (Click-Probability Matrix (P))**.** *The click-probability matrix $P \in (0,1)^{K \times M}$ defines the probability of a user clicking on a recommended category. Specifically, $P_{a,x}$ is the probability that a user of type x clicks on category a.*

Definition 1.4 (Departure-Probability Matrix (Λ))**.** *The departure-probability matrix $\Lambda \in (0,1]^{K \times M}$ defines the probability of a user departing after not clicking on a recommended category. Specifically, $\Lambda_{a,x}$ is the probability that a user of type x departs after being recommended category a without clicking.*

Definition 1.5 (Prior Distribution Vector (q))**.** *The vector $q \in [0,1]^M$ specifies the prior distribution over user types. The value q_x represents the prior probability of a user being of type x, with $\sum_{x=1}^{M} q_x = 1$.*

Definition 1.6 (Return of a Policy (V^π))**.** *The return of a policy π, denoted by V^π, is the cumulative reward obtained by the learner while interacting with a user following policy π. Formally, for a user t,*

$$V^\pi = \sum_{j=1}^{N^{\pi_t}(t)} r_{t,j}(\pi_{t,j}),$$

where $N_\pi(t)$ is the total number of iterations before the user departs, and $r_{t,j}(\pi_{t,j})$ is the reward at iteration j for the recommendation $\pi_{t,j}$.

Definition 1.7 (Regret (R_T))**.** *The regret R_T after T episodes is defined as the difference between the return of the optimal policy and the return of the learner's selected policies:*

$$R_T = T \cdot \mathbb{E}[V^{\pi^*}] - \sum_{t=1}^{T} V^{\pi_t},$$

where π^* is the optimal policy, and π_t is the policy selected by the learner in episode t.

1.3.2 Objective

The learner's goal is to maximize the cumulative rewards over T users. Intuitively, this involves striving to keep each user engaged with the system for as long as possible to maximize the number of clicks obtained from them.

1.3.3 Model Assumptions

Several key assumptions are made in the departing bandits framework:

- The user type distribution is stationary and known to the learner.
- The departure probabilities depend only on the user's satisfaction with the recommendation (no-click).
- The reward probabilities and departure probabilities for each user type-arm pair are unknown and must be learned over time.

1.3.4 Problem Types:

This section describes the two interconnected problems addressed in the departing bandits framework: the **planning problem**, where the parameters of the system are known, and the **learning problem**, where these parameters must be learned over time.

- **Planning Problem:** The planning problem assumes that the reward and departure probabilities for each user type-arm pair are **known**. The goal is to determine the optimal policy that maximizes cumulative rewards over a single user's interaction sequence. This is modeled as a partially observable Markov decision process (POMDP), where the user's type is treated as a hidden state. Based on the user's interaction history, the system updates its belief about the user type and uses this belief to derive the optimal recommendation strategy. This problem is localized and focuses solely on optimizing outcomes for a single user, leveraging the fixed matrices P and Λ.
- **Learning Problem:** In contrast, the learning problem assumes that the reward and departure probabilities are **unknown** and must be estimated through interaction with a stream of users. The objective here is to minimize *regret*, which measures the cumulative gap between the rewards obtained by the algorithm and those achievable under the optimal policy with known parameters. This problem is global, addressing the optimization of cumulative rewards across multiple users.

1.3.5 Interaction Protocol

The interaction between the decision-maker and users follows the protocol described in Algorithm 1. This protocol models the iterative process of user engagement, recommendation, and departure while capturing dynamic user behavior. For each user, a hidden type is sampled from a prior distribution representing their preferences. The decision-maker selects a recommendation (arm), and the user either clicks (reward = 1) or does not click (reward = 0), based on predefined click probabilities. If the user does not click, they may depart with a probability determined by the departure probabilities. The interaction continues until the user departs, after which the system moves on to the next user.

This protocol provides a foundation for modeling realistic interactions between a system and its users, capturing the dynamics of user behavior and system responses.

Algorithm 1 The Departing Bandits Protocol

Input: Number of types M, number of categories K, and number of users (episodes) T.
Hidden Parameters: Types prior q, click-probability P, and departure-probability Λ.

for episode $t \leftarrow 1$ to T **do**
  A new user with type type$(t) \sim q$ arrives.
  $j \leftarrow 1$, depart $\leftarrow$ false.
  while depart = false **do**
    The learner picks a category $a \in [K]$.
    With probability $P_{a,x}$, user t clicks on a and $r_{t,j}(a) \leftarrow 1$; otherwise, $r_{t,j}(a) \leftarrow 0$.
    if $r_{t,j}(a) = 0$ **then**
      With probability $\Lambda_{a,x}$: depart $\leftarrow$ true and user t departs.
    end if
    The learner observes $r_{t,j}(a)$ and depart.
    if depart = false **then**
      $j \leftarrow j + 1$.
    end if
  end while
end for

1.3.6 Performance Metric

The primary performance metric is regret, defined as the difference between the expected cumulative reward of the optimal policy and the reward achieved by the learner. For T users, regret is expressed as:

$$R_T = T \cdot \mathbb{E}[V^*] - \sum_{t=1}^{T} V^{\pi_t},$$

where V^* is the expected return of the optimal policy, and V^{π_t} is the return of the policy used for user t.

1.3.7 Challenges

The departing bandits problem introduces complexities not found in traditional MAB settings:

- **User-Type Inference**: Since user types are not observed directly, the learner must update its beliefs based on observed interactions.
- **Dynamic Optimization**: Balancing short-term gains with long-term learning requires solving a partially observable decision problem.

This model and its protocol form the foundation for the theoretical analysis and algorithmic development presented in the paper. It represents a significant extension of classical bandit frameworks by incorporating dynamic user interactions and policy-dependent engagement.

1.4 Main Results

The departing bandits framework introduces innovative algorithms and theoretical guarantees for regret minimization under policy-dependent horizons. These results demonstrate the effectiveness in single-type and two-type user scenarios, as well as its ability to handle sub-exponential returns. Below is a summary of the key findings.

- **Single User Type**: In the single-user-type scenario, the planning problem is straightforward since we know the user type and the click and departure probabilities, which reduces the problem into a knowledge MAB problem. The optimal policy, a fixed-arm strategy, is determined in constant time $O(K)$ by comparing the known parameters of each arm. For the learning problem,

where these parameters are unknown, the proposed algorithm based on the Upper Confidence Bound (UCB) framework achieves sublinear regret $O(\sqrt{KT \log T})$ (or $\tilde{O}(\sqrt{T})$), where T is the number of users. This result demonstrates the algorithm's ability to efficiently learn the unknown parameters while minimizing the cumulative difference between the learner's policy and the optimal policy over time.

