On Differentially Private Online Collaborative Recommendation Systems

Seth Gilbert, Xiao Liu(✉), and Haifeng Yu

School of Computing, National University of Singapore, Singapore, Singapore
{seth.gilbert,liuxiao,haifeng}@comp.nus.edu.sg

Abstract. In collaborative recommendation systems, privacy may be compromised, as users' opinions are used to generate recommendations for others. In this paper, we consider an online collaborative recommendation system, and we measure users' privacy in terms of the standard notion of differential privacy. We give the first quantitative analysis of the trade-offs between recommendation quality and users' privacy in such a system by showing a lower bound on the best achievable privacy for any algorithm with non-trivial recommendation quality, and proposing a near-optimal algorithm. From our results, we find that there is actually little trade-off between recommendation quality and privacy, as long as non-trivial recommendation quality is to be guaranteed. Our results also identify the key parameters that determine the best achievable privacy.

Keywords: Differential privacy · Collaborative recommendation system · Lower bound · Online algorithm

1 Introduction

In this paper we consider an *online collaborative recommendation system* that attempts to predict which objects its users will like. Imagine, for example, a news website which publishes articles every day. When a user enjoys an article, he/she votes on the article (e.g., upvotes it, likes it, +1s it, etc.). Users can also ask the system for a recommendation, i.e., to suggest an article that they might like. After reading the recommended article, the user gives the system feedback on the recommendation so that it can improve its recommendation quality. In this paper, we work with a simplified, abstract version of this very common paradigm.

Due to the way it works, a collaborative recommendation system has the risks of leaking its users' privacy. Clearly, there are trade-offs between recommendation quality and privacy: a system that gives completely random recommendations certainly leaks no one's privacy, but it is also useless; in contrast, a recommendation system that gives high quality recommendations has to make "full use" of its users' data, which is more prone to privacy leakage.

A full version [21] of this paper is available at http://arxiv.org/abs/1510.08546. This research was supported by MOE ARC-2 grant MOE2014-T2-1-157.

© Springer International Publishing Switzerland 2016
S. Kwon and A. Yun (Eds.): ICISC 2015, LNCS 9558, pp. 210–226, 2016.
DOI: 10.1007/978-3-319-30840-1_14

In this paper, we adopt ϵ-differential privacy [17] as our formal definition of privacy, and we give the first quantitative analysis of these trade-offs for online collaborative recommendation systems. Prior to this paper, the topic of differentially private recommendation systems has primarily been examined under *offline matrix* models [12,13,23,24,28,32,42]. From the theoretical perspective, our recommendation model can be viewed as a variant of an *online learning* problem. Currently, there are only a limited number of existing papers on differentially private online learning [18,26,41], and their privacy models do not fit the recommendation problem (see Sect. 3 for more details).

We first study the best achievable privacy for a fixed recommendation quality by showing a near-tight lower bound on the privacy parameter ϵ (smaller ϵ means better privacy). For example, if we were to guarantee a trivial recommendation quality only, then we can achieve "perfect privacy" (i.e., $\epsilon = 0$) by ignoring users' opinions on objects and recommending randomly. As we set better and better target recommendation quality, it might be expected that the best achievable ϵ smoothly gets larger and larger. However, we show that the transition is sharp: although $\epsilon = 0$ is achievable for the trivial recommendation quality, the lower bound of ϵ rises to a certain level as long as non-trivial recommendation quality is to be guaranteed, and it remains essentially the same (up to a logarithmic factor) as the target recommendation quality increases.

We then propose a novel ϵ-differentially private algorithm. Our algorithm's ϵ is within a logarithmic factor to the aforementioned lower bound, and meanwhile its recommendation quality is also near-optimal up to a logarithmic factor, even when compared to algorithms providing no privacy guarantee.

Our near matching results surprisingly imply that there are actually little trade-offs between recommendation quality and privacy — an inherent "amount of privacy" (up to a logarithmic factor) must be "leaked" for *any* algorithm with non-trivial recommendation quality. Our results also identify the key parameters that fundamentally determine the best achievable recommendation quality and privacy. We provide more details about our results in Sect. 4.

2 Model and Problem Statement

2.1 Recommendation System Model

We now describe the model in more detail, abstracting away some of the complications in the scenario above in order to focus on the fundamental trade-offs.

We consider an online collaborative recommendation system that contains *voters*, *clients* and *objects*, and it repeatedly recommends objects to clients based on voters' opinions on objects. A voter/client either *likes* or *dislikes* an object. Voters submit their opinions on objects to the system in the form of *votes*, where a vote by voter i on object j indicates that voter i likes object j; clients receive recommendations from the system and provide *feedback* to the system which tells whether they like the recommended objects or not. Since every client has his/her own personalized preferences, the system will serve each client separately.

We now describe how the model operates for a particular client C. The system runs for T *rounds*. In each round $t \in \{1, \ldots, T\}$, a set of m *new* candidate objects

arrives in the system, out of which the client C likes at least one of them. We assume that m is a constant, and totally the system has mT objects over all the T rounds. Let $\mathcal{U}$ denote the set of all the voters, and $\mathcal{B}_t$ denote the set of candidate objects in the tth round. After $\mathcal{B}_t$ arrives, each voter $i \in \mathcal{U}$ votes on one object in $\mathcal{B}_t$; the system then recommends one object $b_t \in \mathcal{B}_t$ to the client C (based on the voters' votes and the previous execution history), and C responses the system with his/her feedback which tells whether he/she likes b_t or not. The system proceeds into the next round after that.

We measure the recommendation quality by *loss*, which is defined as the number of objects that the algorithm recommends to the client C but C dislikes.

A client C is fully characterized by specifying C's preferences on every object. However, in a recommendation system, whether a client C likes an object j or not is unknown until the system has recommended j to C and gotten the feedback.

We denote the votes of all the voters in $\mathcal{U}$ by $\mathcal{V}\langle\mathcal{U}\rangle$, and we call $\mathcal{V}\langle\mathcal{U}\rangle$ the *voting pattern of* $\mathcal{U}$, or simply a *voting pattern* when $\mathcal{U}$ is clear from the context. Given a client C and a voting pattern $\mathcal{V}\langle\mathcal{U}\rangle$, a (randomized) recommendation algorithm $\mathcal{A}$ maps the pair $(C, \mathcal{V}\langle\mathcal{U}\rangle)$ to a (random) sequence of objects in $\mathcal{B}_1 \times \cdots \times \mathcal{B}_T$. We call a particular sequence in $\mathcal{B}_1 \times \cdots \times \mathcal{B}_T$ a *recommendation sequence*.

