Supplementary Material: Learning User Preferences in Non-Stationary Environments

A Typical Values of the Parameters in Assumptions A1–A3

In this section, we provide two examples for which we derive the typical values of the various parameters in Assumptions **A1**–**A3**.

Example 2. *Consider the noiseless case where $\Delta = 1/2$. In this case, the users' ratings are deterministic given their user-types. Accordingly we generate K d-dimensional binary vectors $\{\mathbf{b}_i\}_{i=1}^{\mathsf{K}}$ by randomly drawing d statistically independent $\mathsf{Bernoulli}(1/2)$ random variables, for each user-type. Here $\mathsf{d} \leq \mathsf{M}$ is some parameter. Then, the preference vector of any user in the ℓ user-type (i.e., $\mathcal{T}_\ell$) will be the concatenation of $\mathbf{b}_\ell$ with $\mathsf{M}-\mathsf{d}$ statistically independent $\mathsf{Bernoulli}(1/2)$ random variables. To wit, the preference vector of user $u \in \mathcal{T}_\ell$ is $\mathbf{p}_u = [\mathbf{b}_\ell; \mathbf{e}_u]$, where $\mathbf{e}_u$ is a binary vector whose $\mathsf{M}-\mathsf{d}$ elements are statistically independent $\mathsf{Bernoulli}(1/2)$ random variables. Now, for any two users u and v from different user-types, it should be clear that the inner product $\frac{1}{\mathsf{M}}\langle 2\mathbf{p}_u - \mathbf{1}, 2\mathbf{p}_v - \mathbf{1}\rangle$ is merely a sum of M Rademacher random variables normalized by M. Accordingly, a standard concentration inequality on sum of Rademacher random variables tell us that the value of this inner product is in the interval $[-\Theta(\sqrt{\frac{\log \mathsf{M}}{\mathsf{M}}}), \Theta(\sqrt{\frac{\log \mathsf{M}}{\mathsf{M}}})]$, with probability at least $1 - \mathsf{poly}(\mathsf{M}^{-1})$. Therefore, for the incoherence condition to hold with high probability we need $\gamma_1 > \Theta(\sqrt{\frac{\log \mathsf{M}}{\mathsf{M}}})$. On the other hand, if u and v are from the same user-type, the inner product of the first d items is maximal (i.e., unity) by construction. Therefore, using the same arguments it can be shown that the value of the above inner product is at least $\frac{\mathsf{d}}{\mathsf{M}} - \Theta(\sqrt{\frac{(\mathsf{M}-\mathsf{d})\log(\mathsf{M}-\mathsf{d})}{\mathsf{M}^2}}) \geq \frac{\mathsf{d}}{\mathsf{M}} - \Theta(\sqrt{\frac{\log \mathsf{M}}{\mathsf{M}}})$, with high probability. This implies that the coherence condition holds if $\gamma_2 \leq \frac{\mathsf{d}}{\mathsf{M}} - \Theta(\sqrt{\frac{(\mathsf{M}-\mathsf{d})\log \mathsf{M}}{\mathsf{M}^2}})$. When $\mathsf{d} = \mathsf{M}$, which means that users of the same user-type have exactly the same preference vectors and therefore γ_2 can get as large as 1. Otherwise, there is a certain payment depending on how similar the preference vectors are, controlled by d. Finally, the typical value of μ is clearly around $1/2$ with high probability.*

Example 3. *We generalize the previous example. Consider the case where each entry of the d-dimensional vectors $\{\mathbf{b}_\ell\}_{\ell=1}^{\mathsf{K}}$ is $\frac{1}{2} + \Delta$ with probability μ and $\frac{1}{2} - \Delta$ with probability $1 - \mu$, for a fixed Δ. Then, as in the previous example, the preference vector of user $u \in \mathcal{T}_\ell$ is $\mathbf{p}_u = [\mathbf{b}_\ell; \mathbf{e}_u]$, where $\mathbf{e}_u$ is now a random vector whose $\mathsf{M}-\mathsf{d}$ elements are statistically independent, and each element is either $\frac{1}{2} + \Delta$ with probability μ and $\frac{1}{2} - \Delta$ with probability $1 - \mu$. Then, using the same arguments as in the previous example, it can be shown that if users u and v are of different user types, then the incoherence condition holds with high probability when $\gamma_1 > (1-2\mu)^2 + \Theta(\sqrt{\frac{\log \mathsf{M}}{\mathsf{M}}})$. On the other hand, if users u and v are of the same user type, then the coherence condition holds with high probability when $\gamma_2 \leq \frac{\mathsf{d}}{\mathsf{M}} - (1-2\mu)^2 - \Theta(\sqrt{\frac{(\mathsf{M}-\mathsf{d})\log(\mathsf{M}-\mathsf{d})}{\mathsf{M}^2}})$.*

B Proof of Theorem 1

To prove Theorem 1, we establish first a few accompanying results. We start with the following lemma which bounds the probability that user of different (same) type have the same response. For this lemma, we assume that users *cannot* change their type over time, and denote the type of user $u \in [\mathsf{N}]$ by $\mathcal{T}_u$.

Lemma 1 (Same Response Lemma)**.** *Consider the latent source model and the incoherence Assumption **A3**. Let ℓ be an item chosen uniformly at random from $[\mathsf{M}]$. Then, the probability that two users u and v rate ℓ in the same way is:*

$$\mathbb{P}\left[\mathbf{R}_{u\ell} = \mathbf{R}_{v\ell} | \mathcal{T}_u \neq \mathcal{T}_v\right] \leq 2\gamma_1\Delta^2 + \frac{1}{2}, \tag{6}$$

for users of different types, and,

$$\mathbb{P}\left[\mathbf{R}_{u\ell} = \mathbf{R}_{v\ell} | \mathcal{T}_u = \mathcal{T}_v\right] \geq 2\gamma_2\Delta^2 + \frac{1}{2}, \tag{7}$$

for users of the same type.

Proof. Notice that for two users u and v belonging to different user groups, the probability in question is

$$\begin{aligned}\mathbb{P}\left[\mathbf{R}_{u\ell} = \mathbf{R}_{v\ell} | \mathcal{T}_u \neq \mathcal{T}_v\right] &= \frac{1}{\mathsf{M}}\sum_{i=1}^{\mathsf{M}}\left[p_{u,i}p_{v,i} + (1-p_{u,i})(1-p_{v,i})\right]\\ &= \frac{1}{\mathsf{M}}\sum_{i=1}^{\mathsf{M}}\left[\frac{(2p_{u,i}-1)(2p_{v,i}-1)}{2} + \frac{1}{2}\right]\\ &= \frac{1}{\mathsf{M}}\langle 2\mathbf{p}_u - \mathbf{1}, 2\mathbf{p}_v - \mathbf{1}\rangle + \frac{1}{2}\\ &\leq 2\gamma_1\Delta^2 + \frac{1}{2},\end{aligned}$$

where the inequality follows from the incoherence Assumption **A3**. Similarly, for two users of the same type,

$$\begin{aligned}\mathbb{P}\left[\mathbf{R}_{u\ell} = \mathbf{R}_{v\ell} | \mathcal{T}_u = \mathcal{T}_v\right] &= \frac{1}{\mathsf{M}}\langle 2\mathbf{p}_u - \mathbf{1}, 2\mathbf{p}_v - \mathbf{1}\rangle + \frac{1}{2}\\ &\geq 2\gamma_2\Delta^2 + \frac{1}{2},\end{aligned}$$

where, again, the inequality follows from the coherence Assumption **A3**. □

The following lemma gives a condition on the number of random recommendations needed for the cosine-similarity test to output the correct clustering with high probability, assuming that no variations happened during the test. We establish a few notations. Let $\mathcal{T}_{\mathsf{test}} \subseteq [\mathsf{M}]$ be a set of L items chosen uniformly at random from M. Let $\mathsf{Y}_{u,v} \in \{0,1\}$ be a binary variable indicating whether (u,v) are in the same cluster or not, for $u,v \in [\mathsf{N}]$. Using the responses $\{\mathbf{R}_{u,i}\}_{u\in[\mathsf{N}], i\in\mathcal{T}_{\mathsf{test}}}$, we would like to infer the values of $\mathsf{Y}_{u,v}$ for all $u,v \in [\mathsf{N}]$. For any pair of distinct users $u,v \in [\mathsf{N}]$, let $\mathsf{X}_{u,v}$ be the random variable corresponding to the number of items for which u and v had the same responses. Finally, we let

$$\hat{\mathsf{Y}}_{u,v} = \begin{cases}1, & \text{if } \mathsf{X}_{u,v} \geq \lambda\cdot\mathsf{L}\\ 0, & \text{otherwise,}\end{cases} \tag{8}$$

for some $\lambda \geq 0$. We have the following result.