- **Two User Types And Two Categories**: For this case, the planning problem is modeled as a partially observable Markov decision process (POMDP), where the user type is a hidden state. By characterizing the optimal policy, it can be derived in $O(1)$, making its computation effectively constant in this setting. This ensures the algorithm efficiently handles the heterogeneity of user types. In the learning problem, where the reward and departure probabilities are unknown, the algorithm dynamically updates its belief about user types based on observed interactions. It achieves sub-linear regret $O(\sqrt{T} \log T)$ (or $\tilde{O}(\sqrt{T})$), ensuring an effective balance between exploring the preferences of different user types and exploiting the learned preferences to maximize rewards in complex and dynamic user populations.

These results demonstrate the effectiveness in single-type and two-type user scenarios, as well as its ability to handle sub-exponential returns. Below is a summary of the key findings.

1.4.1 General Observations

The paper provides additional insights into the robustness and applicability of the departing bandits framework:

- The proposed algorithms for single-type and two-type user scenarios adapt dynamically to user behavior and demonstrate superior performance in cases where naive policies, such as fixed-arm strategies, are suboptimal and may suffer from linear regret. However, in specific scenarios, particularly in the planning problem, the optimal policy may coincide with a fixed-arm strategy, depending on the system's parameters and user dynamics.

- The belief-based approach efficiently manages uncertainty in user types, prioritizing arms with high expected returns and adapting dynamically as more interactions occur.

- The framework leverages the fact of sub-exponential returns, as proven in the article, to extend beyond simple reward models, such as Gaussian or Bernoulli distributions. This makes it applicable to diverse real-world scenarios while maintaining regret guarantees.

These findings establish the departing bandits framework as a versatile and effective extension of traditional multi-armed bandits. By addressing challenges such as policy-dependent horizons, user heterogeneity, dynamic decision-making, and sub-exponential reward distributions, the paper algorithms and suggestions form a basis to solve the planning and learning departing bandits problem with sub-linear regret bounds.

2 Implementation

In this section we will present the main ideas, claim and proofs steps as presented in the paper. As written in the paper, we will divide this part into 3: (1) Problem definition, notations and other related tools to the Implementation. (2) Analyses of Single User Type framework. (3) Analysis of two user types and two categories framework.

2.1 Introduction

2.1.1 Sub-exponential Returns

Sub-exponential returns are a generalization of sub-Gaussian rewards often used in bandit algorithms. These returns allow for heavier tails compared to sub-Gaussian distributions, making the analysis more robust to different frameworks. Specifically, a random variable is sub-exponential if it satisfies the following property:

Definition 2.1 (Sub-exponential Random Variables)**.** *A random variable X is sub-exponential with parameters (τ^2, b) if for every γ such that $|\gamma| < 1/b$,*

$$\mathbb{E}\left[\exp\left(\gamma(X - \mathbb{E}[X])\right)\right] \leq \exp\left(\frac{\gamma^2\tau^2}{2}\right).$$

In addition, for every (τ^2, b)-sub-exponential random variables, there exist constants C_1, $C_2 > 0$ such that the above is equivalent to each of the following properties:

1. ***Tails:*** *For all $v \geq 0$, $\Pr[|X| > v] \leq \exp\left(1 - \frac{v}{C_1}\right)$,*
2. ***Moments:*** *For all $p \geq 1$, $(\mathbb{E}[|X|^p])^{1/p} \leq C_2 p$,*

where C_1 and C_2 are constants dependent on (τ^2, b).

2.1.2 UCB-Hybrid Algorithm

The main algorithm in the paper is a modification of UCB — UCB-hybrid Jia et al. (2021) for sub-exponential rewards. Given a set of sub-exponential policies, the algorithm tries to find the best policy, as in the MAB problem. In the algorithm, each policy is an arm, and its reward is the cumulative return from a user. The "hybrid" part means that the algorithm adapts its exploration-exploitation balance dynamically based on the number of samples collected for each policy. As we will show in the learning problem, the algorithm is important because we can find a set of sub-exponential policies, containing the optimal policy, and use it to achieve sub-linear regret (with respect to the number of users).

Algorithm 2 UCB-based algorithm with hybrid radii: UCB-Hybrid Jia et al. (2021)

Require: Set of policies Π, number of users T, parameters $\tilde{\tau}$, η

Initialize: $\forall \pi \in \Pi$: $U_0(\pi) \leftarrow \infty$, $n(\pi) \leftarrow 0$
for user $t \leftarrow 1$ to T **do**
  Execute $\pi_t \in \arg\max_{\pi \in \Pi} U_{t-1}(\pi)$
  Receive return $\hat{V}^{\pi_t}[n(\pi_t)] \leftarrow \sum_{j=1}^{N_{\pi_t}(t)} r_{t,j}(\pi_{t,j})$
  $n(\pi_t) \leftarrow n(\pi_t) + 1$
  if $n(\pi_t) < 8\eta \ln T$ **then**
    Update $U_t(\pi_t) \leftarrow \frac{\sum_{i=1}^{n(\pi_t)} \hat{V}^{\pi_t}[i]}{n(\pi_t)} + 8\frac{\sqrt{n}\tilde{\tau}\ln T}{n(\pi_t)}$
  else
    Update $U_t(\pi_t) \leftarrow \frac{\sum_{i=1}^{n(\pi_t)} \hat{V}^{\pi_t}[i]}{n(\pi_t)} + \sqrt{\frac{8\tilde{\tau}^2 \ln T}{n(\pi_t)}}$
  end if
end for

Theorem 2.2. *Let Π be a set of policies with parameters $\tilde{\tau}$ and η. Let $\pi_1, \pi_2, \ldots, \pi_T$ be the policies selected by Algorithm 2. It holds that:*

$$\mathbb{E}\left[\max_{\pi \in \Pi} T \cdot V^{\pi} - \sum_{t=1}^{T} V^{\pi_t}\right] = O\left(\sqrt{|\Pi| T \ln T} + |\Pi| \ln T\right).$$

Where $\tilde{\tau}$ and η are the associated parameters from section 3.1.2.

Proof. The theorem follows directly from Theorem 3 of Jia et al. (2021) by using policies as arms and returns as rewards. □

Derived from Theorem 2.2 that if we can find small sub-set of policies containing the optimal policy, we can solve the problem with sub-linear regret (in the users number T).