2.2 Differential Privacy in Recommendation Systems

Voters' votes are assumed to be securely stored by the system, which are not accessible from the public. Nevertheless, a curious client may still try to infer voters' votes by analyzing the recommendation results. In this paper, we adopt *differential privacy* [17] as our definition of privacy. Roughly speaking, differential privacy protects privacy by ensuring that the outputs are "similar" for two voting patterns $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$ if they differ by one voter. Such a pair of voting patterns are called *adjacent voting patterns*, and they are formally defined as:

Definition 1 (Adjacent Voting Patterns). Two voting patterns $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$ are adjacent voting patterns iff i) $|\mathcal{U} \triangle \mathcal{U}'| = 1$, and ii) for any voter $i \in \mathcal{U} \cap \mathcal{U}'$ and in any round $t \in \{1, \ldots, T\}$, i always votes on the same object in both $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$.

Generalizing Definition 1, we say that two voting patterns $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$ are *k-step adjacent*, if there exists a sequence of $k+1$ voting patterns $\mathcal{V}\langle\mathcal{U}_0\rangle = \mathcal{V}\langle\mathcal{U}\rangle, \mathcal{V}\langle\mathcal{U}_1\rangle, \ldots, \mathcal{V}\langle\mathcal{U}_{k-1}\rangle, \mathcal{V}\langle\mathcal{U}_k\rangle = \mathcal{V}\langle\mathcal{U}'\rangle$ such that $\mathcal{V}\langle\mathcal{U}_\ell\rangle$ and $\mathcal{V}\langle\mathcal{U}_{\ell+1}\rangle$ are adjacent for any $\ell = 0, \ldots, k-1$.

Having defined adjacent voting patterns, we can then apply the standard differential privacy in [17] to our setting:

Definition 2 (ϵ-Differential Privacy). A recommendation algorithm $\mathcal{A}$ preserves ϵ-differential privacy if for any client C, any pair of adjacent voting patterns $\mathcal{V}\langle\mathcal{U}\rangle, \mathcal{V}\langle\mathcal{U}'\rangle$, and any subset $S \subseteq \mathcal{B}_1 \times \cdots \times \mathcal{B}_T$,

$$\Pr[\mathcal{A}(C, \mathcal{V}\langle\mathcal{U}\rangle) \in S] \leq e^{\epsilon} \Pr[\mathcal{A}(C, \mathcal{V}\langle\mathcal{U}'\rangle) \in S],$$

where the probabilities are over $\mathcal{A}$'s coin flips.

2.3 Attack Model, Power of the Adversary

As indicated by Definitions 1 and 2, we protect voters' privacy against the client. We do not need to protect the client's privacy because voters receive nothing from the system.

Our research goal is to study the theoretical hardness of the aforementioned recommendation problem, therefore we assume that there is an adversary with unlimited computational power who controls how the voters vote and which objects the client likes. The adversary tries to compromise our algorithm's loss/privacy by feeding the algorithm with "bad" inputs. From the perspective of game theory, our recommendation model can be viewed as a repeated game between the algorithm, who chooses the objects to recommend, and the adversary, who chooses the client's preferences on objects and the voting pattern. For our lower bounds, we consider an *oblivious adversary* that chooses the client's preferences on objects and the voting patterns in advance; for our upper bounds, we consider an *adaptive adversary* whose choice in time t can depend on the execution history prior to time t. By doing so, our results are only strengthened.

2.4 Notations

Next we introduce some notations that characterize the system. Some of them are also the key parameters that determine the best achievable loss/privacy.

The Client's Diversity of Preferences. A client C's *diversity of preferences* D_C is defined to be the number of rounds in which C likes more than one objects.

The Client's Peers. Inherently, a collaborative recommendation system is able to achieve small loss only if some voters have similar preferences to the client. Let the *distance* between a client C and a voter i be the total number of objects that are voted on by i but are disliked by C. Given a radius parameter $R \in \{0, \ldots, T\}$, we define a voter i to be a client C's *peer* if their distance is within R. Given a client C, a voting pattern $\mathcal{V}\langle\mathcal{U}\rangle$ and a radius parameter R, we can count the number of C's peers in $\mathcal{U}$, and we denote it by $P_{C,\mathcal{V}\langle\mathcal{U}\rangle,R}$.

Other Notations. We define n to be an upper bound of $|\mathcal{U}|$ (i.e., the number of voters), D to be an upper bound of D_C (i.e., the client's diversity of preferences), and P to be a lower bound of $P_{C,\mathcal{V}\langle\mathcal{U}\rangle,R}$ (i.e., the number of the client's peers). The reader may wonder why these parameters are defined as upper/lower bounds. The purpose is to give a succinct presentation. Take n as an example: since differential privacy needs to consider two voting patterns with different numbers of voters, if we define n as the number of voters, it would be unclear which voting pattern we are referring to. The reader can verify that by choosing the right directions for the parameters (e.g., we define n to be an upper bound, and P to be a lower bound), our definition does not weaken our results.

In general, we consider a large system that consists of many voters, many objects (over all the rounds), and runs for a long time. That is, n and T can be very large. In this paper, we also impose a (quite loose) requirement that $n = O(\text{poly}(T))$, i.e., n is not super large compared to T.

In reality, a client shall find more peers as more voters join the system. Otherwise, the client has an "esoteric tastes" and it is inherently hard for any collaborative system to help him/her. Thus, in this paper, we consider the case that $P \geq 6m$, i.e., the client has at least a constant number of peers.

2.5 Loss/Privacy Goal

In this paper, we consider the *worst-case expected loss* of the algorithm, that is, we aim to bound the algorithm's expected loss for any client C and any voting pattern $\mathcal{V}\langle\mathcal{U}\rangle$ such that $|\mathcal{U}| \leq n$, $D_C \leq D$ and $P_{C,\mathcal{V}\langle\mathcal{U}\rangle,R} \geq P$. Notice that $O(T)$ loss can be trivially achieved by ignoring voters' votes and recommending objects randomly. However, such an algorithm is useless, and hence we consider the more interesting case when *non-trivial* loss (i.e., $o(T)$ worst-case expected loss) is to be guaranteed. It can be shown that the worst-case expected loss is $\Omega(R)$ for any algorithm (Theorem 3). Therefore, sub-linear loss is achievable only when R is sub-linear. In this paper, we focus on the case when $R = O(T^\nu)$ for some constant $\nu < 1$.[1]

For the privacy, we aim to preserve ϵ-differential privacy. We study the best achievable ϵ-differential privacy for any given target loss.

