Differentially Private Recommendation Systems: Practical Implementation and Interpretation

Yubin Park and Ayan Acharya
Electrical and Computer Engineering Department
The University of Texas at Austin
Email: yubin.park, masterayan@gmail.com

Dec 08, 2011

Abstract

Recommendation systems have became vital components in daily web services such as the DVD recommendation system in *Netflix*, the book recommendation system in *Amazon.com*, and the friend recommendation system in *Facebook*. However, collaborative filtering-based recommendations might leak sensitive information for some group of people, as their training data range from customers' transaction logs, social connections and even banking information. Furthermore, previous privacy attack cases confirm that the privacy concern for recommendation systems is not merely a symptom of growing privacy alarmism. In this project, we derive and analyze "differentially private" recommendation systems, and provide experimental results using the MovieLens dataset. Moreover, we propose a novel interpretation of the privacy parameter ϵ in differential privacy, which would help in determining the optimal value of ϵ in real settings. We explore two major frameworks: (i) item-to-item and (ii) user-to-item collaborative filterings. Both cases cover a wide range of applications in various domains, and our output perturbation frameworks are readily applicable to many existing algorithms. The main objective of this project is to suggest an implementable differentially private algorithm, which would fit general industry environments.

1 Introduction

Collaborative filtering-based recommendations might leak sensitive information for some group of people, even though they provide only aggregated and anonymized information of the crowd. In recent years, recommendation systems have became vital components in daily web services such as the DVD recommendation system in *Netflix*, the book recommendation system in *Amazon.com*, and the friend recommendation system in *Facebook*. They assist customers to find suitable products, items or users from vast number of products and users, as increasing number of products sometimes distracts customers from choosing their exact tastes. Such recommendation systems are generally based on "collaborative filtering", which uses information from multiple agents or users. In collaborative filtering, each data point in a dataset represents an individual object's or user's record. For the state-of-the-art recommendation systems, the range of data used is not limited to a simple transaction data, but also covers user profiles and social connections to improve recommendation quality. For example, social link recommendation systems such as "People You May Know" in *Facebook* use social connection data of users and user profiles, and financial solution recommendations such as *Mint.com* utilize users' bank accounts and balance information. Due to the huge varieties and complexities of training data, a tiny privacy breach of recommendation outputs might reveal immensely sensitive information.

Previous privacy attack cases confirm that the privacy concern for recommendation systems is not merely a symptom of growing privacy alarmism. Information privacy, or data privacy, is gaining rapidly growing attention, as many data products or data publications, which might leak some of sensitive information, are becoming an inevitable choice of system designs. Data privacy issues arise in a wide range of applications, such as healthcare records, criminal justice investigations, financial institutions and transactions, biological traits, etc. In many privacy incidents, traditional privacy protection methods focused on the existence of personally identifiable information such as name or address [20]. However, k-anonymity or anonymization are shown to be vulnerable to data base link attacks e.g. *Netflix* prize example [18]. Furthermore, even for anonymized social network data, the recent results [19] [17] show that individuals can be identified just by utilizing network structure and moderate amount of side information.

In this project, we derive and analyze "differentially private" recommendation systems, and provide experimental results using our proposed frameworks. Differential privacy [6] is a recently proposed privacy measure or privacy definition. By its definition, any particular record in data cannot affect the outcome of algorithms or queries to the extent of $\exp(\epsilon)$, where ϵ being a predefined privacy parameter. Although differential privacy does not provide perfect privacy guarantee, the definition of differential privacy enables us to easily adjust the trade-off between utility and risk in privacy protecting frameworks. Besides, differential privacy offers several beneficial properties such as "composability" and "group privacy". Our main focus is to suggest an implementable differentially private algorithm, which would fit general industry environments.

The rest of the paper is organized as follows: We begin by reviewing the definition of differential privacy and two general methods of implementing differential private algorithms, then we provide several examples on privacy attacks in recommendation systems. In Section 3, we approach the problem of item-to-item based recommendation systems. We start from deriving the sensitivities of outputs, then we propose a novel interpretation of ϵ, the privacy parameter. We provide experimental results using the MovieLens dataset for the item-to-item framework. In Section 4, we visit more complicated and useful recommendation systems, user-to-item recommendation settings. We outline several attempts tried during this project, and describe our Markov Chain Monte Carlo (MCMC)-based output perturbation scheme. The MCMC-based perturbation is a general framework that does not depend on recommendation algorithms, and can be applied in many other domains. Finally, we discuss the possible constraints of our framework and future work in Section 5.

2 Related Work

In this section, we outline three bodies of related works, starting from the definition of ϵ-differential privacy. Differential privacy is nearly accepted standard privacy measure in theoretical computer science communities, and we detail out two methods, which achieve ϵ-differential privacy. Then, we provide recent literature on privacy attacks in recommendation systems. Finally, existing differentially private recommendation system will be discussed, highlighting main differences from our framework.

2.1 Differential Privacy

The amount of privacy is inherently difficult to measure as it involves mostly subjective opinions and decision rules. Differential privacy [6] is a recently proposed privacy measure or privacy definition to overcome such difficulty. The definition of differential privacy is strict and concise. By its definition, any particular record in data cannot affect the outcome of algorithms or queries to the extent of $\exp(\epsilon)$, where ϵ being a predefined privacy parameter.

Definition 1. *A randomized function $\mathcal{A}$ gives ϵ-differential privacy if for all datasets $D,\ D' \in \mathcal{D}$ differing on at most one row, and all $S \subseteq Range(\mathcal{A})$,*

$$Pr[\mathcal{A}(D) \in S] \leq exp(\epsilon) Pr[\mathcal{A}(D') \in S] \tag{1}$$

where the probability space in each case is over the coin flips of $\mathcal{A}$.