2.2 Single User Type

In this section we focus on the case of a single user type and K categories. The paper shows that using Algorithm 2 and policy set of all the fixed-arm policies we can derive $\tilde{O}(\sqrt{T})$ regret bound. Note we assume that the number of categories is much less than users number T.
The paper shows that the planing problem, which in we know P is degenerated because we can always choose the best arm to use. As a result, the paper analyzes the learning problem of this case. The steps we will follow to achieve the result are:

- Proof that the optimal policy is a fixed-arm policy.
- Proof that the fixed-arm policies returns are sub-exponential and calculate their parameters.
- Combining those insights with algorithm 2 with all fixed-arm policies we and Thorem 2.2 we will achieve the above regret bound.

Lemma 2.3. *A policy π_{a^*} is optimal if*

$$a^* \in \arg\max_{a \in [K]} \frac{P_a}{\Lambda_a(1-P_a)}$$

Proof Sketch. The proof proceeds as follows:

- Having single type user reduce our problem into the classic MAB problem, there exists a fixed-arm policy which is optimal.
- Note that the return of fixed-arm policy in single user type case is a binomial random variable with N_{π_t} number of rounds and P_a success probability (the rewards are 0 or 1).
- It is easy to see that the number of rounds N_{π_t} for each user follows a geometric distribution with a success probability parameter $\Lambda_a(1-P_a)$.
- Using Wald's Equation we can achieve the expected return of fix-arm policy:

$$\mathbb{E}[V_{\pi_a}] = \mathbb{E}[\sum_{i=1}^{N^{\pi_a}(t)} r_a(\pi_a)] = \mathbb{E}[N^{\pi_a}(t)] \cdot \mathbb{E}[r_a(\pi_a)] = \frac{P_a}{\Lambda_a(1-P_a)}$$

  as written in the Lemma above.

It directly follows from the Lemma that the planning problem is trivial and can be solved in $O(K)$. However, to address the learning problem with the specified regret bound, we must first demonstrate that fixed-arm policies yield sub-exponential returns. Once this is established, the regret bound will follow directly from algorithm 2. It is worth mentioning that the optimal policy is not the one that will choose the category with the highest click probability or lowest departs probability, but a combination of both, as we saw in the Lemma above.

Lemma 2.4. *For each category $a \in [K]$, the centered random variable $V^{\pi_a} - \mathbb{E}[V^{\pi_a}]$ is sub-exponential with parameters (τ_a^2, b_a), such that*

$$\tau_a = b_a = -\frac{8e}{\ln(1-\Lambda_a(1-P_a))}.$$

Proof Sketch. The proof proceeds as follows:

- Sub-exponentiality of N_a: As we mentioned before, N_a is a geometric random variable, based on Lemma 4.3 of Hillar and Wibisono (2018) it is Sub-exponentiality.
- Bounding V^{π_a}: Since V^{π_a} is bounded by $N_a - 1$, Corollary D.4 show that the tails of V^{π_a} are also sub-exponential.
- By applying Minkowski's and Jensen's inequalities, we bound the moments of $V^{\pi_a} - \mathbb{E}[V^{\pi_a}]$, ensuring that they satisfy Property (1) from Definition 3.1 for each return of the fixed-arm policy.

- packing all that by applying a normalization trick and bound the Taylor expansion of the moment-generating function $\mathbb{E}[\exp(\gamma(V^{\pi_a} - \mathbb{E}[V^{\pi_a}]))]$. This step establishes that the returns of fiexd-arm policies are sub-exponential with parameters (τ_a^2, b_a).

After establish that the optimal policy is one of the fixed-arm policies and each return of those policies is sub-exponential, the following Theorem, which gives us the requested regret bound, derived directly from applying algorithm 2 (and using Thorem 2.2) with all fixed-arm policy set and parameters as described in the Theorem.

Theorem 2.5. *For single-type users ($M = 1$), running Algorithm 2 with $\Pi = \{\pi_a : a \in [K]\}$, $\tilde{\tau} = \frac{8e}{\ln\left(\frac{1}{1-\epsilon}\right)}$, and $\eta = 1$ achieves an expected regret of at most:*

$$\mathbb{E}[R_T] = O\left(\sqrt{KT\log T} + K\log T\right).$$

Note that $\tilde{\tau} \geq \tau$ for each category $a \in [K]$ due to the assumptions $\max_{a,x} P_{a,x} \leq 1 - \epsilon$ and $\Lambda_a \leq 1$.

To conclude this section, we saw that the single type user case planning problem is trivial and degenerated. On the other hand, in the learning problem, where the algorithm doesn't know P, we achieved sub-linear regret bound.

2.3 Two User Types and Two Categories

In this section, the paper analyses the cases of two user types and two categories assuming that the departure-probability $\Lambda_{a,x}$ is 1 for each category a and user type x. As we will see, even this "simple" case is not trivial and requires smart planning or learning. The paper shows that for all fixed-arm policies exists at least one instance that will cause linear regret (see section 2.1 in the paper for example). Note that we are considering partially observable MDP since that user type is unknown. Unexpectedly, although the policy space size grows exponentially, the paper proves we can find the optimal policy in constant time ($O(1)$) in the planning problem. Later on, we can leverage the optimal policy characterization (see later in the planning problem) to derive a small policy set of size $O(ln(T))$ that applying Algorithm 2 with this set will result in $\tilde{O}(\sqrt{T})$ regret bound. The steps we will follow to achieve the results are:

- We will define our world belief update rule and Belief-category walk. Using those, the paper proves that a policy expectation can be computed using a closed formula.
- Characterizing the Optimal Policy (will result in 4 possible optimal policy types).
- Combining those insights and finding a small "good" policy set, using algorithm 2 will achieve sub-linear regret bound.

2.3.1 Introduction to the Planning Problem

To maintain brevity, we will not redefine the POMDP, as defined in section 5.1 in the paper. Also, we will not repeat Remark 5.1 and Definition 5.2 (Belief-category walk) as written in the paper.

First, because we are in POMDP setting we need to define our belief state. In our case, the states are the user types. We denote by b_j the belief that the state is (type) x at iteration j. The update rule of our belief is described in equation (2) in the paper, here we want to explain this rule.