3 Related Work

Recommendation Systems and Online Learning. The research on recommendation systems has a long history [1,40]. A classic recommendation model is the *offline matrix-based* model, in which the user-object relation is represented by a matrix. In this paper, we consider a very different *online* recommendation model. From the theoretical perspective, our model can be viewed as a variant of the "Prediction with Expert Advice" (PEA) problem in online learning [9]. Such an approach that models the recommendation systems as online learning problems has been adopted by other researchers as well, e.g., in [2,31,33,37,43].

Differential Privacy. There has been abundant research [14–16,18,20] on differential privacy. Much of the early research focused on answering a single query on a dataset. Progress on answering multiple queries with non-trivial errors was made later on, for both offline settings [4,19,25,38] (where the input is available in advance), and online settings [5,10,11,18,26,29,41] (where the input continuously comes). We will introduce the work on *differentially private online learning* in [18,26,41] with more details soon after, as they are most related to this paper.

[1] Technically, the assumptions that $n = O(\text{polylog}(T))$, $P \geq 6m$ and $R = O(T^\nu)$ are only for showing the near-optimality of our lower bound. Our lower bound itself remains to hold without these assumptions.

Protecting Privacy in Recommendation Systems. People are well aware of the privacy risks in collaborative recommendation systems. Two recent attacks were demonstrated in [34] (which de-anonymized a dataset published by Netflix) and [6] (which inferred users' historical data by combining passive observation of a recommendation system with auxiliary information). The research in [34] even caused the second Netflix Prize competition to be cancelled.

Many of the existing privacy-preserving recommendation systems adopted privacy notions other than differential privacy (e.g., [3,7,8,35,36,39]). For studies on *differentially private* recommendation systems, prior to our paper, most of them were for *offline matrix-based* models. Some experimentally studied the empirical trade-offs between loss and privacy (e.g., [13,32,42]); the others focused on techniques that manipulate matrices in privacy-preserving ways (e.g., [12,23,24,28]). In a recent work [22], the authors proposed a modified version of differential privacy (called distance-based differential privacy), and they showed how to implement distance-based differential privacy in matrix-based recommendation systems.

Differentially Private Online Learning. This paper is most related to *differentially private online learning*, as our recommendation model is a variant of the PEA problem in online learning. Currently, only a limited number of studies have been done on this area [18,26,41]. In [18], Dwork et al. proposed a differentially private algorithm for the PEA problem by plugging privacy-preserving online counters into "Follow the Perturbed Leader" algorithm [27]. In [26,41], differential privacy was considered under a more general online learning model called "Online Convex Programming."

Despite the similarity between our recommendation model and the learning models in [18,26,41], there is an important difference. Since their research is not for recommendation systems, they considered somewhat different notions of privacy from ours. Roughly speaking, if interpreting their models as recommendation problems, then their privacy goal is to ensure that each *voter* is "followed" with similar probabilities when running the algorithm with two adjacent voting patterns. Such a guarantee is not sufficient for a recommendation system. For example, an algorithm that always "follows" voter Alice is perfectly private in terms of their privacy definition, but completely discloses Alice's private votes.[2] Besides the difference in privacy definition, we provide both lower bound and upper bound results, while [18,26,41] only have upper bound results.

4 Our Results and Contributions

Main Results. Our first result is a lower bound on the best achievable privacy:

[2] On the other hand, our privacy definition does not imply their definitions either. Therefore these two types of privacy models are incomparable.

Theorem 1. *For any recommendation algorithm that guarantees* $L = O(T^\eta)$ *worst-case expected loss (*$\eta < 1$ *is a constant) and preserves* ϵ*-differential privacy,* $\epsilon = \Omega(\frac{1}{P}(D+R+\log\frac{T}{L})) = \Omega(\frac{1}{P}(D+R+\log T))$*, even for an oblivious adversary.*

Our second result is a near-optimal algorithm (the p-REC algorithm in Sect. 7.2):

Theorem 2. *The p-REC algorithm guarantees* $O((R+1)\log\frac{n}{P})$ *worst-case expected loss, and it preserves* $O(\frac{1}{P}(D+R+1)\log\frac{T}{R+1})$*-differential privacy, even for an adaptive adversary.*

It can be shown that the worst-case expected loss is $\Omega(R + \log\frac{n}{P})$ even for algorithms with no privacy guarantee (Theorem 3). Thus, p-REC's worst-case expected loss is within a logarithmic factor to the optimal. Recall that $R = O(T^\nu)$ for a constant $\nu < 1$ and $\log n = O(\log T)$, hence p-REC's worst-case expected loss is within $O(T^\eta)$ for some constant $\eta < 1$ too. Then, by Theorem 1, p-REC's privacy is also within a logarithmic factor to the optimal.

Discussion of our Results. Theorem 1 shows that a minimal amount of "privacy leakage" is inevitable, even for the fairly weak $O(T^\eta)$ target loss.

Moreover, unlike many other systems in which the utility downgrades linear to the privacy parameter ϵ, the loss in an online recommendation system is much more sensitive to ϵ: according to Theorem 2, we can achieve near-optimal loss for an $\epsilon = O(\frac{1}{P}(D+R+1)\log\frac{T}{R+1})$; meanwhile, only trivial loss is achievable for just a slightly smaller $\epsilon = o(\frac{1}{P}(D+R+\log T))$. In other words, the trade-offs between loss and privacy are rather little — the best achievable ϵ is essentially the same (up to a logarithmic factor) for *all* the algorithms with $O(T^\eta)$ worst-case expected loss.[3] For this reason, instead of designing an algorithm that has a tunable privacy parameter ϵ, we directly propose the p-REC algorithm that simultaneously guarantees both near-optimal loss and privacy.

From our results, we identify the key parameters D, P and R that determine the best achievable loss and/or privacy.

The parameter R characterizes the correlation between the client and the voters, and it is not surprised that the best achievable loss is inherently limited by R, because a basic assumption for any collaborative system is the existence of correlation in the data (e.g., the low-rank assumption in matrix-based recommendation systems), and the system works by exploring/exploiting the correlation.

We notice that a larger P gives better privacy. This is consistent with our intuition, as an individual's privacy is obtained by hiding oneself in a population.