The parameter ϵ controls the level of privacy. Lower values of ϵ mean stronger privacy and vice versa. The definition of differential privacy directly leads to a composability property [7]: t consecutive ϵ-differential private queries are $(t\epsilon)$-differential private. Therefore, the value of ϵ is sometimes treated as a privacy cost for a specific query. However, the value of ϵ lacks concrete physical or visible interpretation, and determining the optimal value of ϵ has been inherently difficult. As a part of this project, in Section 3.3, we suggest alternative interpretation of ϵ by making it tangible in recommendation systems.

For numeric outputs, differential privacy can be typically achieved by adding calibrated noise to a true outcome of a query [8]. To add calibrated noise, we first define the sensitivity of a function:

Definition 2. *(Sensitivity). For $f:\ \mathcal{D} \rightarrow \mathbb{R}^d$, the sensitivity of f is:*

$$\Delta f = \max_{D,D'} \parallel f(D) - f(D') \parallel_1 \tag{2}$$

where $D,\ D' \in \mathcal{D}$ and $|D \ominus D'| \leq 1$.

Given the sensitivity of a function f, Theorem 2.1 provides the simplest implementation of ϵ-differential privacy.

Theorem 2.1. *(Additive Laplace Noise). For $f:\ \mathcal{D} \rightarrow \mathbb{R}^d$, the mechanism $\mathcal{A}$ that adds independently generated noise with distribution $Lap(\Delta f/\epsilon)$ to each of the d output terms enjoys ϵ-differential privacy.*

For non-numeric outputs, differential privacy can be achieved by Exponential mechanism [15]:

Theorem 2.2. *(Exponential Mechanism). $\phi : \mathcal{D} \times \mathbb{R} \rightarrow \mathbb{R}$ such that $|\log(\phi(D,z)) - \log(\phi(D',z))| \leq |D \ominus D'|$, where $D,\ D' \in \mathcal{D}$, then the mechanism defined by $Pr(\mathcal{A}(D) = s) = \phi(D,s)^\epsilon / \sum_{s'} \phi(D,s')^\epsilon$ satisfies 2ϵ-differential privacy.*

Exponential mechanism can be applied to non-numeric outputs such as text by incorporating an appropriate scoring function $\phi(\cdot,\cdot)$. Moreover, adding Laplace noise can be derived as a special case of Exponential mechanism by choosing an appropriate $\phi(\cdot,\cdot)$. Both methods of achieving differential privacy will be used extensively in this project: Laplace noise in item-to-item framework and Exponential mechanism in user-to-item framework.

2.2 Privacy Attacks in Recommendation Systems

Many recommendation systems are based on collaborative filtering, which uses information from multiple agents or users. As each data point in a collaborative filtering dataset represents an individual object or user, anonymization or aggregation is usually accompanied when releasing the data. However, the *Netflix* prize provides a well-known privacy attack in this case. In 2006, Netflix, an online DVD-rental service, opened a competition for a ten percent improvements in its recommendation system, and released a training dataset with random user IDs, anonymization. However, Narayanan and Shmatikov linked the Internet Movie Database (IMDB) dataset with the Netflix dataset [18], then successfully de-anonymized the Netflix dataset. Although a commercial recommendation system does not disclose its raw data in many cases, a recent work by Calandrino *et al.* [3] shows that passive observations of Amazon.com's recommendations are vulnerable to privacy attacks.

The most concrete and recent privacy attack is demonstrated in [2]. In [2], the paper highlights that there exists vast amount of side information that can be used to make privacy attacks. For example, Hunch.com provides its item-to-tiem correlation matrix through API. Moreover, some users publicly reveal their tastes or behaviors through the web. This side information is utilized to make the privacy inference in the paper. Using this moderate amount of side information, for item-to-item recommendation systems, 6 inferences per user resulted in 90% accuracy in the LibraryThing experiment. The results in the paper show that privacy leakage from recommendation systems is not merely over-amplified anxiousness, but actual threat that can be executed in real life.

2.3 Differentially Private Recommendation System

Originally, differential privacy is designed for a statistical database, and several statistical queries are analyzed in this framework such as counting queries, contingency tables and maximum likelihood estimates [7] [9]. Applicability of differential privacy is not limited to only simple structured queries. In [10], the sensitivities of

several splitting criteria in decision trees, such as Information gain and Gini index, are derived and analyzed. Chaudhuri *et al.* [4] [5] suggested differentially private logistic regression, also providing a novel objective perturbation technique. In [21], probabilistic graphical model is analyzed in the differential private framework with an efficient Exponential mechanism.

For recommendation systems, McSherry *et al.* suggested differentially private data publication for *Netflix* contenders [16]. However, the outputs of the algorithm is not for general users but for algorithm designers who build recommendation systems. Contrast to vast literature of differential private machine learning techniques, to the best of our knowledge, our work is the first differential private recommendation system with direct output perturbation. One possible reason of the sparse previous literature would be that the objective functions in the state-of-the-art recommendation systems are usually not convex and even if it is convex, its learning algorithm usually finds local minima e.g. Expectaion-Maximization algorithm. This brings difficulties in deriving the sensitivities of the objective functions. Furthermore, output formats are usually different from internal objectives, so that multiple steps of differential private frameworks should be considered. In our work, instead of focusing on sophisticated learning algorithms, we start from a basic item-to-item recommendation system, which is simple but performs almost the same as more complicated ones. Then we move to relatively sophisticated user-to-item recommendation systems. A main difference from other differentially private machine learning algorithms is that we rather focus on only the outputs of the algorithms, not the algorithms themselves, to provide a general output perturbation framework in recommendation systems.

3 Item-to-item framework

In this section, we introduce a differentially private item-to-item recommendation system. We first visit the Slope One item-to-item recommendation algorithm, which has been successfully adopted in various industry domains, then we derive the sensitivity of the Slope One algorithm and its variations. After providing a differentially private item-to-item recommendation algorithm using additive Laplace noise, we propose a novel practical interpretation of ϵ, a parameter of differential privacy, to find the optimal ϵ value in practice. Finally, we demonstrate our results using the MovieLens dataset.