Belief updating is based on the Bayes' theorem which provides a principled way to update probabilities based on new evidence. At iteration j, we aim to compute the updated belief b_j, which represents the probability that the state is x given a positive reward ($r_j = 1$) and the prior belief b_{j-1}. By Bayes' theorem:

$$b_j = P(\text{state} = x \mid r_j = 1) = \frac{P(r_j = 1 \mid \text{state} = x) \cdot P(\text{state} = x)}{P(r_j = 1)}.$$

Breaking this into components in our setting:

- $P(r_j = 1 \mid \text{state} = x) = P_{a,x}$, the probability of observing a positive reward when the state is x.
- $P(\text{state} = x) = b_{j-1}$, the prior belief that the state is x.
- Based on Law of Total Probability:

$$P(r_j = 1) = b_{j-1} \cdot P_{a,x} + (1 - b_{j-1}) \cdot P_{a,y}.$$

Substituting these into Bayes' theorem:

$$b_j = \frac{b_{j-1} \cdot P_{a,x}}{b_{j-1} \cdot P_{a,x} + (1 - b_{j-1}) \cdot P_{a,y}}. \tag{1}$$

To sum the above, this formula updates the belief b_j by incorporating the likelihoods of observing the reward under both states, weighted by the prior beliefs.

Let us proceed to important closed formula to compute the expected return of a policy:

Theorem 2.6. *For every policy π and an initial belief $b \in [0, 1]$, the expected return is given by:*

$$\mathbb{E}[V^{\pi}(b)] = \sum_{i=1}^{\infty} b \cdot P_{1,x}^{m_1,i} \cdot P_{2,x}^{m_2,i} + (1 - b) \cdot P_{1,y}^{m_1,i} \cdot P_{2,y}^{m_2,i},$$

where $m_1, i := |\{a_j = 1, j \leq i\}|$ and $m_2, i := |\{a_j = 2, j \leq i\}|$ are calculated based on the belief-category walk $b_1, a_1, b_2, a_2, \ldots$ induced by π.

Proof. This Theorem derives directly from summing policy π category recommendation click-probebility in time i and using the Law of Total Probability over the state belief in time i □

2.3.2 Characterizing the Optimal Policy

In this section the paper shows that the planning problem can be solved in O(1), Using Theorem 2.6. It doing so by prove that any matrix P takes exactly one of the three following structures:

- **Dominant Row**, where $P_{1,y} \geq P_{2,y}$ for each user type y.
- **Dominant Column**, where $P_{2,x} \geq P_{2,y} > P_{1,y}$.
- **Dominant Diagonal**, where $P_{1,x} \geq P_{2,y} > P_{1,y}, P_{2,x}$.

Because of the length and the similarity between the cases proofs, we chose to focus in the Dominant Column (DC) case, which it's optimal policy set containing the other cases possible optimal policies. The main Theorem in this case is as follows:

Theorem 2.7. *For any instance such that P has a dominant column, one of the following four policies is optimal:*

$$\pi^1, \quad \pi^2, \quad \pi^{2:\lfloor N^* \rfloor}, \quad \pi^{2:\lceil N^* \rceil},$$

where $N^* = N^*(P, q)$ is a constant, and $\pi^{2:\lfloor N^* \rfloor}$ ($\pi^{2:\lceil N^* \rceil}$) stands for recommending Category 2 until iteration $\lfloor N^* \rfloor$ ($\lceil N^* \rceil$) and then switching to Category 1.

Proof Sketch. The proof proceeds as follows:

- First, the author proves optimal policy recommends topic 1 for all iteration $j` \geq j + 1$ such inequality (9) from the paper holds (Lemma E.1). It doing in two parts. The first part assume by contradiction that an optimal policy π recommends topic 2 at iteration $j+1$ while inequality (9) holds. Construct a modified policy π_j that switches to topic 1 at $j+1$, keeping all earlier recommendations identical. By expanding the difference in expected returns, $\mathbb{E}[V^{\pi_j}(b)] - \mathbb{E}[V^{\pi}(b)]$, and using the given probability relationships, it is shown that this violates inequality (9). For the second part, the inequality (9) proved if π is optimal and recommends category 1, by exploiting the optimal policy recommendations and the existence of $j + 1$.

- An immediate corollary from Lemma E.1 is that the optimal policy first recommends topic 2 for at most N^* times, and than recommends topic 1 permanently (see Theorem 5.5 full proof in the paper for exact N^*). The existence of N^* is guaranteed for each policy because of the updating belief and the DC (Dominant Column) matrix P. If N^* exists so from it's inequality we can derive inequality (9). As we saw in Lemma E.1, if inequality (9) holds for some time for the optimal policy, than it will recommend category 1 permanently.
  Note that N^* might be 0 if the prior belief sustains $q_x \geq q_y$ because of Remark 5.1 in the paper.

- Last, the exact calculation of N^* is done by writing the return of a policy as a function f of some time N. The function f can be maximized in the boundaries ($N = 0, N = \inf$) or in the closest integers to the single saddle point ($\lceil N^* \rceil, \lfloor N^* \rfloor$).
  Note that the boundaries represent the fixed-arm policies and the others the "switch in the middle" policies

- It's important to mention that the proof didn't show the possibility of optimal policy to be π^2. This is because of Remark 5.1, without it (real-world scenarios) the policy π^2 can also be the optimal policy in the DC case.

At this point we ready to solve the planning problem:

Planning Problem Remembering that in the planning problem we have full observation of P and by combining Theorem 2.6 with Theorem 2.7, we get 4 possible optimal policy that each one's return can computed in $O(1)$. Therefor, we can computed the 4 returns in constant time and simply get the optimal policy by choosing the one with the maximum return. Note that the optimal policy set in the DC case contain the optimal policy sets of DR and DD cases, so by doing the above we also solve the other cases planning problem.

2.3.3 Learning Problem

In this section we move from the planning problem to the learning problem. Based on the previous section we know that the optimal policy fall into one of the four types as in Theorem 2.7, yet resulting in infinite policies (because we don't have access to P). To address this, the paper shows how to find a small threshold policy (see definition 5.7 in the paper) set of size $O(ln(T))$ with negligible additive factor. The intuition behind the reduced set creation is the observation that the probability that a user will stay beyond H iterations is exponentially decreasing with H (due to the assumption: $\max_{a,x} P_{a,x} \leq 1 - \epsilon$). With that observation we infer that we might satisfied with restricted horizon threshold policy set. Based on that set and Theorem 2.8, we can apply algorithm 2 with the corresponding parameters to achieve sub-linear regret bound.