Lemma 2. *Consider the latent source model and the incoherence Assumption **A3**. Let $\delta \in (0,1)$. For any $\mathsf{L} \geq \frac{2\log(3\mathsf{N}^2/\delta)}{\Delta^4(\gamma_2-\gamma_1)^2} \triangleq \mathsf{T}_{\mathsf{static}}$, and any $\lambda \in [\lambda_-, \lambda_+]$ with $\lambda_- = 2\gamma_1\Delta^2 + \frac{1}{2} + \sqrt{\frac{2}{\mathsf{L}}\log(3\mathsf{N}^2/\delta)}$ and $\lambda_+ = 2\gamma_2\Delta^2 + \frac{1}{2} - \sqrt{\frac{2}{\mathsf{L}}\log(3\mathsf{N}^2/\delta)}$, the test in (8) discriminates between $\mathsf{Y}_{u,v} = 0$ and $\mathsf{Y}_{u,v} = 1$, for any pair of users $u,v \in [\mathsf{N}]$, with probability at least $1-\delta/3$.*

Proof. First, it is clear that Lemma 1 implies that

$$\begin{aligned}\mathbb{E}\left[\mathsf{X}_{u,v} | \mathsf{Y}_{u,v} = 0\right] &\leq \mathsf{L}\Big(2\gamma_1\Delta^2 + \frac{1}{2}\Big),\\ \mathbb{E}\left[\mathsf{X}_{u,v} | \mathsf{Y}_{u,v} = 1\right] &\geq \mathsf{L}\Big(2\gamma_2\Delta^2 + \frac{1}{2}\Big).\end{aligned}$$

Then, we note that $\mathsf{X}_{u,v}$ is a sum of L random variables in $[-1,1]$, drawn without replacement from $[\mathsf{M}]$. Accordingly, Hoeffding's inequality gives,

$$\begin{aligned}\mathbb{P}\left[\mathsf{X}_{u,v} \geq \lambda_-\cdot\mathsf{L} | \, \mathsf{Y}_{u,v} = 0\right] &\leq \exp\left[-\frac{\left(\lambda_-\cdot\mathsf{L} - \mathbb{E}\left[\mathsf{X}_{u,v}|\mathsf{Y}_{u,v} = 0\right]\right)^2}{2\mathsf{L}}\right] &(9)\\ &\leq \exp\left[-\frac{\left(\lambda_- - 2\gamma_1\Delta^2 - \frac{1}{2}\right)^2}{2}\mathsf{L}\right], &(10)\end{aligned}$$

and

$$\mathbb{P}\left[\mathsf{X}_{u,v} \leq \lambda_+\cdot\mathsf{L} | \, \mathsf{Y}_{u,v} = 1\right] \leq \exp\left[-\frac{\left(2\gamma_2\Delta^2 + \frac{1}{2} - \lambda_+\right)^2}{2}\mathsf{L}\right]. \tag{11}$$

Therefore, taking $\lambda_- = 2\gamma_1\Delta^2 + \frac{1}{2} + \sqrt{\frac{2}{\mathsf{L}}\log(3\mathsf{N}^2/\delta)}$ and $\lambda_+ = 2\gamma_2\Delta^2 + \frac{1}{2} - \sqrt{\frac{2}{\mathsf{L}}\log(3\mathsf{N}^2/\delta)}$, we obtain that

$$\mathbb{P}\left[\mathsf{X}_{u,v} \geq \lambda_- \cdot \mathsf{L} \middle| \mathsf{Y}_{u,v} = 0\right] \leq \frac{\delta}{3\mathsf{N}^2}, \tag{12}$$

and

$$\mathbb{P}\left[\mathsf{X}_{u,v} \leq \lambda_+ \cdot \mathsf{L} \middle| \mathsf{Y}_{u,v} = 1\right] \leq \frac{\delta}{3\mathsf{N}^2}. \tag{13}$$

Picking any $\lambda \in [\lambda_-, \lambda_+]$, we can see that the bounds in (12)–(13), with λ_- and λ_+ replaced by λ. This is equivalent to $\mathbb{P}\left[\hat{\mathsf{Y}}_{u,v} \neq \mathsf{Y}_{u,v} \middle| \mathsf{Y}_{u,v} = \ell\right] \leq \delta/(3\mathsf{N}^2)$, for $\ell = 0, 1$. Such λ exists if $\lambda_+ \geq \lambda_-$, which holds whenever,

$$\mathsf{L} \geq \frac{2\log(3\mathsf{N}^2/\delta)}{\Delta^4(\gamma_2 - \gamma_1)^2} = \mathsf{T}_{\mathsf{static}}. \tag{14}$$

Finally, taking a union bound over all pairs of users (we trivially have at most N^2 such pairs) we conclude that we can correctly infer the values of $\mathsf{Y}_{u,v}$ for all $u, v \in [\mathsf{N}]$ (and therefore cluster all such pairs of users correctly), with probability at least $1 - \delta/3$, as claimed. □

We would like to mention here that the test described above can only distinguish between whether $\mathsf{Y}_{u,v} = 1$ or $\mathsf{Y}_{u,v} = 0$, *assuming* that users did not change their type during the test. If, however, a test is conducted when there are switches, we can still infer the clustering of those users who have not changed during the test correctly.

We are now in a position to prove Theorem 1. With some abuse of notation, let us denote by $\mathsf{reward}(\mathcal{B}_\ell)$ the expected reward accumulated in batch $\mathcal{B}_\ell$, i.e.,

$$\begin{aligned}\mathsf{reward}(\mathcal{B}_\ell) &\triangleq \mathbb{E}\left[\sum_{t\in\mathcal{B}_\ell}\frac{1}{\mathsf{N}}\sum_{u=1}^{\mathsf{N}}\mathbb{1}[\mathbf{R}_{u\pi_{u,t}} = 1]\right] \\ &= |\mathcal{B}_\ell| - \mathbb{E}\left[\sum_{t\in\mathcal{B}_\ell}\frac{1}{\mathsf{N}}\sum_{u=1}^{\mathsf{N}}\mathbb{1}[\mathbf{R}_{u\pi_{u,t}} = 0]\right] \\ &\triangleq |\mathcal{B}_\ell| - \mathsf{regret}(\mathcal{B}_\ell),\end{aligned} \tag{15}$$

where $\mathsf{regret}(\mathcal{B}_\ell)$ is the regret accumulated during batch $\mathcal{B}_\ell$. As can be seen from Algorithm Collaborative, we decompose the recommendation horizon T to a sequence of batches of size Δ_{T} each. To obtain Theorem 1, we will relate the total reward/regret with the local reward/regret of the static algorithm Recommend. Specifically, let $\mathcal{B}_\ell$, for $\ell = 1, 2, \ldots, \lceil \mathsf{T}/\Delta_{\mathsf{T}}\rceil$, denote the ℓ'th batch of size Δ_{T}, and let $t_{\mathcal{B}_\ell}$ be the ending time of batch $\mathcal{B}_\ell$. We will keep track of a set of users $\mathcal{V}_t \subseteq [\mathsf{N}]$ which will include all those users for whom we have been able to identify that they have have changed their user groups at some point of time during the batch $\mathcal{B}_\ell$. We initialize $\mathcal{V}_{t_{\mathcal{B}_{\ell-1}}+1} = \phi$ at the beginning of the batch to be the empty set. We define

$$\mathsf{V}_{\mathcal{B}_\ell,1} \triangleq \sum_{t\in\mathcal{B}_\ell\setminus t_{\mathcal{B}_\ell}} \mathbb{1}\left[\mathcal{T}_u(t) \neq \mathcal{T}_u(t+1),\ \text{for some } u \in [\mathsf{N}]\right], \tag{16}$$

as the number of variations that have occurred during the batch $\mathcal{B}_\ell$. Furthermore, we let

$$\mathsf{V}_{\mathcal{B}_\ell,2} \triangleq \frac{1}{\mathsf{N}}\sum_{u\in[N]}\sum_{t\in\mathcal{B}_\ell\setminus t_{\mathcal{B}_\ell}} \mathbb{1}\left[\mathcal{T}_u(t) \neq \mathcal{T}_u(t+1)\right], \tag{17}$$

as the total number of variations that have occurred during the batch $\mathcal{B}_\ell$. For $\tau \in \mathcal{B}_\ell$, we define Z_τ to be an indicator random variable which is unity if some user switches its type in a window of $2 \cdot \mathsf{T}_{\mathsf{static}}$ around round τ within the batch $\mathcal{B}_\ell$. For $\tau \in \mathcal{B}_\ell$, let us denote $\mathsf{W}_\tau := \{\max\{\tau - \mathsf{T}_{\mathsf{static}}, t_{\mathcal{B}_{\ell-1}+1}\}, \ldots, \min\{\tau + \mathsf{T}_{\mathsf{static}}, t_{\mathcal{B}_\ell}\}\}$ as the window of size $2 \cdot \mathsf{T}_{\mathsf{static}}$ around τ. Then, note that Z_τ can be written as

$$\mathsf{Z}_\tau = \mathbb{1}\left[\sum_{u\in[\mathsf{N}]}\sum_{t\in\mathsf{W}_\tau} \mathbb{1}\left[\mathcal{T}_u(t) \neq \mathcal{T}_u(t+1)\right] > 0\right]. \tag{18}$$