3.1 Slope One algorithm

Item-to-item collaborative filtering (or item-based collaborative filtering) has many variations depending on how an item similarity matrix is formed. Among many variations, Slope One [13] is arguably the simplest form of non-trivial item-based collaborative filtering based on ratings. Its simplicity and scalability are readily applicable to many industry domains, and its performance is often par with more complicated and computationally expensive algorithms [1]. We first introduce the Slope One algorithm as we mainly focus on rating-based recommendation systems, then extends to other item-based collaborative filtering algorithms.

In collaborative filtering, we typically assume that user-to-item rating history or purchase history are given. For a rating history matrix $\mathbf{R}$ for N users and M items, r_{ij} represents its element, which means a user i rated an item j as r_{ij}. Note that a rating history matrix $\mathbf{R}$ is an incomplete matrix. In the Slope One algorithm, an item similarity matrix, $\mathbf{S}$, is constructed as follows:

$$s_{jk} = \sum_{i \in \Phi(j,k)} \frac{r_{ij} - r_{ik}}{|\Phi(j,k)|} \tag{3}$$

where s_{jk} represents a similarity measure between an item j and an item k, $\Phi(j,k)$ is a set of users who rated both item j and k and $|\Phi(j,k)|$ means its cardinality. When $\Phi(j,k)$ does not have enough number of users, to prevent over-fitting, the similarity matrix can be modified as:

$$s_{jk} = \begin{cases} \sum_{i \in \Phi(j,k)} \frac{r_{ij} - r_{ik}}{|\Phi(j,k)|} & \text{when } |\Phi(j,k)| > \phi_m \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where ϕ_m is the minimum number of users. After constructing $\mathbf{S}$, for a user i who already rated $|\mathbf{r}_{i\cdot}|$ number of items where $\mathbf{r}_{i\cdot}$ is a set of rated items for a user i, an estimated ratings for an unrated item j is given as:

$$\hat{r}_{ij} = \frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{k \in \mathbf{r}_{i\cdot}} (s_{jk} + r_{ik}) \tag{5}$$

$$= \bar{r}_{i\cdot} + \frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{k \in \mathbf{r}_{i\cdot}} s_{jk} \tag{6}$$

where $\bar{r}_{i\cdot} = \frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{j \in \mathbf{r}_{i\cdot}} r_{ij}$. Equation (6) can be understood as the sum of a user bias ($\bar{r}_{i\cdot}$) and an averaged item similarity between the unrated item j and the already rated items, $\frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{j \in \mathbf{r}_{i\cdot}} s_{jk}$. Actually, the similarity

measure between an item j and an item k, Equation (3), can be obtained from fitting a linear regression equation $f(x) = x + b$ where x being existing ratings, and the name Slope One comes from this property.

The similarity matrix $\mathbf{S}$ in Slope One is a special case of many other item-based recommendation systems. For example, if user-to-item purchase history $\mathbf{P}$ is given, where $p_{ij} = 1$ denotes a user i purchased an item j and $p_{ij} = 0$ vice versa, a similarity matrix can be formed using cosine similarity measure:

$$s_{jk} = \frac{\vec{p}_{\cdot j} \cdot \vec{p}_{\cdot k}}{|\vec{p}_{\cdot j}||\vec{p}_{\cdot k}|} \tag{7}$$

where $\vec{p}_{\cdot j}$ is an item j's purchase history vector. Note that, in this case, $\mathbf{P}$ is treated as a complete matrix. After building $\mathbf{S}$, future purchase estimates for $p_{ij} = 0$ can be given as:

$$\hat{p}_{ij} = \frac{1}{|\mathbf{p}_{i\cdot}|} \sum_{k \in \mathbf{p}_{i\cdot}} s_{kj} \tag{8}$$

where $\mathbf{p}_{i\cdot} = \{p_{ij} |\ p_{ij} = 1\}$. This algorithm is basically a simple version of Amazon's item-to-item patented algorithm [14]. Although there is a huge freedom of choosing a similarity measure in item-to-item recommendation systems such as Pearson correlation coefficient, in this project, we mainly focus on these two algorithms.

3.2 Sensitivity Analysis

For the Slope One algorithm, the output function $f_{ij}(\mathbf{R})$ for a user i and an item j is:

$$f_{ij}(R) = \bar{r}_{i\cdot} + \frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{k \in \mathbf{r}_{i\cdot}} s_{jk}. \tag{9}$$

Consider two rating matrices $\mathbf{R}$ and $\mathbf{R}'$, which differ on at most one element. Then, the sensitivity of the Slope One algorithm is now:

$$\Delta f = \max_{\mathbf{R},\mathbf{R}',i,j} \parallel f_{ij}(\mathbf{R}) - f_{ij}(\mathbf{R}') \parallel_1 \tag{10}$$

where $\| \cdot \|_1$ denotes $L1$ norm. In practice, to prevent over-fitting, a recommendation is provided to a user who rated at least T items.

Theorem 3.1. *Given rating matrices $\mathbf{R}$ and $\mathbf{R}'$, where each user has at least T number of ratings, the sensitivity of the Slope One output is:*

$$\Delta f \leq \max(\frac{\Delta r}{T}(1 + \frac{2}{\phi_m}),\ \frac{\Delta r}{\phi_m}) \tag{11}$$

where $\Delta r = r_{max} - r_{min}$.

Proof. Without loss of generality, we divide the problem into two cases: (i) when a user i has an additional rating on an item l in $\mathbf{R}'$ and (ii) when another user has an additional rating in $\mathbf{R}'$.