We also notice that the best achievable privacy linearly depends on the client's diversity of preferences D and the radius parameter R. The parameter D looks to be unnatural at the first sight, and no prior research on recommendation systems has studied it. The reason might be that most of the prior research

[3] This statement actually holds for all the algorithms with $o(T)$ loss. In Theorem 1, we choose $O(T^\eta)$ target loss to get a clean expression for the lower bound on ϵ, and a similar (but messier) lower bound on ϵ holds for $o(T)$ target loss too.

focused on the loss, and D has no impact on the loss (the loss should only be smaller if a client likes more objects). Nevertheless, in this paper, we discover that D is one of the fundamental parameters that determine the best achievable privacy. We provide an intuitive explanation of ϵ's linear dependence on D and R with an illustrative example in Sect. 7.1.

5 Preliminaries

Let $\mathcal{P}$ and $\mathcal{Q}$ be two distributions over sample space Ω. The *relative entropy* between $\mathcal{P}$ and $\mathcal{Q}$ is defined as $\sum_{\omega\in\Omega}\mathcal{P}(\omega)\ln\frac{\mathcal{P}(\omega)}{\mathcal{Q}(\omega)}$, where $\mathcal{P}(\omega)$ and $\mathcal{Q}(\omega)$ is the probability of ω in $\mathcal{P}$ and $\mathcal{Q}$, respectively. We adopt the conventions that $0\log\frac{0}{0}=0$, $0\log\frac{0}{x}=0$ for any $x>0$ and $x\log\frac{x}{0}=\infty$ for any $x>0$. It is well known that relative entropy is always non-negative [30].

In this paper, we often simultaneously discuss two executions $\mathcal{A}(C,\mathcal{V}\langle\mathcal{U}\rangle)$ and $\mathcal{A}(C,\mathcal{V}\langle\mathcal{U}'\rangle)$ for some algorithm $\mathcal{A}$, some client C and two voting patterns $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$. As a notation convention, we will use $\Pr[\cdot]$ and $\Pr'[\cdot]$ to denote the probability of some event in the execution of $\mathcal{A}(C,\mathcal{V}\langle\mathcal{U}\rangle)$ and $\mathcal{A}(C,\mathcal{V}\langle\mathcal{U}'\rangle)$, respectively. For any recommendation sequence $b=(b_1,\ldots,b_T)\in\mathcal{B}_1\times\cdots\times\mathcal{B}_T$ and any round t, we define the random variables $\mathcal{E}_t(b)=\ln\frac{\Pr[b_t|b_1,\ldots,b_{t-1}]}{\Pr'[b_t|b_1,\ldots,b_{t-1}]}$ and $\mathcal{E}(b)=\ln\frac{\Pr[b]}{\Pr'[b]}$. It then follows that $\mathcal{E}(b)=\sum_{t=1}^{T}\mathcal{E}_t(b)$. We also define random variable $\mathcal{L}_t$ to be the loss of execution $\mathcal{A}(C,\mathcal{V}\langle\mathcal{U}\rangle)$ in the tth round.

Finally, we list the following lower bound for the worst-case expected loss. Theorem 3 can be proved by constructing a client with random opinions on objects. Please see the full version [21] of this paper for its proof.

Theorem 3. *The worst-case expected loss of any recommendation algorithm is $\Omega(R+\log\frac{n}{P})$, even for an algorithm providing no privacy guarantee and an oblivious adversary.*

6 The Special Setting Where $D=R=0$

In order to better explain our ideas, we start by discussing the simple setting where $D=R=0$. That is, the client likes exactly one object in every round, and the client's peers never vote on any object that the client dislikes. We discuss the general setting where $D+R\geq 0$ in the next section.

6.1 Lower Bound

When $D=R=0$, we have the following lower bound on the privacy:

Theorem 4. *For any recommendation algorithm that guarantees $L=o(T)$ worst-case expected loss and preserves ϵ-differential privacy, if $D=R=0$, then $\epsilon=\Omega(\frac{1}{P}\log\frac{T}{L})$, even for an oblivious adversary.*

Proof (Sketch). Due to the limitation of space, here we provide a proof sketch of Theorem 4. The reader can refer to [21] for the full proof of this theorem. Our proof consists two main steps. In the first step, we consider a particular round t and two clients Alice and Bob who have different preferences in the tth round. Since the algorithm has to provide good recommendations to both Alice and Bob, the output distributions must be very different. Meanwhile, we carefully construct Alice's and Bob's executions, such that their voting patterns are $O(P)$-step adjacent to each other, hence the output distributions cannot be much different. From this dilemma we can establish a lower bound for the tth round. In the second step, we then extend this lower bound to all the T rounds using mathematical induction.

6.2 Algorithm

We propose the following Algorithm 1 for the simple setting where $D = R = 0$. As we will see, it is a special case of the general p-REC algorithm in Sect. 7.2. Therefore, we call it the p-REC$_{\text{sim}}$ algorithm ("sim" is short for "simple").

The p-REC$_{\text{sim}}$ algorithm maintains a weight value $\mathsf{weight}[i]$ for each voter i, and it recommends objects according to voters' weight in each round by invoking the procedure `RecommendByWeight()`. When it receives the client's feedback, it invokes the procedure `UpdateWeight()` to update voters' weight. In p-REC$_{\text{sim}}$, each voter's weight is either 1 or 0. A voter with 0 weight has no impact on the algorithm's output, and once a voter's weight is set to 0, it will never be reset to 1. Therefore, we can think of that `UpdateWeight()` works by "kicking out" voters from the system. We call the voters who have not been kicked out (i.e., those who have non-zero weight) *surviving voters*.

The p-REC$_{\text{sim}}$ algorithm shares a similar structure to the classic Weighted Average algorithm for the PEA problem [9], as they both introduce weight to voters and output according to the weight. Our core contribution is the dedicated probability of recommending objects. In each round t, p-REC$_{\text{sim}}$ recommends object j with probability $\gamma \cdot \frac{1}{m} + (1-\gamma) \cdot \frac{\phi(x_{j,t})}{\sum_{k \in \mathcal{B}_t} \phi(x_{k,t})}$, where $x_{j,t}$ is the fraction of surviving voters voting on object j in round t. We have:

Theorem 5. *If $D = R = 0$, then the p-REC$_{sim}$ algorithm guarantees $O(\log \frac{n}{P})$ worst-case expected loss and it preserves $O(\frac{1}{P} \log T)$-differential privacy, even for an adaptive adversary.*

According to Theorem 3, p-REC$_{\text{sim}}$'s loss is within a constant factor to the optimal. Then by Theorem 4, p-REC$_{\text{sim}}$'s ϵ is also within a constant factor to the optimal ϵ among all the algorithms that guarantee $O(T^\eta)$ worst-case expected loss. We prove p-REC$_{\text{sim}}$'s loss bound in [21]. Here we briefly introduce the main steps to analyze p-REC$_{\text{sim}}$'s privacy.