As can be seen from Algorithm RECOMMEND, at every round in each batch, we start a test with probability $1/\sqrt{\Delta_{\mathsf{T}}}$, which involves recommending randomly sampled items to every user for $\mathsf{T}_{\mathsf{static}}$ rounds. After each such test, we can use Lemma 2 to partition the set of users. In addition, in the fourth step of Algorithm RECOMMEND we conduct a *reference test* at the beginning of the batch. In the sequel, we denote this test by Test_0, and further denote the $(j+1)^{\mathsf{th}}$ test by Test_j. The partition induced by the $(j+1)^{\mathsf{th}}$ test is denoted by $\mathcal{P}_{\mathsf{Test}_j}$. By comparing the partitions $\mathcal{P}_{\mathsf{Test}_j}$ and $\mathcal{P}_{\mathsf{Test}_0}$, we will be able to partially identify users who have changed their user groups in the batch. This is done in Algorithm TEST. We will call those users who have changed their user groups in a particular batch as *bad* users and those users who have not changed their user groups throughout the batch as *good* users. Moreover, a user is also *good* until he changes his user group and will be denoted as *bad* from the round he changes his group. In order to bound the regret over each batch we will consider the following three cases:

- **Case 1:** Consider the situation where at least 2/3 of the users of any particular user group have changed their user group. We denote this event by $\mathcal{E}_1$. In such a case, we will upper bound the regret in the batch $\mathcal{B}_\ell$ by Δ_T. Notice that, since $\mathsf{V}_{\mathcal{B}_\ell,2} \geq \frac{2\nu}{3}$, therefore conditioned on $\mathcal{E}_1$, we have $\mathsf{regret}(\mathcal{B}_\ell) \leq \frac{3\Delta_{\mathsf{T}}\mathsf{V}_{\mathcal{B}_\ell,2}}{2\nu}$.
- **Case 2:** In this case, we will assume that for every user group, at most 1/3 of the users change their user groups in the batch. For any test $\mathcal{P}_{\mathsf{Test}_j}$, notice that we can actually end up with more than K clusters (say we have K' clusters) because of variations. In that case, we will identify all the users in the smallest $\mathsf{K}'-\mathsf{K}$ clusters as users who have changed their user group. Note that, it is possible that we make a mistake in this process because one of the clusters in the smallest $\mathsf{K}'-\mathsf{K}$ clusters might correspond to good users who have not changed their user group. This, however, must mean that a larger cluster among the largest K clusters must correspond to users who have changed. This in turn implies that at least $\frac{2\nu\mathsf{N}}{3}$ users have changed since the size of the smaller cluster corresponding to users who do not change their group throughout the batch $\mathcal{B}_\ell$ is at least $\frac{2\nu\mathsf{N}}{3}$. We will denote this event by $\mathcal{E}_2$. As in the previous case, we trivially upper bound the regret in the batch $\mathcal{B}_\ell$ by Δ_{T}, and similarly to Case 1, we have $\mathsf{regret}(\mathcal{B}_\ell) \leq \frac{3\Delta_{\mathsf{T}}\mathsf{V}_{\mathcal{B}_\ell,2}}{2\nu}$, conditioned on $\mathcal{E}_2$.
- **Case 3:** In this case, as in the previous case, we assume that for every user group, at most 1/3 of the users change their user groups in the batch. Contrary to the previous case, we will also assume that in every test with more than K clusters (say, K' clusters), the users in the smallest $\mathsf{K}'-\mathsf{K}$ users correspond to users who have changed their user group. For a future test j started at round $\tau_{\mathsf{test},j}$ such that $\mathsf{Z}_{\tau_{\mathsf{test},j},u} = 0$, we will compare the partitions $\mathcal{P}_{\mathsf{Test}_0}$ and $\mathcal{P}_{\mathsf{Test}_j}$ by establishing a bijective mapping between the clusters of the two partitions. For every cluster $\mathcal{C}$ in $\mathcal{P}_{\mathsf{Test}_0}$, we can find a cluster $\mathcal{C}'$ in $\mathcal{P}_{\mathsf{Test}_j}$ such that at least two-thirds of the elements in $\mathcal{C}, \mathcal{C}'$ are common. Subsequently, for all those users in $\mathcal{C}$ which are not present in $\mathcal{C}'$, we correctly identify them as users who have changed their user groups. For a pair of distinct users $(u, v) \subset [\mathsf{N}] \times [\mathsf{N}]$ belonging to the same user group at the beginning of the batch $\mathcal{B}_\ell$, we call them *interesting* if one of them have changed their user group. Note that, for any pair of interesting users (u, v) where one of them have changed their user group at any round after the reference test is conducted and remains in different user group before the $(j+1)^{\mathsf{th}}$ test is started will belong to different clusters in $\mathcal{P}_{\mathsf{Test}_j}$. Note that it is possible that $\mathsf{Z}_{t_{\mathcal{B}_{\ell-1}}+1,u} = 1$, i.e., some user u might change their user group during the first $\mathsf{T}_{\mathsf{static}}$ rounds when the reference test is being conducted. Since we can label the top K clusters (by the corresponding user-group) returned by the reference test as we know that two-thirds users of every user group did not change. We denote by $\mathcal{P}_{\mathsf{Test}_0}(u)$ the cluster (label) u belongs to in the partition returned by the reference test Test_0. Let us define an indicator random variable L_u which is unity if user u has changed his user group in the first $\mathsf{T}_{\mathsf{static}}$ rounds. Consider such a user u for which $\mathsf{L}_u = 1$. In that case, three things are possible at the end of the reference test:

  1. u might belong to the smallest $\mathsf{K}'-\mathsf{K}$ clusters in the reference test in which case u is identified as a user who has changed his user group and he is not involved in the main algorithm started after the reference test, i.e., u is added to the set $\mathcal{V}_{\mathsf{T}_{\mathsf{static}}}$. We define an indicator random variable $\mathsf{X}_{u,1}$ which is unity if user u has changed his user group during the first $\mathsf{T}_{\mathsf{static}}$ rounds in the batch, and is returned in the smallest $\mathsf{K}'-\mathsf{K}$ clusters at the end of the reference test.
  2. u belongs to the cluster corresponding to his new user group in which case we will not be able to infer that u has changed his user group. In this case, we will consider u to be a *good* user unless he changes his user group later. We will consider his user group at the end of the reference test ($\mathcal{P}_{\mathsf{Test}_0}(u)$ which is same as $\mathcal{T}_u(t_{\mathcal{B}_{\ell-1}} + 1 + \mathsf{T}_{\mathsf{static}})$) to be his actual user group. We will call this a *special case* and we

define an indicator random variable $\mathsf{X}_{u,2} \triangleq \mathbb{1}[\mathcal{P}_{\mathsf{test}_0}(u) = \mathcal{T}_u(t_{\mathcal{B}_{\ell-1}} + 1 + \mathsf{T}_{\mathsf{static}})]$ which is unity if user u changes his user group during the first $\mathsf{T}_{\mathsf{static}}$ rounds in the batch, and belongs to his final user group (the user group he belongs to at the end of the reference test).

3. u remains in his original user group (or an intermediate user group if he changes his user group multiple times during the reference test). We define an indicator random variable $\mathsf{X}_{u,3}$ which is unity if the user changes his user group in the first $\mathsf{T}_{\mathsf{static}}$ rounds and does not belong to his final user group (the user group he belongs to at the end of the reference test) at the end of the reference test, i.e., $\mathsf{X}_{u,3} \triangleq \mathbb{1}[\mathcal{P}_{\mathsf{test}_0}(u) \neq \mathcal{T}_u(t_{\mathcal{B}_{\ell-1}} + 1 + \mathsf{T}_{\mathsf{static}})]$. For a round $\tau > \mathsf{T}_{\mathsf{static}}$ in $\mathcal{B}_\ell$, we will define an indicator random variable $\mathsf{J}_{u,\tau} = \mathbb{1}[\mathcal{T}_u(\tau) \neq \mathcal{P}_{\mathsf{Test}_0}(u)]$, which is unity if user u is in a different group at round τ than the user group of u that was returned by the reference test.