For the first case:

$$\begin{aligned}
\parallel f_{ij}(\mathbf{R}) - f_{ij}(\mathbf{R}') \parallel_1 =& \parallel \bar{r}_{i\cdot} + \frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{k \in \mathbf{r}_{i\cdot}} s_{jk} - \bar{r}'_{i\cdot} - \frac{1}{|\mathbf{r}'_{i\cdot}|} \sum_{k \in \mathbf{r}'_{i\cdot}} s'_{jk} \parallel_1 \\
=& \parallel (\bar{r}_{i\cdot} - \bar{r}'_{i\cdot}) + (\frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{k \in \mathbf{r}_{i\cdot}} s_{jk} - \frac{1}{|\mathbf{r}'_{i\cdot}|} \sum_{k \in \mathbf{r}'_{i\cdot}} s'_{jk}) \parallel_1 \\
\leq& \parallel \bar{r}_{i\cdot} - \bar{r}'_{i\cdot} \parallel_1 + \parallel \frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{k \in \mathbf{r}_{i\cdot}} s_{jk} - \frac{1}{|\mathbf{r}'_{i\cdot}|} \sum_{k \in \mathbf{r}'_{i\cdot}} s'_{jk} \parallel_1 \\
\leq& \frac{\Delta r}{T} + \parallel \frac{s'_{jl}}{|\mathbf{r}'_{i\cdot}|} + (\frac{1}{|\mathbf{r}_{i\cdot}|} - \frac{1}{|\mathbf{r}'_{i\cdot}|}) \sum_{k \in \mathbf{r}_{i\cdot}} s_{jk} \parallel_1 \\
\leq& \frac{\Delta r}{T} + \parallel \frac{s'_{jl}}{|\mathbf{r}'_{i\cdot}|} \parallel_1 + \parallel (\frac{1}{|\mathbf{r}_{i\cdot}|} - \frac{1}{|\mathbf{r}'_{i\cdot}|}) \sum_{k \in \mathbf{r}_{i\cdot}} s_{jk} \parallel_1 \\
\leq& \frac{\Delta r}{T} + \frac{\Delta r}{T\phi_m} + \frac{\Delta r}{T\phi_m} \qquad (12) \\
=& \frac{\Delta r}{T}(1 + \frac{2}{\phi_m}). \qquad (13)
\end{aligned}$$

Equation (12) follows from $\frac{1}{|\mathbf{r}_{i\cdot}|} - \frac{1}{|\mathbf{r}'_{i\cdot}|} = \frac{1}{|\mathbf{r}_{i\cdot}||\mathbf{r}'_{i\cdot}|}$. The rest of equations are derived from the fact that s_{jk} and s'_{jk} are the same except s_{jl} and s'_{jl}, plus triangle inequality of $L1$ norm.

For the second case:

$$\begin{aligned} \| f_{ij}(\mathbf{R}) - f_{ij}(\mathbf{R}') \|_1 =& \| \bar{r}_{i\cdot} + \frac{1}{|\mathbf{r}_{i\cdot}|} \sum_{k \in \mathbf{r}_{i\cdot}} s_{jk} - \bar{r}'_{i\cdot} - \frac{1}{|\mathbf{r}'_{i\cdot}|} \sum_{k \in \mathbf{r}'_{i\cdot}} s'_{jk} \|_1 \\ =& \| \frac{1}{|\mathbf{r}_{i\cdot}|} (\sum_{k \in \mathbf{r}_{i\cdot}} (s_{jk} - s'_{jk})) \|_1 \\ \leq & \frac{\Delta r}{\phi_m}. \end{aligned} \tag{14}$$

Taking the maximum for both cases, Theorem 3.1 follows. □

Similarly, the output function $f_{ij}(\mathbf{P})$ of the cosine similarity-based recommendation for a user i and an item j and its sensitivity are defined as:

$$f_{ij}(\mathbf{P}) = \frac{1}{|\mathbf{p}_{i\cdot}|} \sum_{k \in \mathbf{p}_{i\cdot}} s_{kj} \tag{15}$$

$$\Delta f = \max_{\mathbf{P}, \mathbf{P}', i, j} \| f_{ij}(\mathbf{P}) - f_{ij}(\mathbf{P}') \|_1 . \tag{16}$$

Theorem 3.2. *Given purchase matrices* $\mathbf{P}$ *and* $\mathbf{P}'$*, where each user has at least* T *number of purchases, the sensitivity of the cosine similarity-based recommendation output is:*

$$\Delta f \leq \frac{2}{T}. \tag{17}$$

Proof. Without loss of generality, let's consider two cases: (i) when a user i additionally purchased an item l in $\mathbf{P}'$ and (ii) when another user purchased an item l in $\mathbf{P}'$.

For the first case:

$$\begin{aligned} \| f_{ij}(\mathbf{P}) - f_{ij}(\mathbf{P}') \|_1 =& \| \frac{1}{|\mathbf{p}_{i\cdot}|} \sum_{k \in \mathbf{p}_{i\cdot}} s_{kj} - \frac{1}{|\mathbf{p}'_{i\cdot}|} \sum_{k \in \mathbf{p}'_{i\cdot}} s'_{kj} \|_1 \\ =& \| \frac{s'_{lj}}{|\mathbf{p}'_{i\cdot}|} + (\frac{1}{|\mathbf{p}_{i\cdot}|} - \frac{1}{|\mathbf{p}'_{i\cdot}|}) \sum_{k \in \mathbf{p}_{i\cdot}} s_{kj} \|_1 \\ \leq & \| \frac{s'_{lj}}{|\mathbf{p}'_{i\cdot}|} \|_1 + \| (\frac{1}{|\mathbf{p}_{i\cdot}|} - \frac{1}{|\mathbf{p}'_{i\cdot}|}) \sum_{k \in \mathbf{p}_{i\cdot}} s_{kj} \|_1 \\ \leq & \frac{1}{T} + \| (\frac{1}{|\mathbf{p}_{i\cdot}|} - \frac{1}{|\mathbf{p}'_{i\cdot}|}) \sum_{k \in \mathbf{p}_{i\cdot}} s_{kj} \|_1 \\ \leq & \frac{2}{T}. \end{aligned} \tag{18}$$

For the second case:

$$\| f_{ij}(\mathbf{P}) - f_{ij}(\mathbf{P}') \|_1 \leq \frac{1}{T}. \tag{19}$$

Thus, Theorem 3.2 is proved. □

Having obtained the sensitivities for both recommendation frameworks, we can add Laplace noise parametrized by $(\Delta f / \epsilon)$. From Theorem 2.1, this mechanism guarantees ϵ-differential privacy.