Consider the executions p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}\rangle)$ and p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}'\rangle)$, where C is any client and $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$ are any pair of adjacent voting patterns ($\mathcal{U}$ contains one more voter than $\mathcal{U}'$). To show that p-REC$_{\text{sim}}$ preserves $O(\frac{1}{P} \log T)$-differential privacy, it is sufficient to show that $|\mathcal{E}(b)| = O(\frac{1}{P} \log T)$ for any

Input : A client C, a voting pattern $\mathcal{V}\langle\mathcal{U}\rangle$
Output : Recommend an object from $\mathcal{B}_t$ to client C in each round t
Initialization: $\gamma \leftarrow \frac{m}{3T-1}$, $\lambda \leftarrow 2m \ln T$, $\rho \leftarrow \frac{1}{2m}$, $\mathsf{weight}[i] \leftarrow 1$ for each $i \in \mathcal{U}$

Procedure `Main()`
  foreach *round* $t = 1, \ldots, T$ **do**
    obj $\leftarrow$ `RecommendByWeight(weight[ ])`;
    Recommend object obj to the client C;
    feedback $\leftarrow$ the client C's feedback on object obj;
    `UpdateWeight(weight[ ], obj, feedback)`;

Procedure `RecommendByWeight(weight[ ])`
  foreach *object* $j \in \mathcal{B}_t$ **do**
    $x_{j,t} \leftarrow \frac{\sum_{i \in \mathcal{U}_{j,t}} \mathsf{weight}[i]}{\sum_{i \in \mathcal{U}} \mathsf{weight}[i]}$, where $\mathcal{U}_{j,t}$ is the set of voters who vote on object j in round t;
  Independently draw a Bernoulli random variable Z_t with $\Pr[Z_t = 1] = \gamma$;
  if $Z_t = 1$ **then**
    Independently draw an object obj from $\mathcal{B}_t$ uniformly at random;
  else
    Independently draw an object obj from $\mathcal{B}_t$ according to the following distribution: each object $j \in \mathcal{B}_t$ is drawn with probability proportional to $\phi(x_{j,t})$, where $\phi(x) = \begin{cases} 0 & \text{if } x \le \rho, \\ e^{\lambda x} - e^{\lambda \rho} & \text{otherwise;} \end{cases}$
  return obj;

Procedure `UpdateWeight(weight[ ], obj, feedback)`
  if feedback = *"dislike"* **then**
    $\mathsf{weight}[i] \leftarrow 0$ for every voter i who votes on object obj;
  else
    $\mathsf{weight}[i] \leftarrow 0$ for every voter i who does not vote on object obj;

Algorithm 1. The p-REC$_{\text{sim}}$ algorithm.

recommendation sequence $b = (b_1, \ldots, b_T)$. From now on, we will consider a fixed b, a fixed C and a fixed pair of $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$.

Given $b = (b_1, \ldots, b_T)$, let $W_t(b) = \sum_{i \in \mathcal{U}} \mathsf{weight}[i]$ be the number of surviving voters at the beginning of round t in execution p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}\rangle)$, conditioned on that the recommendations in the first $t-1$ rounds are $b_1, \ldots, b_{t-1}$. Since p-REC$_{\text{sim}}$ never kicks out the client's peers, $W_t(b) \ge P \ge 6m$.

First, we upper-bound the "privacy leakage" in each single round:

Lemma 6. *For any round* t, $|\mathcal{E}_t(b)| \le 3\lambda \cdot \frac{1}{W_t(b)}$.

Lemma 6 can be shown by a straightforward but rather tedious calculation, see the full version [21] of this paper for the proof.

Next, we show that a constant fraction of surviving voters are kicked out whenever there is non-zero "privacy leakage:"

Lemma 7. *For any round t, if $|\mathcal{E}_t(b)| \neq 0$, then $W_{t+1}(b) \leq W_t(b) \cdot (1 - \frac{1}{3m})$.*

Proof. Notice that $|\mathcal{E}_t(b)| \neq 0$ iff $\Pr[b_t|b_1, \ldots, b_{t-1}] \neq \Pr'[b_t|b_1, \ldots, b_{t-1}]$. Let x and x' be the fraction of surviving voters voting on the recommended object b_t in execution p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}\rangle)$ and p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}'\rangle)$, respectively. Since there are $W_t(b)$ surviving voters, $|x - x'| \leq \frac{1}{W_t(b)} \leq \frac{1}{P} \leq \frac{1}{6m}$.

We claim that $x > \frac{1}{3m}$. Assume for contradiction that $x \leq \frac{1}{3m}$. Since $|x - x'| \leq \frac{1}{6m}$, both x and x' will be no larger than $\frac{1}{3m} + \frac{1}{6m} = \frac{1}{2m} = \rho$. Notice that $\phi(\zeta) = 0$ for any variable $\zeta \leq \rho$, it then follows that $\phi(x) = \phi(x') = 0$ and $\Pr[b_t|b_1, \ldots, b_{t-1}] = \Pr'[b_t|b_1, \ldots, b_{t-1}] = \gamma \cdot \frac{1}{m}$, contradiction.

If the clients dislikes the recommended object b_t, by p-REC$_{\text{sim}}$'s rule of updating weight, $x > \frac{1}{3m}$ fraction of surviving voters will be kicked out.

If the clients likes b_t, then there must exist another object $\xi \in \mathcal{B}_t$ which is different from b_t, such that $\Pr[\xi|b_1, \ldots, b_{t-1}] \neq \Pr'[\xi|b_1, \ldots, b_{t-1}]$. Otherwise, if all the other objects are recommended with the same probability in executions p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}\rangle)$ and p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}'\rangle)$, so will be b_t, contradiction. By similar arguments, there are at least $\frac{1}{3m}$ fraction of surviving voters voting on the object ξ in both p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}\rangle)$ and p-REC$_{\text{sim}}(C, \mathcal{V}\langle\mathcal{U}'\rangle)$. Since p-REC$_{\text{sim}}$ kicks out all the voters who do not vote on b_t (including those who vote on ξ), again we get the desired result. □

Lemma 6 states that $|\mathcal{E}_t(b)|$ is $O(\frac{\lambda}{W_t(b)}) = O(\frac{1}{P}\log T)$. Lemma 7 implies that there can be at most $O(\log \frac{n}{P})$ rounds with $|\mathcal{E}_t(b)| \neq 0$. A combination of these two lemmas immediately shows that overall we have $O(\frac{1}{P}\log T \cdot \log \frac{n}{P})$-differential privacy. With a bit more careful analysis, we can remove the extra $\log \frac{n}{P}$ factor. We leave the details to [21].