We are now in a position to bound the regret over each batch. To that end, we will decompose the regret into a few terms and analyze the contribution of each term separately. First, as we described above conditioned on Cases 1 and 2, namely, $\mathcal{A} \triangleq \mathcal{E}_1 \cup \mathcal{E}_2$ we have

$$\mathsf{regret}(\mathcal{B}_\ell|\mathcal{A}) \triangleq \frac{1}{\mathsf{N}} \sum_{t \in \mathcal{B}_\ell \setminus \mathcal{T}_{\mathsf{test},\ell}} \sum_{u \in [\mathsf{N}]} \mathbb{E}\left[\mathbb{1}\left[\mathbf{R}_{u,\pi_{u,t}} = 0 \mid \mathcal{A}\right]\right] \tag{19}$$

$$\leq \frac{3\Delta_{\mathsf{T}} \mathsf{V}_{\mathcal{B}_\ell,2}}{2\nu}. \tag{20}$$

Next, we analyze Case 3, where we condition on $\mathcal{A}^c$, namely,

$$\mathsf{regret}(\mathcal{B}_\ell|\mathcal{A}^c) \triangleq \frac{1}{\mathsf{N}} \sum_{t \in \mathcal{B}_\ell \setminus \mathcal{T}_{\mathsf{test},\ell}} \sum_{u \in [\mathsf{N}]} \mathbb{E}\left[\mathbb{1}\left[\mathbf{R}_{u,\pi_{u,t}} = 0 \mid \mathcal{A}^c\right]\right]. \tag{21}$$

We do that by considering each of the sub-cases listed above.

B.1 Variations When Testing

We bound the regret for those rounds in the batch for which $\mathsf{Z}_\tau = 1$. Specifically, for a round $\tau \in \mathcal{B}_\ell$, we denote the event $\mathcal{E}_{\tau,1}$ when $\mathsf{Z}_\tau = 1$, which by definition imply that there is a variation in a window of size $2 \cdot \mathsf{T}_{\mathsf{static}}$ around round τ for some user. In particular, using the definitions in (16) and (18), we note that

$$\sum_{\tau \in \mathcal{B}_\ell \setminus \mathcal{T}_{\mathsf{test},\ell}} \mathsf{Z}_\tau \leq \sum_{\tau \in \mathcal{B}_\ell} \sum_{t \in \mathsf{W}_\tau} \mathbb{1}\left[\mathcal{T}_u(t) \neq \mathcal{T}_u(t+1), \text{ for some } u \in [\mathsf{N}]\right] \tag{22}$$

$$\leq 2 \cdot \mathsf{V}_{\mathcal{B}_\ell,1} \cdot \mathsf{T}_{\mathsf{static}}. \tag{23}$$

Therefore, we can bound the regret in those rounds and users where $\mathsf{Z}_\tau = 1$ by

$$\mathsf{A}_2 \triangleq \frac{1}{\mathsf{N}} \mathbb{E}\left[\sum_{t \in \mathcal{B}_\ell \setminus \mathcal{T}_{\mathsf{test},\ell}} \sum_{u \in [\mathsf{N}]:\mathsf{Z}_t = 1} \mathbb{1}\left[\mathbf{R}_{u,\pi_{u,t}} = 0 \mid \mathcal{A}^c\right]\right] \tag{24}$$

$$\leq \mathbb{E}\left[\sum_{\tau \in \mathcal{B}_\ell} \mathbb{1}\left[\mathsf{Z}_t = 1\right]\right] \tag{25}$$

$$\leq 2 \cdot \mathsf{V}_{\mathcal{B}_\ell,1} \cdot \mathsf{T}_{\mathsf{static}}. \tag{26}$$

B.2 Regret Due To Testing

We bound the regret for those rounds where we test in Algorithm Recommend. Specifically, for a round $\tau \in \mathcal{B}_\ell$, we define the indicator random variable Y_τ which is unity when a test is being conducted at the round τ. We have

$$\mathsf{A}_3 \triangleq \frac{1}{\mathsf{N}}\mathbb{E}\left[\sum_{t\in\mathcal{B}_\ell\setminus\mathcal{T}_{\mathsf{test},\ell}:\mathsf{Y}_\tau=1}\sum_{u\in[\mathsf{N}]}\mathbb{1}\left[\mathbf{R}_{u,\pi_{u,t}}=0\mid\mathcal{A}^c\right]\right] \tag{27}$$

$$\leq \mathbb{E}\left[\sum_{t\in\mathcal{B}_\ell}\mathbb{1}\left[\mathsf{Y}_\tau=1\right]\right] \tag{28}$$

$$\leq \Delta_{\mathsf{T}}\cdot p\cdot\mathsf{T}_{\mathsf{static}}, \tag{29}$$

where we have used the fact that $\mathbb{P}[\mathsf{Y}_\tau=1]=\mathbb{P}[\tau\in\text{Test}]=p$, $|\mathcal{B}_\ell|=\Delta_{\mathsf{T}}$, and each test takes $\mathsf{T}_{\mathsf{static}}$ rounds.

B.3 Undetected Bad Users

For a user $u\in[\mathsf{N}]$, we define an indicator random variable $\mathsf{B}_{u,t}$ which is unity if the user is not included in the set of bad users $\mathcal{V}_t$ at round $t\in\mathcal{B}_\ell$. Furthermore, for a round t after the reference test, namely, $t\in\mathcal{B}_\ell\setminus\mathcal{T}_{\mathsf{test},\ell}$, where $\mathcal{T}_{\mathsf{test},\ell}\triangleq[t_{\mathcal{B}_{\ell-1}}+1,\ldots,t_{\mathcal{B}_{\ell-1}}+\mathsf{T}_{\mathsf{static}}]$, define an indicator random variable H_t which is unity if there is a *bad* user which is undetected (or, untested) involved in the algorithm. As we explain Below this random variable can be decomposed into the union of three sub-cases which we discussed above. For any round $t\in\mathcal{B}_\ell\setminus\mathcal{T}_{\mathsf{test},\ell}$, we have:

- A user u who satisfies $\mathsf{L}_u=1,\mathsf{B}_{u,t}=1,\mathsf{X}_{u,3}=1,\mathsf{J}_{u,t}=1$ and $\mathsf{Z}_t=0$, is one who has changed his user group in the first $\mathsf{T}_{\mathsf{static}}$ rounds in the batch, was not in his final user group at the end of the reference test, and his user group at round t is different from his user group that was returned by the reference test, i.e.,

$$\mathcal{T}_u(t_{\mathcal{B}_{\ell-1}}+1+\mathsf{T}_{\mathsf{static}})\neq\mathcal{P}_{\mathsf{Test}_0}(u)\quad\text{and}\quad\mathcal{T}_u(t)\neq\mathcal{P}_{\mathsf{Test}_0}(u).$$

- A user u who satisfies $\mathsf{L}_u=1,\mathsf{B}_{u,t}=1,\mathsf{X}_{u,2}=1,\mathsf{J}_{u,t}=1$ and $\mathsf{Z}_t=0$, is one who has changed his user group in the first $\mathsf{T}_{\mathsf{static}}$ rounds in the batch, and his user group at the end of the reference test is also same as the one provided by the estimate of the reference test, but his user group at round t is different from his user group at the end of the reference test, i.e.,

$$\mathcal{T}_u(t_{\mathcal{B}_{\ell-1}}+1+\mathsf{T}_{\mathsf{static}})=\mathcal{P}_{\mathsf{Test}_0}(u)\quad\text{and}\quad\mathcal{T}_u(t)\neq\mathcal{P}_{\mathsf{Test}_0}(u).$$

- A user u who satisfies $\mathsf{L}_u=0,\mathsf{B}_{u,t}=1,\mathsf{J}_{u,t}=1$ and $\mathsf{Z}_t=0$ is one who has not changed his user group in the first $\mathsf{T}_{\mathsf{static}}$ rounds in the batch, but his user group at round t is different from his user group at the beginning of the batch, i.e.,

$$\begin{aligned}&\mathcal{T}_u(t_{\mathcal{B}_{\ell-1}}+1+\mathsf{T}_{\mathsf{static}})=\mathcal{T}_u(t),\quad\text{for } t\in\mathcal{T}_{\mathsf{test},\ell},\\&\mathcal{T}_u(t)\neq\mathcal{P}_{\mathsf{Test}_0}(u).\end{aligned}$$