3.3 Practical Interpretation of ϵ

The privacy parameter ϵ in differential privacy controls the level of protection: a low value of ϵ means strict protection and vice versa. However, this lacks practical choice of ϵ as there is no visible gain and loss. Thus, we propose a novel practical interpretation of the parameter ϵ, making it tangible in the domain of recommendation systems. We first define ideal adversaries:

Definition 3. *For* ϵ*-differentially private algorithm* $\mathcal{A}^\epsilon(\cdot)$*,* ***$\epsilon$-ideal adversaries*** *are a set of functions* $\{g_{\mathcal{S}^{info}}\}$*, where* $\mathcal{S}^{info}$ *denotes side information derived from* $\mathcal{D}$ *(*$\mathcal{D} \rightarrow \mathcal{S}^{info}$*), such that* $\forall\ d \in \mathcal{D}$*,*

$$\lim_{\epsilon \to \infty} Pr[d = g_{\mathcal{S}^{info}}(\mathcal{A}^\epsilon(\mathcal{D}))] = 1 \tag{20}$$

$$\lim_{\epsilon \to 0} Pr[d = g_{\mathcal{S}^{info}}(\mathcal{A}^\epsilon(\mathcal{D}))] = \textit{Random Guessing}. \tag{21}$$

In other words, ideal adversaries are a set of functions, which can reconstruct any individual record when noise is not added to outputs. However, any function, which reconstructs individual records, is not necessarily a member of ideal adversaries, as when the maximum amount of privacy protection is imposed, $\epsilon \to 0$, its performance should be the same as random guessing. The number of ideal adversaries can be immensely large. Next, we define a specific member of ideal adversaries.

Definition 4. ***ϵ-Linearly Ideal Adversary*** *(LIA) is a family of Ideal enemies, such that:*

$$LIA(\mathcal{A}^{\epsilon}(\mathcal{D})) = a\mathcal{A}^{\epsilon}(\mathcal{D}) + b \tag{22}$$

where $a,\ b \in \mathcal{S}^{info}$.

Theorem 3.3. *For the Slope One algorithm, given an estimated rating of an item j for a user i, $r_{il} \in \mathbf{r}_{i\cdot}$ can be revealed by:*

$$LIA_{il}(\mathcal{A}^{\epsilon}_{ij}) = |\mathbf{r}_{i\cdot}|\mathcal{A}^{\epsilon}_{ij}(\mathbf{R}) - (\sum_{k \in \mathbf{r}_{i\cdot}} s_{jk} + \sum_{k \in \mathbf{r}_{i\cdot} \setminus r_{il}} r_{ik}). \tag{23}$$

Proof. Note that $\mathcal{A}^{\epsilon}_{ij}(\mathbf{R}) = f_{ij}(\mathbf{R}) + \delta$, where $\delta \sim Lap(\Delta f/\epsilon)$. Therefore,

$$LIA_{il} = |\mathbf{r}_{i\cdot}|(\mathcal{A}^{\epsilon}_{ij}(\mathbf{R}) - \frac{1}{|\mathbf{r}_{i\cdot}|}\sum_{k \in r_{i\cdot}} s_{jk}) - \sum_{k \in \mathbf{r}_{i\cdot} \setminus r_{il}} r_{ik} \tag{24}$$

$$= r_{il} + |\mathbf{r}_{i\cdot}|\delta. \tag{25}$$

When $\epsilon \to \infty$, $Pr(\delta = 0) = 1$. Thus, $\lim_{\epsilon \to \infty} Pr[r_{il} = LIA_{il}(\mathcal{A}^{\epsilon}_{ij}(\mathbf{R}))] = 1$. Also, when $\epsilon \to 0$, $LIA_{il} \sim Unif(-\infty, \infty)$, which satisfies the definition of ϵ-ideal adversaries. □

Theorem 3.4. *For the cosine similarity-based recommendation systems, given a purchase estimate of an item j for a user i, $p_{il} \in \mathbf{p}_{i\cdot}$ can be revealed by:*

$$LIA_{il}(\mathcal{A}^{\epsilon}_{ij}) = |\mathbf{p}_{i\cdot}|\mathcal{A}^{\epsilon}_{ij}(\mathbf{P}) - \sum_{k \in \mathbf{p}_{i\cdot} \setminus p_{il}} s_{kj} \tag{26}$$

where $p_{il} = 1$ when $LIA_{il} = s_{lj}$ and vice versa.

Proof. As in the proof of Theorem 3.3,

$$LIA_{il} = s_{lj} + |\mathbf{p}_{i\cdot}|\delta. \tag{27}$$

When $\epsilon \to \infty$, $Pr(\delta = 0) = 1$. Thus, $\lim_{\epsilon \to \infty} Pr[s_{lj} = LIA_{il}(\mathcal{A}^{\epsilon}_{ij}(\mathbf{P}))] = 1$. Also, when $\epsilon \to 0$, $LIA_{il} \sim Unif(-\infty, \infty)$, which satisfies the definition of ϵ-ideal adversaries. □

Although, from the definition of LIA, it might seem that too much of information is required in practice, partial side information of LIA sometimes can be obtained through public API's or other sources. For example, Hunch.com provides its item-to-item correlation matrix through API. In practice, a sophisticated inference technique should be accompanied with LIA, as only partial side information might be available. Thus, the definition of LIA can be viewed as the skeleton form of the privacy attack algorithms described in [2].

The definition of LIA enables us to derive two useful concepts, which help to visualize implicit utility and risk.