Let Π_H be the set of all (a, h)-threshold policies for $a \in \{1, 2\}$ and $h \in [H] \cup \{0\}$. Clearly, $|\Pi_H| = 2H + 2$. Next, we analyze the impact of this restriction:

Lemma 2.8. *For every $H \in \mathbb{N}$, it holds that:*

$$\mathbb{E}\left[V^{\pi^*} - \max_{\pi \in \Pi_H} V^{\pi}\right] \leq \frac{1}{2^{O(H)}}.$$

Proof Summary. The proof begins by expressing the return of the optimal policy π as:

$$V^{\pi} = \sum_{j=1}^{H} \mathbb{I}[j \leq N^{\pi}] \cdot r_j(\pi_j) + \sum_{j=H+1}^{\infty} \mathbb{I}[j \leq N^{\pi}] \cdot r_j(\pi_j),$$

where $\mathbb{I}[\cdot]$ is an indicator function. A similar representation is used for $V^{\pi'}$. We know the optimal policy is a threshold policy and therefor until time H it recommends the same as at least one policy in Π_H. Taking expectations and evaluating the difference $\mathbb{E}[V^{\pi} - V^{\pi'}]$ with that policy, it is shown that:

$$\mathbb{E}[V^{\pi} - V^{\pi'}] \leq \sum_{j=H+1}^{\infty} (1-\epsilon)^j (1-\epsilon) \leq (1-\epsilon)^H \cdot \frac{1}{\epsilon}.$$

By bounding the terms and simplifying using the geometric series, the result is:

$$\mathbb{E}[V^{\pi} - V^{\pi'}] \leq \frac{1}{2^{O(H)}}$$

Before using Algorithm 2 we need to show that the returns of the policies of Π_H are sub-exponential and find the corresponding parameters. It is shown in Lemma 5.9 in the paper. We chose not to deepen that Lemma because of similarity to Lemma 2.4 (the proof also based on it). Note that the parameters of the current policies set is the same as before.

Let us proceed to finalize our solution by define $H = \Theta(ln(T))$, Π_H as defined before and $\tilde{\tau}$, η be constants as in Lemma 2.5.

Theorem 2.9. *Applying Algorithm 2 with $\Pi_H, T, \tilde{\tau}, \eta$ on the class of two-types, two-categories instances considered in this section always yields an expected regret of:*

$$\mathbb{E}[R_T] \leq O(\sqrt{T} \ln T).$$

The proof derive directly from bounding the expected regret of algorithm 2 with Π_H and the upper bound of the difference between the optimal policy and the optimal policy in Π_H.

To conclude this section, we can achieve sub-linear regret bound in the learning problem by learn the best policy within sub-set of threshold policies of horizon that derive from the users number T.

3 Our Extension

3.1 Motivation

The framework of departing bandits introduces a fascinating perspective in dynamic decision-making by modeling user interactions in recommender systems while incorporating attrition and policy-dependent horizons. However, the framework relies on certain simplifications that limit its applicability to more complex scenarios, particularly in real-world systems where user behaviors are dynamic and context-dependent. One of the primary limitations lies in the assumption of fixed probabilities for user interactions, which fails to capture evolving dynamics influenced by contextual or temporal factors. These strategies also overlook the complexities of systems where user retention and recommendation quality are interdependent.

The motivation for addressing these limitations lies in the importance of creating models that better reflect the complexity of real-world systems, where user behaviors are dynamic, influenced by context, and evolve over time. In our extension, we aim to learn how group dynamics influence retention and satisfaction and translate this understanding into actionable strategies for optimizing recommendations and resource allocation. By understanding these dynamics, we seek to identify the optimal investment required to attract and retain users based on their collective and individual projected value. This approach can help systems adapt dynamically to user behavior, ensuring both short-term engagement and long-term sustainability while addressing the complexities of real-world scenarios where user interactions are interdependent and context-driven.

3.2 Problem Defenition

We introduce an extension to the Departing Bandits problem, referred to as Departing Bandits with Network Effect (DBNA). This variant introduces two key differences: (1) the departure probabilities now depend on the number of users present in the system, and (2) the algorithm determines how many users it should "pay" for. While the overall objective remains unchanged, the algorithm must now account for additional variables. To maintain consistency, most of the definitions remain as outlined in the original problem. To describe the new protocol, we define the following:

Definition 3.1 (Departure-Probability Matrix with Network affect ($\Lambda(x)$)). *The departure-probability matrix $\Lambda(x) \in (0,1]^{K \times M}$ defines the probability of a user departing after not clicking on a recommended category in x users environment. Specifically, The departure probability should be decreasing as x increasing.*

We assume $\forall a \in K, x \in M: \quad \Lambda_{a,x}(x) \geq \epsilon$ for a small $\epsilon > 0$, i.e., the departure probability does not converge to 0 and no user type is staying forever.

Definition 3.2 (Cost Function ($f(c)$)). *The Cost Function $f : R \to N$ defines the users number arrives for each price $c \in R$.*

The DBNA protocol goes as follows:

Algorithm 3 DBNA Protocol

Input: Number of types M, number of categories K, and number of phases (episodes) T.
Hidden Parameters: Types prior q, click-probability P, and departure-probability $\Lambda(x)$.

for phase $t \leftarrow 1$ to T **do**
  The learner chooses how much to "pay" $c^t \in R$.
  $x_1^t \leftarrow f(c^t)$ users with i.i.d types arrives, where $\forall i \in [f(c^t)]: \quad \text{type}(i) \sim q$.
  $r \leftarrow 1$, depart $\leftarrow$ false, left $\leftarrow 0$.
  while $x_r^t > 0$ **do**
    for each user $x \in x_r^t$ **do**
      The learner picks a category $a \in [K]$.
      With probability $P_{a,x}$, user x clicks on a and $reward_{t,r,x}(a) \leftarrow 1$;
      otherwise, $reward_{t,r,x}(a) \leftarrow 0$.
      if $reward_{t,r,x}(a) = 0$ **then**
        With probability $\Lambda_{a,x}(x_r^t)$: depart $\leftarrow$ true and user x departs.
      end if
      if depart = True **then**
        left = left + 1
      end if
    end for
    The learner observes the rewards and left.
    $x_{r+1}^t \leftarrow x^t -$ left
    if $x_{r+1}^t > 0$ **then**
      $r \leftarrow r + 1$.
    end if
  end while
end for

Note we use t for phase and r for round. Policy for phase t will be marked as π_t.
Having described the protocol, we move on to the goals of the learner. The goals are same to the paper, but the return of a policy is slightly different:

Definition 3.3 (Return of a Policy (V^π)). *The return of a policy π, denoted by V^π, is the cumulative reward obtained by the learner while interacting with a user following policy π. Formally, for a user t,*

$$V^\pi = \sum_{x=1}^{x^\pi} \sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x)) \quad - \quad c^\pi$$

where $N_\pi(t)$ is the total number of iterations before all users in phase t departs, x^π is the number of users the phase, c^π is the cost for x^π users and $reward_{t,r,x}(\pi_{t,r}(x))$ is the reward at iteration r in phase t of user x for the recommendation $\pi_{t,r}(x)$. Intuitively, if a user departs before the end of the phase we define his future rewards to be 0.