Given the above three sub-cases, it is clear that H_t for $t\in\mathcal{B}_\ell\setminus\mathcal{T}_{\mathsf{test},\ell}$, can be written as

$$\begin{aligned}\mathsf{H}_t=\mathbb{1}\Bigg[&\sum_{u\in[\mathsf{N}]}\mathbb{1}\left[\mathsf{L}_u=1,\mathsf{B}_{u,t}=1,\mathsf{X}_{u,3}=1,\mathsf{J}_{u,t}=1,\mathsf{Z}_t=0\right]\\&+\sum_{u\in[\mathsf{N}]}\mathbb{1}\left[\mathsf{L}_u=0,\mathsf{B}_{u,t}=1,\mathsf{J}_{u,t}=1,\mathsf{Z}_t=0\right]\\&+\sum_{u\in[\mathsf{N}]}\mathbb{1}\left[\mathsf{L}_u=1,\mathsf{B}_{u,t}=1,\mathsf{X}_{u,2}=1,\mathsf{J}_{u,t}=1,\mathsf{Z}_t=0\right]>0\Bigg].\end{aligned} \tag{30}$$

Basically, H_t indicates whether at time $t \in \mathcal{B}_\ell \setminus \mathcal{T}_{\mathsf{test},\ell}$ a bad user is present or not. Accordingly, we bound the regret in this case as follows

$$\begin{aligned}
\mathsf{A}_4 &\triangleq \frac{1}{\mathsf{N}}\mathbb{E}\left[\sum_{t\in\mathcal{B}_\ell\setminus\mathcal{T}_{\mathsf{test},\ell}:\mathsf{H}_t=1}\;\sum_{u\in[\mathsf{N}]}\mathbb{1}\left[\mathbf{R}_{u,\pi_{u,t}}=0 \mid \mathcal{A}^c\right]\right] && (31)\\
&\leq \mathbb{E}\sum_{t\in\mathcal{B}_\ell\setminus\mathcal{T}_{\mathsf{test},\ell}}\mathbb{1}\left[\mathsf{H}_t=1\right] && (32)\\
&= \mathbb{E}\sum_{t\in\mathcal{B}_\ell:\exists u\in[\mathsf{N}],\mathsf{J}_{u,t}=1}\mathsf{G}_t, && (33)
\end{aligned}$$

where in (33) we sum over all those rounds where some user changed its type, and G_t counts the number of rounds it takes to detect the bad users. This random variable is clearly stochastically dominated by by a Geometric random variable with mean $1/p$. Indeed, a test can start at every round with probability p, and a test that starts at a round $\mathsf{Z}_t = 0$ will certainly reveal that the user is in a different user group than the one returned in the reference test $\mathcal{P}_{\mathsf{test}_0}$. Accordingly, we will add that user to the set $\mathcal{V}_{t_{\mathcal{B}_{\ell-1}}+1+\mathsf{T}_{\mathsf{static}}+t}$. Therefore, we obtain that,

$$\mathsf{A}_4 \leq \mathbb{E}\sum_{t\in\mathcal{B}_\ell:\exists u\in[\mathsf{N}],\mathsf{J}_{u,t}=1}\mathsf{G}_t \leq \mathbb{E}\sum_{t\in\mathcal{B}_\ell:\exists u\in[\mathsf{N}],\mathsf{J}_{u,t}=1}\frac{1}{p} \leq \frac{\mathsf{V}_{\mathcal{B}_\ell,1}}{p}. \tag{34}$$

B.4 The "Static" Regret

It remains to bound the regret for those round where we do not test and all bad users are detected, i.e.,

$$\mathsf{A}_5 \triangleq \frac{1}{\mathsf{N}}\mathbb{E}\left[\sum_{t\in\mathcal{B}_\ell\setminus\mathcal{T}_{\mathsf{test},\ell}:\mathsf{H}_t=0,\mathsf{Y}_t=0}\;\sum_{u\in[\mathsf{N}]\setminus\mathcal{V}_t}\mathbb{1}\left[\mathbf{R}_{u,\pi_{u,t}}=0 \mid \mathcal{A}^c\right]\right]. \tag{35}$$

We shall refer to this regret as the *static* regret. This static case was studied in [Bresler et al., 2014], where algorithm Recommend was analyzed thoroughly. As discussed before, in [Bresler et al., 2014] it was assumed that users of the same user-type have the same exact preference vectors, while in this paper we assume the weaker coherence Assumption **A3**. Nonetheless, except for a few technical differences (which we highlight in the proof of the following result), our analysis relies on the proof of Theorem 1 in [Bresler et al., 2014].

Theorem 2 (No Variations)**.** *Let $\delta \in (0,1)$, and consider the latent source model and assumptions **A1**–**A3**. Also, assume that $\mathsf{N} = \Omega\left(\frac{\mathsf{M}}{\nu}\log\frac{1}{\delta} + \left(\frac{3}{\delta}\right)^{1/\alpha}\right)$. Then, for any $\mathsf{T}_{\mathsf{static}} \leq \Delta_\mathsf{T} \leq \mu\cdot\mathsf{M}$, we have*

$$\mathsf{A}_5 \leq (\Delta_\mathsf{T} - \mathsf{T}_{\mathsf{static}})\cdot\delta. \tag{36}$$

B.5 Collecting Terms

We finally collect all the above bounds to obtain the result stated in Theorem 1. Specifically, using (20), (26), (29), (34), and Theorem 1, we obtain

$$\begin{aligned}
\mathsf{regret}(\mathcal{B}_\ell) &\leq \mathsf{T}_{\mathsf{static}}\cdot(1-\mu) + (\Delta_\mathsf{T}-\mathsf{T}_{\mathsf{static}})\cdot\delta + 2\cdot\mathsf{V}_{\mathcal{B}_\ell,1}\cdot\mathsf{T}_{\mathsf{static}} + p\cdot\Delta_\mathsf{T}\cdot\mathsf{T}_{\mathsf{static}}\\
&\quad + \frac{\mathsf{V}_{\mathcal{B}_\ell,1}}{p} + \frac{3\Delta_\mathsf{T}\mathsf{V}_{\mathcal{B}_\ell,2}}{2\nu} && (37)\\
&\leq \delta\cdot\Delta_\mathsf{T} + \mathsf{T}_{\mathsf{static}}\cdot(1-\delta-\mu) + 2\cdot\mathsf{V}_{\mathcal{B}_\ell,1}\cdot\mathsf{T}_{\mathsf{static}} + p\cdot\Delta_\mathsf{T}\cdot\mathsf{T}_{\mathsf{static}}\\
&\quad + \frac{\mathsf{V}_{\mathcal{B}_\ell,1}}{p} + \frac{3\Delta_\mathsf{T}\mathsf{V}_{\mathcal{B}_\ell,2}}{2\nu}, && (38)
\end{aligned}$$

where the first term at the r.h.s. of (37) is the regret due to the first $\mathsf{T}_{\mathsf{static}}$ rounds where we recommend random items. Since (38) is true for every batch $\mathcal{B}_\ell$, we can sum-up over ℓ, and obtain that

$$\begin{aligned}
\mathsf{regret}(\mathsf{T}) &\leq \sum_{\ell=1}^{\lceil\mathsf{T}/\Delta_\mathsf{T}\rceil}\mathsf{regret}(\mathcal{B}_\ell) && (39)\\
&\leq \delta\cdot\mathsf{T} + \frac{\mathsf{T}}{\Delta_\mathsf{T}}\mathsf{T}_{\mathsf{static}}\cdot(1-\delta-\mu) + 2\cdot\mathsf{V}_1\cdot\mathsf{T}_{\mathsf{static}} + p\cdot\mathsf{T}\cdot\mathsf{T}_{\mathsf{static}} + \frac{\mathsf{V}_1}{p} + \frac{3\Delta_\mathsf{T}\mathsf{V}_2}{2\nu}. && (40)
\end{aligned}$$

Minimizing the r.h.s. of the above inequality w.r.t. p, we obtain that its optimal value is $p^\star = \sqrt{\mathsf{V}_1/(\mathsf{T}\cdot\mathsf{T}_{\mathsf{static}})}$. Therefore,

$$\mathsf{regret}(\mathsf{T}) \leq \delta\cdot\mathsf{T} + \frac{\mathsf{T}}{\Delta_{\mathsf{T}}}\mathsf{T}_{\mathsf{static}}\cdot(1-\delta-\mu) + 2\cdot\mathsf{V}_1\cdot\mathsf{T}_{\mathsf{static}} + 2\sqrt{\mathsf{V}_1\cdot\mathsf{T}\cdot\mathsf{T}_{\mathsf{static}}} + \frac{3\Delta_{\mathsf{T}}\mathsf{V}_2}{2\nu}. \tag{41}$$