7 The General Setting Where $D + R \geq 0$

7.1 Lower Bound

In this section, we prove Theorem 1. If $0 \leq D + R < 6 \ln T$ and the target loss $L = O(T^\eta)$, then $\Omega(\log \frac{T}{L}) = \Omega(D+R+\log \frac{T}{L})$ and hence Theorem 1 is implied by Theorem 4. When $D + R \geq 6 \ln T$, we have the following Theorem 8. Theorem 1 is then proved because $\Omega(D + R) = \Omega(D + R + \log \frac{T}{L})$ if $D + R \geq 6 \ln T$.

Theorem 8. *For any recommendation algorithm that guarantees $L = o(T)$ worst-case expected loss and preserves ϵ-differential privacy, if $D + R \geq 6 \ln T$, then $\epsilon = \Omega(\frac{1}{P}(D + R))$, even for an oblivious adversary.*

Before proving Theorem 8, we first explain the intuition behind the proof by a simple illustrative example. Imagine that there is one client Alice, and two voting patterns $\mathcal{V}_1$ and $\mathcal{V}_2$. Both $\mathcal{V}_1$ and $\mathcal{V}_2$ contain only one voter named Bob, but Bob may vote differently in $\mathcal{V}_1$ and $\mathcal{V}_2$. We let Bob be Alice's peer in both $\mathcal{V}_1$ and $\mathcal{V}_2$. For simplicity let us set $R = 0$, so Bob never votes on any object that Alice dislikes. By Definition 1, $\mathcal{V}_1$ and $\mathcal{V}_2$ are 2-step voting patterns.

Now consider a particular round t with two candidate objects. If Alice likes only one of the objects, then there is only one way for Bob to cast vote; otherwise Bob will no longer be a peer of Alice. However, if Alice likes both objects, then Bob can vote on different objects in $\mathcal{V}_1$ and $\mathcal{V}_2$ without breaking the promise that he is Alice's peer. Since Bob is the only information source of the system, an recommendation algorithm $\mathcal{A}$ has to somehow "follow" Bob, and hence the distributions of the executions $\mathcal{A}(\text{Alice}, \mathcal{V}_1)$ and $\mathcal{A}(\text{Alice}, \mathcal{V}_2)$ will be different. If Alice's diversity of preferences is D, then this situation can happen for D times, which results an $\epsilon \propto D$. The linear dependency of ϵ on R is for a similar reason.

Proof (Sketch of Theorem 8). Due to the limitation of space, we provide a proof sketch in the main text. The full proof can be found in [21].

We first prove Theorem 8 for the case where $P = 1$ and $6 \ln T \leq D + R \leq T$. To show Theorem 8 for the case where $P = 1$, it is sufficient to show that for any given algorithm $\mathcal{A}$, we can construct a client C and a pair of 2-step adjacent voting patterns $\mathcal{V}\langle\mathcal{U}\rangle, \mathcal{V}\langle\mathcal{U}'\rangle$, such that $\ln \frac{\Pr[b]}{\Pr'[b]} = \Omega(D + R)$ for some recommendation sequence $b \in \mathcal{B}_1 \times \cdots \times \mathcal{B}_T$.

We construct the client C by setting C's preferences on objects. We will always ensure that C likes multiple objects in at most D rounds. For the voting pattern $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$, we let each of them contain one voter U and U', respectively. We construct the voting patterns by setting U and U''s votes in each round. We will always ensure that both U and U' vote on at most R objects that are disliked by the client C, hence U and U' are both the client C's peers.

We construct C, $\mathcal{V}\langle\mathcal{U}\rangle$ and $\mathcal{V}\langle\mathcal{U}'\rangle$ round by round. Imagine that we are in the beginning of the tth round, with the previous recommendation history being $b_{<t} = (b_1, \ldots, b_{t-1})$. In order to better demonstrate our ideas, let us temporarily assume an adaptive adversary who can also see the recommendation history $b_{<t}$. The adversary can then set C's preferences on objects and U and U''s votes based on the algorithm $\mathcal{A}$'s behavior:

- *Case 1*: $\mathcal{A}$ "follows" voter U with probability ≤ 0.75. In this case, the adversary let C like exactly one object in round t, and it let U vote on the only object that C likes. It then follows that $\mathbb{E}[\mathcal{L}_t | b_{<t}] \geq 1 - 0.75 = 0.25$, i.e., the expected loss in round t is at least a constant.
- *Case 2.a*: $\mathcal{A}$ "follows" voter U with probability > 0.75, but it "follows" the voter U' with probability ≤ 0.5. In this case, if the adversary let U and U' vote *identically*, then the distributions $\Pr[\cdot|b_{<t}]$ and $\Pr'[\cdot|b_{<t}]$ will be different. In particular, we can show that the relative entropy $\mathbb{E}[\mathcal{E}_t | b_{<t}] \geq 0.13$, i.e., the expected "privacy leakage" in round t is at least a constant.
- *Case 2.b*: $\mathcal{A}$ "follows" voter U with probability ≥ 0.75, and it "follows" voter U' with probability > 0.5. This case is symmetric to *Case 2.a*, and hence the adversary can force $\mathbb{E}[\mathcal{E}_t | b_{<t}] \geq 0.13$ by letting U and U' vote *differently*. However, it is worth noting that during the entire execution of $\mathcal{A}$, the adversary can let U and U' vote differently for at most $D + R$ times due to the constraints imposed by D and R (see the full proof for more details).

It can be shown that with the above adaptive construction, $\mathbb{E}[\mathcal{E}] = \Omega(D+R)$ for any algorithm $\mathcal{A}$, which implies the existence of one recommendation sequence b such that $\mathcal{E}(b) = \ln \frac{\Pr[b]}{\Pr'[b]} = \Omega(D+R)$. To see why $\mathbb{E}[\mathcal{E}] = \Omega(D+R)$, we first notice that there cannot be too many rounds in *Case 1* on expectation, because $\mathcal{A}$ has to ensure $o(T)$ expected loss. Therefore most of the rounds must be in *Case 2.a* or *Case 2.b*. If there are many rounds in *Case 2.a*, then $\mathbb{E}[\mathcal{E}]$ must be large because $\mathbb{E}[\mathcal{E}_t|b_{<t}] \geq 0.13$ in every *Case 2.a* round, and $\mathbb{E}[\mathcal{E}_t|b_{<t}] \geq 0$ in all the other rounds (relative entropy is non-negative [30]). Otherwise, there must be many rounds in *Case 2.b*. In this case, the adversary can force $\mathbb{E}[\mathcal{E}_t|b_{<t}] \geq 0.13$ for $\Omega(D+R)$ times, and we have $\mathbb{E}[\mathcal{E}] = \Omega(D+R)$.