The regret definition doesn't change and the learner's goal is to minimize $\mathbb{E}[R_T]$.

3.3 Single User Type

To illustrate the complexity of the problem, we analyze scenarios involving a single user type and two categories. We demonstrate that, even in this simplified case, the optimal policy is not a fixed-arm policy as proposed in the original paper. Within the scope and limitations of this project, we aim to explore and propose potential solutions to the problem.
It is important to highlight that the rewards for each category no longer have an expectation dependent on a single parameter, as in the original problem. Consequently, the claim from the paper regarding the optimality of fixed-arm policies does not necessarily hold in our case. In this section, we assume that x^t is given and do not address the trade-off between the return and the cost function. Additionally, we omit the phase marker t in most places since our solution applies universally to any phase.
Lets start with some observation and definitions:

Observation 3.4. *For each round r, given number of users x_{t-1} and recommendation from round $r-1$, it holds that:*

$$x_r \mid x_{r-1} \sim \textit{Binomial}(x_{r-1}, 1-(1-P_a)\cdot\Lambda(x_{r-1})_a)$$

Where a is the recommended category from round $r-1$.

Proof. First note that each user leaving decisions are i.i.d. Since all users received the same recommendation a in round $r-1$ and have the same click probability and departure probability, we can think of it as a game with x_{t-1} users and winning probability of staying in the platform. □

Definition 3.5 ($N^{\pi_t}(t)$)**.** *Denotes the total number of iterations with policy π_t interact with users in phase t until all users departs.*

Definition 3.6 ($N_i^{\pi_t}(t)$)**.** *Denotes the total number of iterations with policy π_t interact with users in phase t from the $i-1$ change in the number of users until the following one.*

Observation 3.7.

$$N^{\pi_t}(t) = \sum_{i=1}^{C_t} N_i^{\pi_t}(t)$$

Where C_t is random variable which count the number of changes in users number in phase t.

Note that C_t is bounded by x^t, the number of users the phase.

Definition 3.8 (Half Fixed-Arm Policy)**.** *A policy π_t is Half Fixed-Arm Policy if it changes the recommended category only after a change in the number of users, formally:*

$$\forall i, \quad 1+\sum_{j=1}^{i-1} N_j^{\pi_t}(t) \le r \le \sum_{j=1}^{i} N_j^{\pi_t}(t), \quad \exists a \in K: \quad \pi_{t,r} = a$$

Where for $i=0$ we define $\sum_{j=1}^{i-1} N_j^{\pi_t}(t) = 0$.

Lemma 3.9. *If π_t is Half Fixed-Arm Policy then for each round r and i, given number of users in round r, it holds that:*

$$N_i^{\pi_t}(t) \sim \textit{Geom}(1-P(x_{r+1}=x_r))$$

Intuitive the i'th change in the number of users is geometric variable with the probability of all users to stay for the next round. Note that we can calculate $P(x_{r+1}=x_r)$ easily using Observation 3.4.

Proof. Given x_r and the fact that π_t is a Half Fixed-Arm Policy, we can conceptualize an experiment where failure is defined as "all users remain in the system." This is because the staying probability remains constant for π_t until the number of users changes and the users decisions are independent. This experiment naturally defines a geometric random variable, where the parameter is the probability that at least one user will leave. □

Now, as in the paper, we rely on the fact that all users have the same type and notice that given x_r each category $a \in K$ expected reward depends on two parameters: $P_a, \Lambda_a(x_r)$. Therefor, there exists an optimal fixed-arm policy (not necessarily the same arm), until the next change in the number of users, i.e., the optimal policy is Half Fixed-Arm Policy, Formally:

Lemma 3.10. *A policy π_t^* is optimal if in each round r, it holds that:*

$$\forall i \in [c], \quad \pi^*_{t,r} \in \arg\max_{a\in[K]} \mathbb{E}[N_i^{\pi_t^*}(t)] \cdot P_a = \frac{P_a}{(1 - P(x_{r+1} = x_r))}$$

Where c is the number of times a change in the number of users has occurred during phase t.

Proof. In the proof we drop the cost function since we assume that part x^t is given. It holds that:

$$\mathbb{E}[V_{\pi_t}] = \mathbb{E}[\sum_{x=1}^{x^t} \sum_{r=1}^{N^{\pi_t}(t)} r_{t,r,x}(\pi_{t,r}(x))] \overset{\text{single user type}}{=} x^t \cdot \mathbb{E}[\sum_{r=1}^{N^{\pi_{t,r}}(t)} r_{t,r,x}(\pi_{t,r}(r))] \overset{\text{observation 3.7}}{=}$$

$$x^t \cdot \mathbb{E}[\sum_{r=1}^{N_1^{\pi_t}(t)} r_{t,r}(\pi_{t,r}) + ... + \sum_{r=N_{c-1}^{\pi_t}(t)}^{N^{\pi_t}(t)} r_{t,r}(\pi_{t,r})] \overset{\text{expectation linearity}}{=}$$

$$x^t \cdot (\mathbb{E}[\sum_{r=1}^{N_1^{\pi_t}(t)} r_{t,r}(\pi_{t,r})] + ... + \mathbb{E}[\sum_{r=N_{c-1}^{\pi_t}(t)}^{N^{\pi_t}(t)} r_{t,r}(\pi_{t,r})]) \overset{\text{Wald's equation}}{=}$$

$$x^t \cdot (\mathbb{E}[N_1^{\pi_a}(t)] \cdot \mathbb{E}[r_{t,r}(\pi_{t,r})] + ... + \mathbb{E}[N_{c-1}^{\pi_a}(t)] \cdot \mathbb{E}[r_{t,r}(\pi_{t,r})]) \overset{\text{indecator expectation}}{=}$$

$$x^t \cdot (\mathbb{E}[N_1^{\pi_a}(t)] \cdot P_{\pi_{t,1}} + ... + \mathbb{E}[N_{c-1}^{\pi_a}(t)] \cdot P_{\pi_{t,1+\sum_{i=1}^{c-1} N_i^{\pi_t}(t)}})$$

To get the equation as in the Lemma we use Lemma 3.9 to place the expectation of a geometric random variable. □

As a consequence of this lemma, the planning problem for single-type users is trivial - after each change in the number of users, find the category that maximize the expected return using Lemma 3.10 and recommend it until the next change. Suppose we consider x^t as constant, we able to solve the planning problem in $O(k)$, as in the paper. In the scope of this work we were unable to find a close solution to the problem with the cost function, but we believe it can be solved in a similar way.