It is left to do is to minimize the r.h.s. of the above inequality over Δ_{T}. The optimal value is given in a form of a solution for a cubic equation. Alternatively, it turns out that the following choice which minimizes the first three terms at the r.h.s. of (41) is

$$\Delta_{\mathsf{T}}^* = \min\left(\mathsf{T}, \sqrt{\frac{2\nu\mathsf{T}}{3\mathsf{V}_2}\kappa}\right), \tag{42}$$

where $\kappa \triangleq \mathsf{T}_{\mathsf{static}}(1-\delta-\mu)$. Substituting this value back in (41) gives

$$\begin{aligned}\mathsf{regret}(\mathsf{T}) &\leq \delta\cdot\mathsf{T} + \max\left(\kappa, \sqrt{\frac{3\mathsf{V}_2\mathsf{T}\kappa}{2\nu}}\right) + 2\cdot\mathsf{V}_1\cdot\mathsf{T}_{\mathsf{static}} + 2\sqrt{\mathsf{V}_1\cdot\mathsf{T}\cdot\mathsf{T}_{\mathsf{static}}}\\ &\quad + \min\left(\frac{3\mathsf{V}_2\mathsf{T}}{2\nu}, \sqrt{\frac{3\mathsf{V}_2\mathsf{T}\kappa}{2\nu}}\right),\end{aligned} \tag{43}$$

and so $\mathsf{reward}(\mathsf{T}) = \mathsf{T} - \mathsf{regret}(\mathsf{T})$ is lower bounded by the same expression as in Theorem 1. Note that the condition $\Delta_{\mathsf{T}} > \mathsf{T}_{\mathsf{static}}$ in Theorem 2 boils down to $\mathsf{T} > \mathsf{T}_{\mathsf{static}}\cdot\max\left\{1, \frac{3\mathsf{V}_2}{2\nu(1-\delta-\mu)}\right\} = \mathsf{T}_{\mathsf{learn}}$. Finally, for $\mathsf{T} \leq \mathsf{T}_{\mathsf{learn}}$, we get that $\mathsf{reward}(\mathsf{T}) \geq \mu\cdot\mathsf{T}$, as claimed.

B.6 Proof of Theorem 2

To prove the result in Theorem 2, it is suffice to lower bound the probability $\mathbb{P}\left[\mathbf{R}_{u\pi_{u,t}} = 1, \mathsf{Y}_t = 0, \mathsf{H}_t = 0\right]$. To that end, for any $u \in [\mathsf{N}]$ and $t \in [\mathsf{T}]$, define

$$\mathcal{G}_{u,t} \triangleq \left\{|\partial_t(u)| \geq \frac{2\nu\mathsf{N}}{3}\right\}, \tag{44}$$

where $\partial_t(u)$ is the set of neighbors at time t user u have from the same user-types, respectively. For t large enough the probability of $\mathcal{G}_{u,t}$ is lower bounded strictly by zero. To show that recall that $|\mathcal{T}_u(t)|$ is the number of users in user's u type at round t. As we argued above at each round we know that $|\mathcal{T}_u(t)| > \frac{2\nu\mathsf{N}}{3}$. Also, recall that in the beginning of the batch we devote the first $\mathsf{T}_{\mathsf{static}}$ recommendations for creating an initial partition $\mathcal{P}_0$ of the users into types (see, the the fourth step in Algorithm 2). We showed in Lemma 2 that the resulted partition is correct with probability at least $1-\delta/3$, and therefore, $|\partial_t(u)|\frac{2\nu\mathsf{N}}{3}$ with the same probability, i.e., $\mathbb{P}[\mathcal{G}_{u,t}] \geq 1-\delta/3$, for $t \in \mathcal{B}_\ell \setminus \mathcal{T}_{\mathsf{test},\ell}$.

Next, using the same steps as in the proof of Lemma 2 in [Bresler et al., 2014], we show that the good neighborhoods have, through random exploration, accurately estimated the probability of liking each item. Thus, we correctly classify each item as likable or not with high probability. In particular, we show Below that

$$\mathbb{P}\left[\mathbf{R}_{u,\pi_{u,t}} = 1, \mathsf{Y}_t = 0, \mathsf{H}_t = 0\middle|\,\mathcal{G}_{u,t}\right] \geq 1 - 2\mathsf{M}\exp\left(-2\frac{\Delta^2\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right) - \frac{1}{\mathsf{N}^\alpha}. \tag{45}$$

Before proving the above inequality let us first show how we can use it lower bound the regret. Indeed, combining the above inequality with the fact that $\mathbb{P}[\mathcal{G}_{u,t}] \geq 1-\delta/3$, we get

$$\mathbb{P}\left[\mathbf{R}_{u,\pi_{u,t}} = 1, \mathsf{Y}_t = 0, \mathsf{H}_t = 0\right] \geq 1 - 2\mathsf{M}\exp\left(-2\frac{\Delta^2\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right) - \frac{1}{\mathsf{N}^\alpha} - \frac{\delta}{3}. \tag{46}$$

It can be seen that if the number of users N satisfy $\mathsf{N} = \Omega\left(\frac{\mathsf{M}}{\nu}\log\frac{1}{\delta} + \left(\frac{3}{\delta}\right)^{1/\alpha}\right)$, and of course $t \geq \mathsf{T}_{\mathsf{static}}$, then the r.h.s. of (46) is at least $1-\delta$, namely, $\mathbb{P}\left[\mathbf{R}_{u,\pi_{u,t}} = 1, \mathsf{Y}_t = 0, \mathsf{H}_t = 0\right] \geq 1-\delta$. Therefore, we obtain,

$$\begin{aligned}\mathsf{A}_5 &\leq \sum_{t\in\mathcal{B}_\ell\setminus\mathcal{T}_{\mathsf{test},\ell}} \frac{1}{\mathsf{N}}\sum_{u=1}^{\mathsf{N}} \mathbb{P}\left[\mathbf{R}_{u,\pi_{u,t}} = 0, \mathsf{Y}_t = 0, \mathsf{H}_t = 0|\mathcal{A}^c\right]\\ &\leq (\Delta_{\mathsf{T}} - \mathsf{T}_{\mathsf{static}})\cdot\delta,\end{aligned} \tag{47}$$

where in the second inequality we have used Assumption **A2**. Next, we prove (45). First, we lower bound the number of times an arbitrary item has been rated by the good neighbors of some user u, conditioned on the event $\mathcal{G}_{u,t}$. To that end, note that the number of good neighbors user u has and who have rated item i is stochastically dominated by $\mathsf{Binomial}\left(\frac{2\nu\mathsf{N}}{3},\frac{t}{\mathsf{M}\mathsf{N}^{\alpha}}\right)$. Let $\mathcal{D}$ be the event "item i has less than $\frac{\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}$ ratings from good neighbors of u". Then, Chernoff's bound then gives

$$\mathbb{P}\left(\mathcal{D}\right)\leq\mathbb{P}\left(\mathsf{Binomial}\left(\frac{2\nu\mathsf{N}}{3},\frac{t}{\mathsf{M}\mathsf{N}^{\alpha}}\right)\leq\frac{\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right) \tag{48}$$

$$\leq\exp\left(-\frac{\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right). \tag{49}$$

Next, conditioned on $\mathcal{G}_{u,t}$ and $\mathcal{D}$ we prove that with high probability when exploiting the algorithm predicts correctly every item as likable or unlikable for user u. Recall our definition for the posterior $\hat{p}_{u\ell}$ in (4). Suppose item i is likeable by user u, and let $\mathsf{G}\triangleq\frac{\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}$. Then, conditioned on G, $\hat{p}_{u\ell}$ stochastically dominates $\tilde{p}_{ui}\triangleq\mathsf{Binomial}(\mathsf{G},p_{ui})/\mathsf{G}$. Then,

$$\mathbb{P}\left(\tilde{p}_{ui}\leq\frac{1}{2}\middle|\,\mathsf{G}\right)=\mathbb{P}\left(\mathsf{Binomial}(\mathsf{G},p_{ui})\leq\frac{\mathsf{G}}{2}\middle|\,\mathsf{G}\right) \tag{50}$$

$$\leq\exp\left(-2\mathsf{G}\Delta^{2}\right) \tag{51}$$

$$\leq\exp\left(-2\frac{\Delta^{2}\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right), \tag{52}$$

where the first inequality follows from Hoeffding's inequality, and the second inequality is because $p_{ui}\geq 1/2+\Delta$. Using monotonicity, we also have

$$\mathbb{P}\left(\tilde{p}_{ui}\leq\frac{1}{2}\middle|\,\mathsf{G}\geq\frac{\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right)\leq\exp\left(-2\frac{\Delta^{2}\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right). \tag{53}$$