The aforementioned adaptive adversary chooses a "bad setting" for the algorithm $\mathcal{A}$ in each round based on which case $\mathcal{A}$ is in. We point out that this is actually not necessary: we still have $\mathbb{E}[\mathcal{E}] = \Omega(D+R)$ if the adversary randomly chooses a "bad setting" in each round with a proper distribution. Such a (random) adversary is oblivious, and it implies the existence of a "bad input" that does not depend on $\mathcal{A}$'s execution. This finishes the proof in the case of $P = 1$.

Finally, we prove Theorem 8 for the cases where $D + R > T$ and/or $P > 1$. These proofs are just simple extensions of the above basic proof. □

7.2 Algorithm

We propose the following p-REC algorithm for the general setting where $D+R \geq 0$. The p-REC algorithm is a generalized version of the p-REC$_{\text{sim}}$ algorithm, and it shares a similar structure as that of p-REC$_{\text{sim}}$, except that the procedure `UpdateWeight()` is replaced by `UpdateCreditAndWeight()`. In fact, we can get back the p-REC$_{\text{sim}}$ algorithm by setting $D = R = 0$ in the p-REC algorithm.

Theorem 2 summarizes p-REC's loss and privacy. According to the lower bounds in Theorems 1 and 3, both its loss and privacy are within logarithmic factors to the optimal.

In the beginning of the p-REC algorithm, each voter $i \in \mathcal{U}$ is initialized with two *credit* values $\mathsf{credit}^{(D)}[i] = 2D$ (which we call *D-credit*) and $\mathsf{credit}^{(R)}[i] = 2R+1$ (which we call *R-credit*). In each round t, the algorithm recommends objects by invoking the `RecommendByWeight()` procedure. After it receives the client's feedback, the algorithm updates each voter's credit and then calculate his/her weight by invoking the `UpdateCreditAndWeight()` procedure.

To see the intuition behind the p-REC algorithm, let us analyze why the p-REC$_{\text{sim}}$ algorithm fails in the general setting where $D + R \geq 0$. If we run p-REC$_{\text{sim}}$ in the general setting, we may end up with a situation where all the client's peers are kicked out from the system. A client's peer can be (wrongly) kicked out in two scenarios:

- when the client likes more than one objects in some round, the peer votes on one such object, but another such object is recommended;
- when the peer votes on an object that the client dislikes, and that object is recommended to the client.

Input : A client C, a voting pattern $\mathcal{V}\langle\mathcal{U}\rangle$
Output : Recommend an object from $\mathcal{B}_t$ to client C in each round t
Initialization: $\gamma \leftarrow \frac{m}{(3T/(R+1))-1}$; $\lambda \leftarrow 2m \ln \frac{T}{R+1}$; $\rho \leftarrow \frac{1}{2m}$; for each $i \in \mathcal{U}$:
$\mathsf{credit}^{(D)}[i] \leftarrow 2D$, $\mathsf{credit}^{(R)}[i] \leftarrow 2R+1$, $\mathsf{weight}[i] \leftarrow 1$

Procedure `Main()`
  foreach *round* $t = 1, \ldots, T$ **do**
    obj ← `RecommendByWeight(`weight[ ]`)`;
    Recommend object obj to the client C;
    feedback ← the client C's feedback on object obj;
    `UpdateCreditAndWeight(`$\mathsf{credit}^{(D)}$[ ], $\mathsf{credit}^{(R)}$[ ], weight[ ], obj, feedback`)`;

Procedure `UpdateCreditAndWeight(`$\mathsf{credit}^{(D)}$[ ], $\mathsf{credit}^{(R)}$[ ], weight[ ], obj, feedback`)`
  if feedback = *"dislike"* **then**
    $\mathsf{credit}^{(R)}[i] \leftarrow \mathsf{credit}^{(R)}[i] - 1$ for every voter i who votes on obj;
  else
    $\mathsf{credit}^{(D)}[i] \leftarrow \mathsf{credit}^{(D)}[i] - 1$ for every voter i who does not vote on obj;
  foreach *voter* $i \in \mathcal{U}$ **do**
    if $\mathsf{credit}^{(R)}[i] > 0$ *and* $\mathsf{credit}^{(D)}[i] + \mathsf{credit}^{(R)}[i] > 0$ **then**
      $\mathsf{weight}[i] \leftarrow 1$;
    else
      $\mathsf{weight}[i] \leftarrow 0$;

Algorithm 2. Privacy-preserving RECommendation (p-REC) algorithm.

However, since these two scenarios can happen for at most $D + R$ times, a natural idea is to give a voter $D+R$ more "chances" before we kick out him/her. Motivated by this, we could initialize each voter i with $D + R + 1$ credit, and deduct i's credit by 1 when i is caught to vote on an object the client dislikes, or when the client likes the recommended object but i does not vote on it. We kick out a voter only when he/she has no credit.

For some technical reasons, p-REC needs to introduce two types of credit (D-credit and R-credit), and deducts different types of credit in different situations. It also initializes each voter with $2D$ (instead of D) D-credit and $2R + 1$ (instead of $R + 1$) R-credit. The analysis of the p-REC algorithm is similar in spirit to that of the p-REC$_{\text{sim}}$ algorithm, and we leave the details to [21].

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. **17**(6), 734–749 (2005)
2. Awerbuch, B., Hayes, T.P.: Online collaborative filtering with nearly optimal dynamic regret. In: Proceedings of the 19th Annual ACM Symposium on Parallelism in Algorithms and Architectures, pp. 315–319. ACM (2007)