Informally, we remark that the learning problem can be solved in a similar way to the paper, with the same regret bound up to a constant. The main difference is that in our case we should hold fixed-arm policies set for each number of users and reboot Algorithm 2 with the corresponding set after each change in the number of users during each phase.

3.4 Cost Function - Discussion

In the context of the DBNA extension, the cost function $f(c)$ plays a pivotal role in determining the number of users x^t that enter the system based on the amount c^t the algorithm decides to "pay." Unlike Departing bandits problem, where the number of participants is fixed, the inclusion of a cost function creates a dynamic environment where the learner must balance the marginal value of acquiring additional users with optimizing the recommendation policy π_t. For example, in a subscription-based service like Netflix, c^t could represent the investment in targeted marketing campaigns designed to attract new subscribers. The platform must decide how much to spend on these campaigns while considering the additional revenue generated by retaining users through personalized recommendations. The goal is to ensure that the cost of acquisition, in our case c^t, does not outweigh the potential long-term value derived from these new users, in our case x^t.

As the number of users x increases, the departure-probability matrix $\Lambda(x)$ decreases, leading to diminishing returns from adding users relative to the cost incurred. Simultaneously, the choice of c^t directly influences the number of users x^t, which in turn affects the expected return of the policy π_t. Thus, optimizing c^t is inherently coupled with selecting the optimal policy, creating an additional layer of complexity in decision-making.

In our extension, we do not aim to formally prove the existence of an optimal policy as such a proof is beyond the scope of this project. The interdependence between c^t, x^t, and π_t creates a highly non-trivial optimization problem.

To illustrate this complexity, we will provide an example that demonstrates how different investment levels c^t can influence both the number of users x^t entering the system and the expected return from the recommendation policy π_t. Specifically, we aim to show that the reward is not necessarily monotonic with respect to the increasing cost c. This highlights that simply increasing the investment does not always lead to proportional gains in rewards, emphasizing the need for careful optimization of both c and π_t to achieve long-term system objectives.

Example

This example aims to demonstrate that increasing c^t does not always lead to the best return due to diminishing returns.

Let us consider a single user type and two categories. Note that our recommendations are with respect to a user type (the hidden state of the user), so in a single user type, all users will receive the same recommendation. For simplicity, in our example, we consider a fixed-arm policy. Furthermore, all users depart at the same phase. The cost function, the click probability matrix, and the departure probability matrix are as follows:

$$f(c^t) = \lceil \ln(c^t) \rceil, \quad P = \begin{bmatrix} 0.3 \\ 0.7 \end{bmatrix}, \quad \Lambda(x) = \begin{bmatrix} \max(0.01, 0.2^{\ln(x)+1}) \\ \max(0.01, 0.8^{\ln(x)+1}) \end{bmatrix} \quad (\epsilon = 0.01)$$

The Cost function $f(c) = \lceil \ln(c) \rceil$ reflects diminishing returns, where initial investments attract users efficiently, but further spending yields smaller gains. This aligns with real-world scenarios, such as marketing, where higher budgets lead to slower growth in user acquisition.

The departure probability function is designed to capture the effect of user retention as the number of users increases. The use of the $\ln(x)$ in the departure-probability formulation makes sense as it models diminishing effects, where initial user increases significantly boost retention, but additional users have progressively less impact. This reflects real-world network effects, where early growth is crucial, but further expansion yields smaller retention gains. Additionally, we impose a lower bound of 0.01 to ensure a nonzero departure probability, preventing unrealistic cases of absolute retention. This structure aligns with real-world settings such as online communities, where a critical mass of users sustains engagement, but departures still occur at a minimum rate.

The results for the selected c^t values are summarized in the table below:

c^t	x^t	V
2	1	0.92
50	4	57.16
1,000	7	−399.99

Detailed Calculations For cost Function Example (Appendix)

The main observation from this example is that increasing c^t isn't always optimal: Raising c^t from 2 to 50 results in a significant increase in return, while raising c^t from 50 to 1,000 leads to a decrease in return. This demonstrates that increasing c^t does not necessarily yield higher returns.

Moreover, since this behavior is observed in a specific instance, it implies that such a phenomenon can occur in the general case, meaning that there exists no universal guarantee that a higher c^t will always lead to better outcomes.

3.5 Conclusion and Future Work

In this work, we introduced the Departing Bandits with Network Effect (DBNA), an extension to the Departing Bandits framework. This extension incorporates a new level of complexity by allowing departure probabilities to depend on the number of users and by enabling the algorithm to determine how many users to "pay" for. Through our analysis, we demonstrated that the inclusion of these dynamics leads to significant changes in the behavior of the system.

The findings underscore the need to go beyond traditional bandit formulations and address the intricacies of user interactions in real-world systems and the "cost" of those users. While our results provide insights into these dynamics, they also reveal the challenges posed by the additional complexity introduced in the DBNA framework.

While this work lays the groundwork for understanding the dynamics of DBNA, several avenues remain open for exploration:

- **Cost Function Analysis:** A more rigorous exploration of the interplay between the cost function $f(c)$, the number of users x, and the policy π is needed. Developing formal proofs for the existence and structure of an optimal policy under the influence of a cost function is an essential next step.
- **Learning Algorithm Improvements:** Extending existing learning algorithms, such as UCB-based approaches, to handle the dynamic nature of departure probabilities and cost functions. Investigating whether the regret bounds of traditional bandit algorithms or departing bandits problem can still be achieved in this extended framework is a critical area for study.
- **General User Types:** Moving beyond the single user type assumption to analyze scenarios with multiple user types and their corresponding interactions. This would allow us to model more complex environments that better reflect real-world systems.
- **Simulation and Real-World Applications:** Implementing the DBNA model in simulations or real-world applications, such as recommender systems, to validate theoretical findings and provide empirical insights.

4 Conclusion

This project builds on the foundation of departing bandits, shedding light on both the theoretical and practical dimensions of dynamic decision-making under uncertainty. By thoroughly analyzing and implementing the methodologies presented in the original framework, we not only validated its significance but also extended its scope through the development of the "Departing Bandits with Network Effects" (DBNA). This extension introduces cost-based user acquisition strategies and group dynamics, providing a more realistic and versatile model for real-world applications.