Using the same steps we can show that if item i is unlikeable by user u then with the same probability $\tilde{p}_{ui}\geq\frac{1}{2}$. Taking a union bound over all items we get that with probability at least $1-\mathsf{M}\exp\left(-2\frac{\Delta^{2}\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right)$ our algorithm correctly classifies every item as likable or unlikable for user u. We are now in a position to prove (45). Specifically, for user u at time t, conditioned on $\mathcal{G}_{u,t}$ we have shown in (49) that with probability at least $1-\mathsf{M}\exp\left(-\frac{\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right)$ *every* item has more than $\frac{\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}$ ratings from good neighbors of u. Now, using the fact that with probability at least $1-\mathsf{M}\exp\left(-2\frac{\Delta^{2}\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right)$ we classify correctly all items, coupled with the fact that we exploit with probability $1-\mathsf{N}^{-\alpha}$, we get

$$\mathbb{P}\left[\mathbf{R}_{u,\pi_{u,t}}=1,\mathsf{Y}_t=0,\mathsf{H}_t=0\middle|\,\mathcal{G}_{u,t}\right]\geq 1-\mathsf{M}\exp\left(-\frac{\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right)-\mathsf{M}\exp\left(-2\frac{\Delta^{2}\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right)$$

$$-\frac{1}{\mathsf{N}^{\alpha}} \tag{54}$$

$$\geq 1-2\mathsf{M}\exp\left(-2\frac{\Delta^{2}\nu t\mathsf{N}^{1-\alpha}}{3\mathsf{M}}\right)-\frac{1}{\mathsf{N}^{\alpha}}, \tag{55}$$

as claimed.

C Experiments

We simulate an online recommender system using real-world data in order to understand whether our algorithm performs well, even when the data is not generated by the probabilistic model introduced in Section 2. To that end, we follow a similar vein as in [Bresler et al., 2014, Heckel and Ramchandran, 2017], and look at movie ratings from the popular Movielens25m dataset,[1] which provides 5-star rating and free-text tagging activity

[1] https://grouplens.org/datasets/movielens/25m/

from Movielens, a movie recommendation service. We parsed the first 7 million ratings for our experiment, and consider only those users who have rated at least 225 movies, ending up with a total number of $\mathsf{N} = 247$ users.

To avoid any kind of biases, we also restrict ourselves to movies which are more or less equally liked and disliked by the users. To that end, we choose those movies whose average ratings is between 2.5 and 3.5, and we found out that $\mathsf{M} = 10149$ such movies exist. Finally, we looked at two genres: `Action` and `Romance`. For each user $u \in [\mathsf{N}]$, we recover piece-wise stationary preferences by the following steps:

1. We sort the movies rated by user u in ascending order according to the time-stamp.
2. We partition the movies rated by user u into 15 bins so that each bin contains equal number of movies. We will consider each bin to be a window of time.
3. For each bin, we find the number $\mathsf{a}_u \in \mathbb{N}$ of `Action` movies rated by user u, as well as $\mathsf{r}_u \in \mathbb{N}$ the number of `Romance` movies rated by the same user.

Accordingly, note that in each bin, the probability of user u: liking a movie tagged `Action` but not `Romance` is $\mathsf{a}_u/(\mathsf{a}_u + \mathsf{r}_u)$; liking a movie tagged `Romance` but not `Action` is $\mathsf{r}_u/(\mathsf{a}_u + \mathsf{r}_u)$; liking a movie tagged both `Action` and `Romance` is `1`, and finally, a movie which does not have any of these tags is `0`. We want to point out that we consider the number of `Action` and `Romance` movies that were *rated* by the user, rather than just *liked*, since any user is biased towards rating the movies he will like (see, [Heckel and Ramchandran, 2017]), and therefore the number of movies rated by the user is a better indicator of his preference towards the genre. Fig. 3 shows the probability of 5 randomly chosen users liking `Action` movies across 10 different bins. It is clear that the preferences exhibit a piece-wise stationary behaviour, and that the variations are significant.

We now assume for simplicity that the number of rounds in each bin is 100 (this value is unknown to the algorithm), and we took the total number of rounds to be $\mathsf{T} = 600$. In lieu of creating the initial disjoint clusters at the beginning of each batch (i.e., $\mathcal{P}_0$), we recommend $\mathsf{T}_{\mathsf{static}}$ randomly chosen items to all users. For each user $u \in [\mathsf{N}]$, we take the neighbors of u to be the top 10 users whose feedback vector has the highest cosine similarity with that of user u, over the $\mathsf{T}_{\mathsf{static}}$ recommended items. Further, since $\mathsf{T} = 600$ is quite small, we do not test for bad users in each batch (namely, we skip lines $13 - 15$ in Algorithm 2). The reasons for this modification are as follows. First, in the theoretical analysis, we have assumed that ratings of a single *bad user* can potentially result in faulty recommendations for all other users in their user group. However, in practice, that might not be the case as future recommendations are determined by multiple other users who can negate the effect of that *bad* user. Secondly, as the dataset for our experiment is not very large (10 neighbors for each user), detecting bad users based on ratings of neighbors can be unreliable. Finally, for small T, $\mathsf{T}_{\mathsf{static}}$ is comparatively large and therefore testing for *bad users* can potentially bias the accumulated reward towards larger batch-sizes. Nevertheless, as we will show, our experiment clearly demonstrates the dependence on Δ_{T} and $\mathsf{T}_{\mathsf{static}}$ in the non-stationary setting. We run Algorithm 1 with $\mathsf{T}_{\mathsf{static}} = 10$ and $p_{\mathsf{R}} = 0.1$, for several different values of the batch-size Δ_{T}, each for 5 different iterations. The performance of the algorithms is measured in terms of the average cumulative reward up to time T, namely,

$$\mathsf{acc\text{-}reward}(\mathsf{T}) \triangleq \sum_{t \in [\mathsf{T}]} \frac{1}{\mathsf{N}} \sum_{u \in [\mathsf{N}]} \mathbf{R}_{u\pi_{u,t}},$$

where $\pi_{u,t}$ is the item recommended by the algorithm to user u at time t. The average cumulative reward up to time T is given in Table 1. From this table, it is clear that the highest average cumulative reward is obtained when the batch-size is $\Delta_{\mathsf{T}} = 100$, and decreases gradually as the batch-size increases. Finally, not that since we are not detecting *bad users* in our experiments, the knowledge of V_1 is not required (V_1 is only used to set p_{T}). Notice that V_2 is used to set the batch-size Δ_{T} correctly. Since an incorrect value of V_2 results in a sub-optimal value for Δ_{T}, computing the average cumulative reward by iterating through different values of Δ_{T} also gives an idea about the sensitivity of our algorithms with respect to this mis-specification. As can be seen from our results, the highest value of $\mathsf{acc\text{-}reward}(\mathsf{T})$ was achieved when $\Delta_{\mathsf{T}} = 100$, while the $\mathsf{acc\text{-}reward}(\mathsf{T})$ degrades gracefully with the mis-specification of Δ_{T} (or, V_2).

Next, we illustrate the benefit of our algorithm compared to the static algorithm even in a stationary environment. To that end, we run Algorithm 1 with $\Delta_{\mathsf{T}} \in \{100, 600\}$, $\mathsf{T}_{\mathsf{static}} \in \{10, 30, 60, 80, 100\}$, and assume a single bin of size $\mathsf{T} = 600$. Our results are presented in Fig. 4, and perhaps surprisingly, Algorithm 1 with $\Delta_{\mathsf{T}} = 100$ achieves a better accumulated reward compared to $\Delta_{\mathsf{T}} = 600$ (static algorithm), for small values of $\mathsf{T}_{\mathsf{static}}$. The

Δ_{T}	$\mathsf{acc\text{-}reward}(\mathsf{T})$
50	316.707
100	325.716
150	306.538
200	278.219
300	278.642
350	224.893
400	239.410
450	204.127
500	162.96
550	169.97
600	137.40

Table 1: Accumulated reward as a function of the batch-size: $\Delta_{\mathsf{T}} = 600$ corresponds to the static case, and $\Delta_{\mathsf{T}} = 100$ corresponds to the optimal value.