3. Berkovsky, S., Eytani, Y., Kuflik, T., Ricci, F.: Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In: Proceedings of the 2007 ACM Conference on Recommender Systems. pp. 9–16. ACM (2007)
4. Blum, A., Ligett, K., Roth, A.: A learning theory approach to non-interactive database privacy. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, pp. 609–618. ACM (2008)
5. Bolot, J., Fawaz, N., Muthukrishnan, S., Nikolov, A., Taft, N.: Private decayed predicate sums on streams. In: Proceedings of the 16th International Conference on Database Theory, pp. 284–295. ACM (2013)
6. Calandrino, J., Kilzer, A., Narayanan, A., Felten, E.W., Shmatikov, V., et al.: "You might also like:" privacy risks of collaborative filtering. In: Proceedings of the 2011 IEEE Symposium on Security and Privacy, pp. 231–246. IEEE (2011)
7. Canny, J.: Collaborative filtering with privacy. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy, pp. 45–57. IEEE (2002)
8. Canny, J.: Collaborative filtering with privacy via factor analysis. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 238–245. ACM (2002)
9. Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, New York (2006)
10. Chan, T.-H.H., Li, M., Shi, E., Xu, W.: Differentially private continual monitoring of heavy hitters from distributed streams. In: Fischer-Hübner, S., Wright, M. (eds.) PETS 2012. LNCS, vol. 7384, pp. 140–159. Springer, Heidelberg (2012)
11. Chan, T.H.H., Shi, E., Song, D.: Private and continual release of statistics. ACM Trans. Inf. Syst. Secur. **14**(3), 26 (2011)
12. Chaudhuri, K., Sarwate, A.D., Sinha, K.: A near-optimal algorithm for differentially-private principal components. J. Mach. Learn. Res. **14**(1), 2905–2943 (2013)
13. Chow, R., Pathak, M.A., Wang, C.: A practical system for privacy-preserving collaborative filtering. In: Proceedings of the 12th IEEE International Conference on Data Mining Workshops (ICDMW), pp. 547–554. IEEE (2012)
14. Dwork, C.: Differential privacy: a survey of results. In: Agrawal, M., Du, D.-Z., Duan, Z., Li, A. (eds.) TAMC 2008. LNCS, vol. 4978, pp. 1–19. Springer, Heidelberg (2008)
15. Dwork, C.: The differential privacy frontier (extended abstract). In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 496–502. Springer, Heidelberg (2009)
16. Dwork, C.: A firm foundation for private data analysis. Commun. ACM **54**(1), 86–95 (2011)
17. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
18. Dwork, C., Naor, M., Pitassi, T., Rothblum, G.N.: Differential privacy under continual observation. In: Proceedings of the 42nd ACM Symposium on Theory of Computing, pp. 715–724. ACM (2010)
19. Dwork, C., Naor, M., Reingold, O., Rothblum, G.N., Vadhan, S.: On the complexity of differentially private data release: efficient algorithms and hardness results. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, pp. 381–390. ACM (2009)
20. Dwork, C., Smith, A.: Differential privacy for statistics: what we know and what we want to learn. J. Priv. confidentiality **1**(2), 2 (2010)
21. Gilbert, S., Liu, X., Yu, H.: On differentially private online collaborative recommendation systems. ArXiv e-prints (2015). arxiv:1510.08546

22. Guerraoui, R., Kermarrec, A.M., Patra, R., Taziki, M.: D2P: distance-based differential privacy in recommenders. Proc. VLDB Endowment **8**(8), 862–873 (2015)
23. Hardt, M., Roth, A.: Beating randomized response on incoherent matrices. In: Proceedings of the 44th annual ACM Symposium on Theory of Computing, pp. 1255–1268. ACM (2012)
24. Hardt, M., Roth, A.: Beyond worst-case analysis in private singular vector computation. In: Proceedings of the 45th annual ACM Symposium on Theory of Computing, pp. 331–340. ACM (2013)
25. Hardt, M., Rothblum, G.N.: A multiplicative weights mechanism for privacy-preserving data analysis. In: Proceedings of the 51th Annual IEEE Symposium on Foundations of Computer Science, pp. 61–70. IEEE (2010)
26. Jain, P., Kothari, P., Thakurta, A.: Differentially private online learning. In: Proceedings of the 25th Annual Conference on Learning Theory, pp. 24.1–24.34 (2011)
27. Kalai, A., Vempala, S.: Efficient algorithms for online decision problems. J. Comput. Syst. Sci. **71**(3), 291–307 (2005)
28. Kapralov, M., Talwar, K.: On differentially private low rank approximation. In: Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1395–1414. SIAM (2013)
29. Kellaris, G., Papadopoulos, S., Xiao, X., Papadias, D.: Differentially private event sequences over infinite streams. Proc. VLDB Endowment **7**(12), 1155–1166 (2014)
30. Kullback, S.: Information Theory and Statistics. Courier Corporation, New York (1968)
31. Lee, W.S.: Collaborative learning for recommender systems. In: Proceedings of the 18th International Conference on Machine Learning, pp. 314–321 (2001)
32. McSherry, F., Mironov, I.: Differentially private recommender systems: building privacy into the net. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 627–636. ACM (2009)
33. Nakamura, A., Abe, N.: Collaborative filtering using weighted majority prediction algorithms. In: Proceedings of the 15th International Conference on Machine Learning, pp. 395–403 (1998)
34. Narayanan, A., Shmatikov, V.: Robust de-anonymization of large sparse datasets. In: Proceedings of the 2008 IEEE Symposium on Security and Privacy, pp. 111–125. IEEE (2008)
35. Polat, H., Du, W.: Privacy-preserving collaborative filtering. Int. J. Electron. Commer. **9**(4), 9–35 (2003)
36. Polat, H., Du, W.: SVD-based collaborative filtering with privacy. In: Proceedings of the 2005 ACM Symposium on Applied Computing, pp. 791–795. ACM (2005)
37. Resnick, P., Sami, R.: The influence limiter: provably manipulation-resistant recommender systems. In: Proceedings of the 2007 ACM Conference on Recommender Systems, pp. 25–32. ACM (2007)
38. Roth, A., Roughgarden, T.: Interactive privacy via the median mechanism. In: Proceedings of the 42nd ACM Symposium on Theory of Computing, pp. 765–774. ACM (2010)
39. Shokri, R., Pedarsani, P., Theodorakopoulos, G., Hubaux, J.P.: Preserving privacy in collaborative filtering through distributed aggregation of offline profiles. In: Proceedings of the 2009 ACM Conference on Recommender Systems, pp. 157–164. ACM (2009)
40. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv. Artif. Intell. **2009**, Article ID 421425, 19 (2009). Doi:10.1155/2009/421425

41. Thakurta, A.G., Smith, A.: (Nearly) optimal algorithms for private online learning in full-information and bandit settings. In: Advances in Neural Information Processing Systems, pp. 2733–2741 (2013)
42. Xin, Y., Jaakkola, T.: Controlling privacy in recommender systems. In: Advances in Neural Information Processing Systems, pp. 2618–2626 (2014)
43. Yu, H., Shi, C., Kaminsky, M., Gibbons, P.B., Xiao, F.: Dsybil: optimal sybil-resistance for recommendation systems. In: Proceedings of the 2009 IEEE Symposium on Security and Privacy, pp. 283–298. IEEE (2009)