Important Findings of the Article

The original article presented an innovative approach to addressing user attrition in recommender systems. Key findings include:

- The introduction of a departing bandits framework.
- Sub-linear regret performance across single and two-type user settings, both in the planning and learning problems, achieved through efficient algorithms.
- Theoretical robustness demonstrated via regret analysis and dynamic optimization.
- Practical relevance for real-world applications requiring dynamic adaptation to user behaviors.

Future Work

Building on these contributions, several promising directions for future exploration emerge:

- **User Diversity:** Expanding the framework to handle heterogeneous user types for broader applicability.
- **Practical Validation:** Testing and refining the extended framework in operational environments like e-commerce and content platforms.

This project highlights the transformative potential of integrating theory and practice to solve complex decision-making problems. By expanding the departing bandits framework, we uncovered new opportunities for optimizing user-centric systems in dynamic environments. The work was both challenging and rewarding, offering a deeper appreciation for the nuances of sequential decision-making and its real-world implications.

References

Omer Ben-Porat, Lee Cohen, Liu Leqi, Zachary C. Lipton, and Yishay Mansour. 2022. Modeling Attrition in Recommender Systems with Departing Bandits. (3 2022). http://arxiv.org/abs/2203.13423

Christopher Hillar and Andre Wibisono. 2018. Maximum entropy distributions on graphs. arXiv:1301.3321 [math.ST] https://arxiv.org/abs/1301.3321

Huiwen Jia, Cong Shi, and Siqian Shen. 2021. Multi-armed bandit with sub-exponential rewards. *Operations Research Letters* 49 (09 2021), 728–733. https://doi.org/10.1016/j.orl.2021.08.004

A Appendix

Detailed Calculations For cost Function Example

The Example Setup:

$$f(c^t) = \lceil \ln(c^t) \rceil, \quad P = \begin{bmatrix} 0.3 \\ 0.7 \end{bmatrix}, \quad \Lambda(x) = \begin{bmatrix} \max(0.01, 0.2^{\ln(x)+1}) \\ \max(0.01, 0.8^{\ln(x)+1}) \end{bmatrix} \quad (\epsilon = 0.01)$$

The return is calculated as:

$$E[V^\pi] = E\left[\sum_{x=1}^{x^\pi} \sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x)) - c^t\right],$$

Using the linearity of expectation and noting that all users receive the same recommendation and share the same type, this formula can be simplified to:

$$E[V^\pi] = x^t \cdot E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] - c^t.$$

In our example, since all users depart at the same phase and their decisions are independent, we can compute:

$$E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] = \frac{P}{\Lambda(x^t) \cdot (1-P)},$$

Case 1: $c^t = 2$

- Calculate x^t:
$$x^t = f(c^t) = \lceil \ln(c^\pi) \rceil = \lceil \ln(2) \rceil = \lceil 0.693 \rceil = 1.$$
- Departure probabilities:
$$\Lambda_1(1) = 0.2^{1+\ln(1)} = 0.2^1 = 0.2, \quad \Lambda_2(1) = 0.8^{1+\ln(1)} = 0.8^1 = 0.8.$$
- Expected rewards:
$$E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] = \frac{P_1}{\Lambda_1(x^t) \cdot (1-P_1)} = \frac{0.3}{0.2 \cdot (1-0.3)} = \frac{0.3}{0.2 \cdot 0.7} = \frac{0.3}{0.14} \approx 2.14,$$
$$E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] = \frac{P_2}{\Lambda_2(x^t) \cdot (1-P_2)} = \frac{0.7}{0.8 \cdot (1-0.7)} = \frac{0.7}{0.8 \cdot 0.3} = \frac{0.7}{0.24} \approx 2.92.$$
- Best Return for category 2:
$$E[V^\pi] = x^t \cdot E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] - c^t = 1 \cdot 2.92 - 2 = 0.92.$$

Case 2: $c^t = 50$

- Calculate x^t:
$$x^t = f(c^\pi) = \lceil \ln(c^\pi) \rceil = \lceil \ln(50) \rceil = \lceil 3.91 \rceil = 4.$$
- Departure probabilities:
$$\Lambda_1(4) = 0.2^{1+\ln(4)} = 0.2^{1+1.386} = 0.2^{2.386} \approx 0.016,$$
$$\Lambda_2(4) = 0.8^{1+\ln(4)} = 0.8^{1+1.386} = 0.8^{2.386} \approx 0.489.$$

- Expected rewards:

$$E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] = \frac{P_1}{\Lambda_1(x^t)\cdot(1-P_1)} = \frac{0.3}{0.016\cdot(1-0.3)} = \frac{0.3}{0.016\cdot 0.7} = \frac{0.3}{0.0112} \approx 26.79,$$

$$E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] = \frac{P_2}{\Lambda_2(x^t)\cdot(1-P_2)} = \frac{0.7}{0.489\cdot(1-0.7)} = \frac{0.7}{0.489\cdot 0.3} = \frac{0.7}{0.1467} \approx 4.77.$$

- Best Return for category 1:

$$E[V^\pi] = x^t \cdot E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] - c^t = 4\cdot 26.79 - 50 = 107.16 - 50 = 57.16.$$

Case 3: $c^t = 1,000$

- Calculate x^t:

$$x^t = f(c^\pi) = \lceil \ln(c^\pi) \rceil = \lceil \ln(1,000) \rceil = \lceil 6.91 \rceil = 7.$$

- Departure probabilities:

$$\Lambda_1(7) = 0.2^{1+\ln(7)} = 0.2^{1+1.945} = 0.2^{2.945} \approx 0.005,$$

$$\Lambda_2(7) = 0.8^{1+\ln(7)} = 0.8^{1+1.945} = 0.8^{2.945} \approx 0.358.$$

- Expected rewards:

$$E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] = \frac{P_1}{\Lambda_1(x^t)\cdot(1-P_1)} = \frac{0.3}{0.005\cdot(1-0.3)} = \frac{0.3}{0.005\cdot 0.7} = \frac{0.3}{0.0035} \approx 85.71,$$

$$E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] = \frac{P_2}{\Lambda_2(x^t)\cdot(1-P_2)} = \frac{0.7}{0.358\cdot(1-0.7)} = \frac{0.7}{0.358\cdot 0.3} = \frac{0.7}{0.1074} \approx 6.52.$$

- Best Return for category 1:

$$E[V^\pi] = x^t \cdot E\left[\sum_{r=1}^{N^{\pi_t}(t)} reward_{t,r,x}(\pi_{t,r}(x))\right] - c^t = 7\cdot 85.71 - 1,000 = 599.97 - 1,000 = -400.03.$$