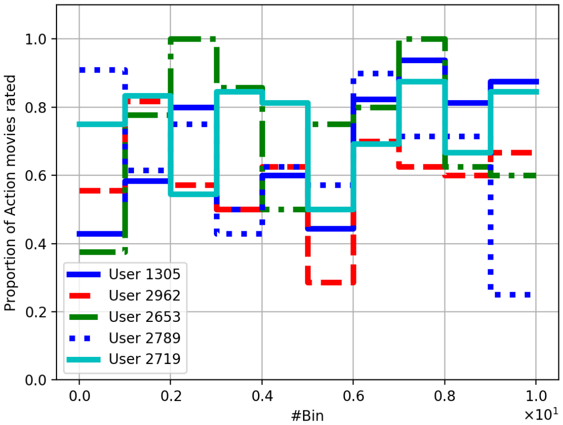

Figure 3: The probability $\mathsf{a}_u/(\mathsf{a}_u + \mathsf{r}_u)$ of user u liking a movie with `Action` tag but not `Romance` tag, for five different users, across 10 different bins/windows.

main reason for this phenomenon is because for Algorithm 1 with $\Delta_{\mathsf{T}} = 600$, the neighbors of any user might not be well chosen due to small values of $\mathsf{T}_{\mathsf{static}}$ because of which the user will receive poor recommendations throughout the entire time frame. On the other hand, running Algorithm 1 with $\Delta_{\mathsf{T}} = 100$ restarts Algorithm 2 at periodic intervals. As a result, the users have a good set of neighbors in some batches and a bad set in others, but the cumulative reward concentrate because the neighbors are independent across the batches. However, the performance of the algorithm with $\Delta_{\mathsf{T}} = 600$ improves as $\mathsf{T}_{\mathsf{static}}$ gets larger since the quality of the estimated neighborhood improves. This experiment hits that it is better to restart the recommendation algorithm periodically, i.e., follow Algorithm 1 (with $\Delta_{\mathsf{T}} < \mathsf{T}$) even in stationary environments. We would like to emphasize that an insufficient number of samples for the initial clustering, results in a worse accumulated reward for $\Delta_{\mathsf{T}} = 600$. In practice, however, the number of samples used for the initial clustering might be difficult to determine a-priori. In that situation, we suggest to restart the algorithm periodically with a small value of $\mathsf{T}_{\mathsf{static}}$. Indeed, since the batches are independent, the accumulated reward concentrates due to the law of large numbers.

Next, we further compare the performance of our algorithm to the static case [Bresler et al., 2014], and to the Popularity Amongst Friends (PAF) algorithm [Barman and Dabeer, 2012]. We consider the same setting as in [Bresler et al., 2014]. In particular, we again quantize movie ratings ≥ 4 as $+1$ (likable), movie ratings < 3 as -1 (unlikable), and missing ratings as 0. We consider the top $\mathsf{N} = 250$ and $\mathsf{M} = 500$ users and movies, respectively. This results in $\approx 80\%$ nonzero entries among the total number of entries in the rating matrix. There are of course missing entries in the resulted rating matrix. Accordingly, in our simulation if at a certain time, item i

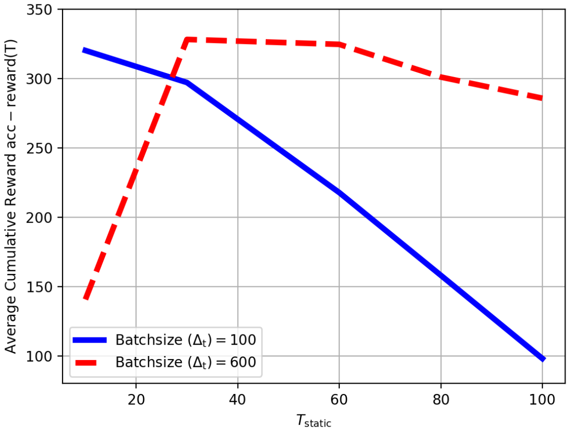

Figure 4: Comparison of Average Cumulative Reward $\mathsf{acc}-\mathsf{reward}(\mathsf{T})$ for batchsize $(\Delta_{\mathsf{T}}) \in \{100, 600\}$ and $\mathsf{T}_{\mathsf{static}} \in \{10, 30, 60, 80, 100\}$.

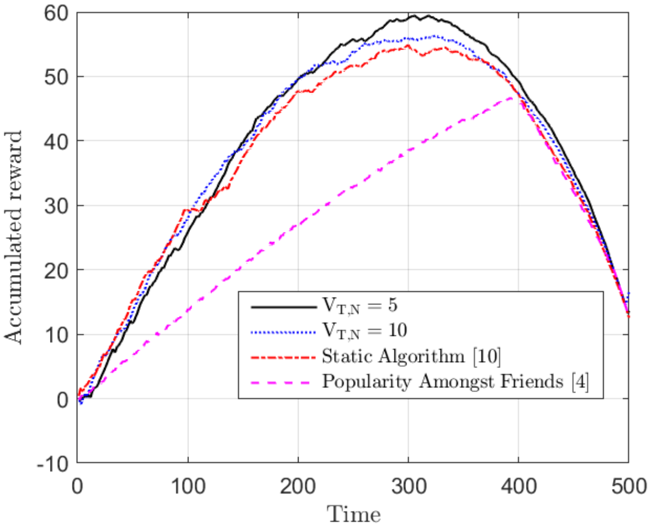

Figure 5: The accumulated reward over time achieved by Algorithm Collaborative and existing recommendation algorithm Popularity Amongst Friends [Barman and Dabeer, 2012], for several values of the variation budget $\mathsf{V} \in \{0, 5, 10\}$, using Movielens10m dataset.

was recommended to user u, who has not rated that item, we receive 0 reward. Despite that we will still treat item i as being consumed by user u, and accordingly, item i cannot be recommended to user u again. Since we allow algorithms to recommend an item to a given user only once, after $\mathsf{T} = \mathsf{M} = 500$ time steps, all items have been recommend to all users. As before, the performance of the algorithms is measured in terms of the average cumulative reward up to time T.

In the simulation, we run Algorithm Collaborative with three different values for the variation budget $\mathsf{V} = \mathsf{V}_1 = \mathsf{V}_2 \in \{0, 5, 10\}$, and recall that $\mathsf{V} = 0$ corresponds to the static case [Bresler et al., 2014]. The results are given in Fig. 5. It is evident that Algorithm Collaborative significantly outperforms PAF algorithm, a fact which was already observed in [Bresler et al., 2014]. More importantly we see that assuming that $\mathsf{V} = 5$, and accordingly recommending in batches, gives the best results among the other values of V, and in particular the static case. Except for coping with variations in the preferences of users, this can be attributed also to *model mismatch*. To wit, the static algorithm Recommend was designed for a certain probabilistic model which may not capture certain phenomena in real-world datasets. Accordingly, it might be the case that the algorithm will "stuck" on a certain wrong rating trajectory which will hinder the rate at which likeable movies are recommended. Working in batches, and by which letting the algorithm to "restart" occasionally, may compensate for this mismatch. Finally, note that the reason for the $\cap$-shape of the obtained curves is the fact that after recommending most of the likable items (around $t \approx 310$), mostly unlikable movies are left to recommend, until we exhaust all possible movies.

D Conclusion and Outlook

In this paper, we introduced a novel model for online non-stationary recommendation systems, where users may change their preferences over time adversarially. For this model, we analyzed the performance of a CF recommendation algorithm, and derived a lower bound on its achievable reward.

We hope our work has opened more doors than it closes. Apart from tightening the obtained lower bound on the reward, there are several exciting directions for future work. First, it is of significant importance to tackle the case where the number of variations is unknown. Devising universal algorithms which are oblivious to the knowledge of the non-stationarity, and proving theoretical guarantees is quite challenging (see, for example, the recent papers [Karnin and Anava, 2016, Auer et al., 2019, Chen et al., 2019] where the problem of non-stationary MAB with unknown number of variations was considered). Secondly, it is very interesting and technically challenging to derive information-theoretic upper bounds on the performance (reward) of any CF algorithm for the general model introduced in this paper. The results of this paper can be rather directly generalized to one-class recommendation systems where users only rate what they like and never reveal what they dislike. It would be interesting to introduce and analyze models which combines both content/graph information on top of the collaborative filtering information. Also, while in this paper our ultimate goal was to design recommender system which maximize the number of likes, in some applications one might want to take into account other aspects, such as fairness, novelty and multi-stakeholder recommender systems. Formally analyzing such aspects has not been done, and is of practical and theoretical importance. Finally, as was mentioned in the Appendix C there are several inherent challenges with standard CF datasets used for simulating (non-stationary) online recommender systems. Implementing a real interactive online recommendation system and testing our algorithms over it is an important step towards a complete understanding of CF based recommender systems.