Definition 5. ***ϵ-Risk*** *is $E[Pr[d = LIA(\mathcal{A}^{\epsilon}(\mathcal{D}))]]$, where $d \in \mathcal{D}$.*

Definition 6. ***ϵ-Utility*** *is $1 - \frac{dist(\mathcal{A}^{\epsilon}(\mathcal{D}), \mathcal{A}^{\infty}(\mathcal{D}))}{\max dist(\mathcal{A}^{\epsilon}(\mathcal{D}), \mathcal{A}^{\infty}(\mathcal{D}))}$.*

ϵ-Risk is the expected privacy attack accuracy (or privacy leakage) of LIA, and ϵ-Utility is the performance difference from non-privacy algorithm $\mathcal{A}^{\infty}(\mathcal{D})$.

3.4 Experimental Results

We demonstrate our results for the Slope One algorithm using the MovieLens[1] dataset. A similarity matrix $\mathbf{S}$ is formed using Equation (3) from the training set. To prevent over-fitting, we use $\phi_m = 10$ while constructing the similarity matrix. For each user in the test set, which has at least 20 previous recommendations ($T = 20$), we calculate estimated ratings for unrated items. As $(T, \phi_m, \Delta r) = (20, 10, 5)$ in this experiment, we have $\Delta f = 0.5$.

[1] http://www.movielens.org/

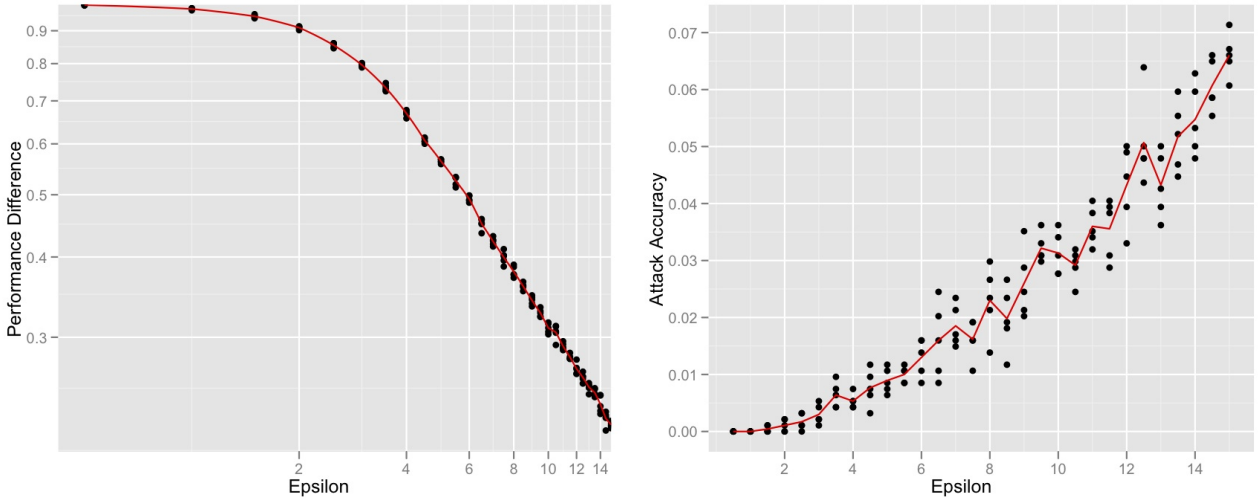

Figure 1: Differentially Private Item-to-item recommendation system. (a) Performance Difference from the non-private Slope One algorithm vs. ϵ (left) and (b) Attack Accuracy of LIA vs. ϵ (right).

Every output is noised using Laplace noise parametrized by $(\Delta f/\epsilon)$. We take top-20 from the estimated ratings, then compare them with the non-private Slope One top-20 results. The performance difference is measured as $\frac{\#\text{ of different items}}{20}$. Figure 1(a) shows the relationship between the performance difference and privacy parameter ϵ in log-log scale. As ϵ takes larger values, the performance difference approaches to zero.

Next, we implement LIA for each user in the test set. We assume that the side information needed to implement LIA is readily available. One rating from a user's rating history is randomly chosen, and the rating is attacked by using LIA. Figure 1(b) shows the graph between the attack accuracy of LIA vs. ϵ. We can observe that when $\epsilon \to 0$, the attack accuracy is close to zero. Moreover, as ϵ increases, the attack accuracy also increases. In theory, the attack accuracy should reach one when $\epsilon \to \infty$.

Finally, Figure 2 shows the plot of ϵ-Risk vs. ϵ-Utility. Risk-utility tradeoff in privacy protecting recommendation systems is well visualized. This curve can be utilized in determining the optimal value of ϵ. Suppose we have monetary values (weights) for both ϵ-Risk and ϵ-Utility. As the curve is convex, there exists a unique ϵ that satisfies the conditions. Also note that with very small perturbation of the outputs, one can reduce the attack accuracy a lot.

4 User-to-item framework

In this section, we will discuss about implementing differential privacy in user-to-item recommender system.

4.1 Categorization of Privacy Attacks in User-to-Item Recommendation System

In any recommendation system, given a sparse rating matrix $D \in \mathcal{D}$, a *learning algorithm* is employed to estimate a complete rating matrix $C \in \mathcal{C}$. This learning algorithm can be anything – row average, column average, matrix factorization like techniques [12] etc. Depending on these two matrices, privacy attack in a user-to-item recommender system can be categorized into three types:

- **Immediate Attack:** In immediate attack, a subset of "active" users (adversaries) change entries in the recommender system before the learning phase – *i.e.* they directly attack on D. The assumption here is that the recommender system allows users to change entires in D but not in C.
- **Dynamic Attack:** In dynamic attack, the adversaries change entries in the system after the learning phase – (*i.e.* they directly attack on C. Usually, in any large scale recommender system, the computationally challenging learning phase is deployed after every few days or weeks and in between two learning phases, users are allowed to impute entries in the complete rating matrix C. Therefore, dynamic attack is very common for large scale recommender systems (*e.g* see [2]).
- **Combined Attack:** In this case, a subset of the adversaries attack on D and others attack on C.

In this paper, however, we will only emphasize on dynamic attack and attacks of other two categories will be a subject of our future endeavor.

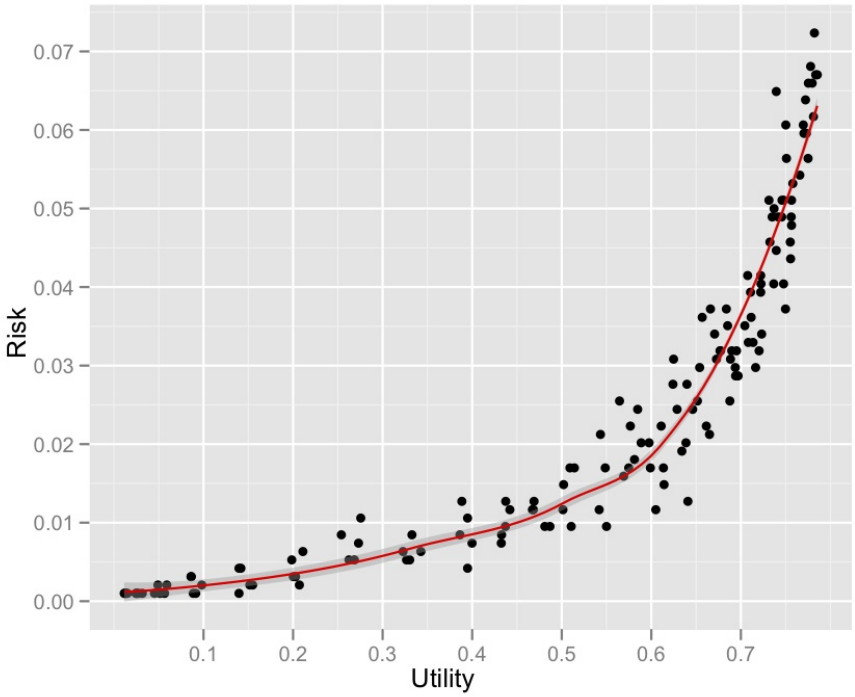

Figure 2: ϵ-Risk vs. ϵ-Utility. Note that the curve is connected from (ϵ-Utility, ϵ-Risk) = (0, 0) to (ϵ-Utility, ϵ-Risk) = (1, 1) by its definition. We can observe that the ideal risk rapidly drops down with a slight decrease in utility e.g. 70% utility has 4% risk.

In the user-to-item recommendation system, the aim is to preserve ϵ-differential privacy for the top-k recommended items. When privacy is not an issue, the optimal recommendation set $S_0^{(i)}$ for the i^{th} user should consist of k highest rated items. However, to achieve differential privacy, this optimal recommendation set is not always prescribed. Rather, a distribution is maintained over all possible ${}^{M}\mathbb{C}_k$ collection of k items from a pool of M items. Let this collection be denoted by $\mathcal{S}$ so that $N_{\mathcal{S}} = |\mathcal{S}| = {}^{M}\mathbb{C}_k$. In any standard recommendation system like *amazon.com* or *eBay.com*, M is typically of the order of thousands or more and the value of k typically lies between 10 to 20. This implies that $N_{\mathcal{S}}$ is too large to enumerate.

To enforce ϵ-differential privacy in dynamic attack, we maintain the following distribution over the elements of the set $\mathcal{S}$ for the i^{th} user:

$$P(S_j^{(i)}) = \frac{\phi(C, S_j^{(i)})^{\epsilon/2}}{\sum_{j'} \phi(C, S_{j'}^{(i)})^{\epsilon/2}}, \tag{28}$$

where $\phi(C, S_j)$ is some mapping from $\mathcal{C} \times \mathcal{S}$ to $\mathbb{R}^+$, $S_j^{(i)}$ being the j^{th} recommended set for the i^{th} user. Note that the normalization term involves summing over $N_{\mathcal{S}}$ terms which is not computationally feasible. We, therefore, propose a Markov chain (MC) to generate samples which, in limit, can approximate the distribution given in Equation (28).

4.2 Approximating the True Distribution

Let us first consider a continuous time Markov chain (CTMC) – the birth-death process – shown in Figure 3. The equilibrium state, $\boldsymbol{\pi}$, should satisfy the following flow balance equation:

$$\lambda_j \pi_j = \mu_{j+1} \pi_{j+1} \ \forall j \in \{0, 1, 2, \cdots\}. \tag{29}$$

So, the equilibrium distribution $\boldsymbol{\pi}$ can be expressed as:

$$\boldsymbol{\pi} = \pi_0 (1, \frac{\lambda_0}{\mu_1}, \frac{\lambda_0 \lambda_1}{\mu_1 \mu_2}, \frac{\lambda_0 \lambda_1 \lambda_2}{\mu_1 \mu_2 \mu_3}, \cdots). \tag{30}$$

With the following values of λ_j and μ_{j+1}:

$$\lambda_j = \frac{\phi(C, S_{j+1}^{(i)})^{\epsilon/2}}{\phi(C, S_j^{(i)})^{\epsilon/2}} \ , \ \mu_{j+1} = \frac{\phi(C, S_j^{(i)})^{\epsilon/2}}{\phi(C, S_{j+1}^{(i)})^{\epsilon/2}}; \tag{31}$$

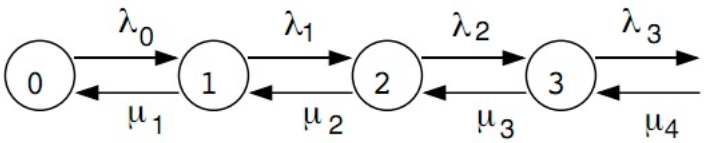

Figure 3: Birth-Death Process

the equilibrium distribution can be expressed as:

$$\boldsymbol{\pi} = \pi_0(1, \frac{\phi(C, S_1^{(i)})^\epsilon}{\phi(C, S_0^{(i)})^\epsilon}, \frac{\phi(C, S_2^{(i)})^\epsilon}{\phi(C, S_0^{(i)})^\epsilon}, \frac{\phi(C, S_3^{(i)})^\epsilon}{\phi(C, S_0^{(i)})^\epsilon}, \cdots). \tag{32}$$

Theorem 4.1. *If* $\boldsymbol{\pi}^* = \frac{1}{Z}(\phi(C, S_0^{(i)})^\epsilon, \phi(C, S_1^{(i)})^\epsilon, \phi(C, S_2^{(i)})^\epsilon, \cdots)$, *then* $\boldsymbol{\pi} = \boldsymbol{\pi}^*$ *where* $Z = \sum_j \phi(C, S_j^{(i)})^\epsilon$.

Proof.

$$\begin{aligned}
&\boldsymbol{\pi} = \boldsymbol{\pi}^* \\
\Rightarrow \pi_0(1, \frac{\phi(x,z_1)^\epsilon}{\phi(x,z_0)^\epsilon}, \frac{\phi(x,z_2)^\epsilon}{\phi(x,z_0)^\epsilon}, \frac{\phi(x,z_3)^\epsilon}{\phi(x,z_0)^\epsilon}, \ ...) &= \frac{1}{Z}(\phi(x,z_0)^\epsilon, \ \phi(x,z_1)^\epsilon, \ \phi(x,z_2)^\epsilon, \ ...) \\
\Rightarrow \pi_0(1, \frac{\phi(x,z_1)^\epsilon}{\phi(x,z_0)^\epsilon}, \frac{\phi(x,z_2)^\epsilon}{\phi(x,z_0)^\epsilon}, \frac{\phi(x,z_3)^\epsilon}{\phi(x,z_0)^\epsilon}, \ ...) &= \frac{\phi(x,z_0)^\epsilon}{Z}(1, \ \frac{\phi(x,z_1)^\epsilon}{\phi(x,z_0)^\epsilon}, \ \frac{\phi(x,z_2)^\epsilon}{\phi(x,z_0)^\epsilon}, \ \frac{\phi(x,z_3)^\epsilon}{\phi(x,z_0)^\epsilon}, \ ...) \\
&\Rightarrow \pi_0 = \frac{\phi(x,z_0)^\epsilon}{Z}
\end{aligned} \tag{33}$$

Therefore, proving $\boldsymbol{\pi} = \boldsymbol{\pi}^*$ is equivalent to proving $\pi_0 = \frac{\phi(C,S_0^{(i)})^\epsilon}{Z}$.

Suppose, $\pi_0 \neq \frac{\phi(C,S_0^{(i)})^\epsilon}{Z}$, so that $\pi_0 = \frac{\phi(C,S_0^{(i)})^\epsilon}{Z} + \delta$, where $\delta \neq 0$. This leads to $||\boldsymbol{\pi}||_1 \neq 1$, which contradicts the definition. Thus, $\boldsymbol{\pi} = \boldsymbol{\pi}^*$ is proved. □

Now, let's try to go back to our original problem. To convert the proposed CTMC to a discrete time MC (DTMC), we derive the transition probabilities according to the following equations:

$$\begin{aligned}
p_{j,j+1} &= \lambda_j \Delta t \\
p_{j,j-1} &= \mu_j \Delta t \\
p_{j,j} &= 1 - (\lambda_j + \mu_j)\Delta t \ \forall j \in \{1, 2, \cdots\}.
\end{aligned} \tag{34}$$

Additionally, for $j = 0$, we should have $p_{0,1} = \lambda_0 \Delta t$, $\mu_0 = 0$, $p_{j,j-1} = 0$ and $p_{0,0} = 1 - \lambda_0 \Delta t$. The derived DTMC has the same steady-state distribution as the CTMC [11]. Let $\bar{\boldsymbol{\pi}}$ be the steady-state distribution. Therefore:

$$\begin{aligned}
(\lambda_j + \mu_j)\Delta t \bar{\pi}_j &= \lambda_{j-1}\Delta t \bar{\pi}_{j-1} + \mu_{j+1}\Delta t \bar{\pi}_{j+1} \\
(\lambda_j + \mu_j)\bar{\pi}_j &= \lambda_{j-1}\bar{\pi}_{j-1} + \mu_{j+1}\bar{\pi}_{j+1}
\end{aligned} \tag{35}$$

The last line is the same as in the CTMC balance equation, Equation (29). So, if:

$$1 - (\lambda_j + \mu_j)\Delta t \geq 0 \tag{36}$$

is satisfied $\forall j \in \{0, 1, 2, \cdots\}$ along with the assumption that $\mu_0 = 0$, this DTMC, in limit, provides the same distribution given in Equation (28). Note that the equilibrium distribution of this DTMC is unique as the number of nodes is finite and the chain is irreducible and aperiodic (aperiodicity being ensured by the constraint, Equation (36)). We now consider the following proposition to propose an upper bound on the value of Δt. An approximate working value of Δt will be provided in Section 4.5.

Proposition 4.1. *If* $\frac{\phi(C,S_j^{(i)})}{\phi(C,S_{j+1}^{(i)})} \leq U_s/2 \ \forall j$ *and* $\phi(C, S_j^{(i)}) \geq \phi(C, S_{j+1}^{(i)}) \ \forall j$, *then* $0 < \Delta t \leq 1/U_s$ *satisfies condition 36* $\forall j$.

Proof. The proof for this is straightforward. Since $1 - (\lambda_j + \mu_j)\Delta t \geq 0$, $\Delta t \leq \frac{1}{(\lambda_j + \mu_j)}$. This implies that, $\Delta t \leq \min_{j \in \{0,1,2,\cdots\}} \frac{1}{(\lambda_j + \mu_j)}$. Now, $(\lambda_j + \mu_j) = \frac{\phi(C,S_{j-1}^{(i)})^\epsilon + \phi(C,S_{j+1}^{(i)})^\epsilon}{\phi(C,S_j^{(i)})^\epsilon} \leq \frac{2\phi(C,S_{j-1}^{(i)})^\epsilon}{\phi(C,S_j^{(i)})^\epsilon} \ \forall j \in \{1, 2, \cdots\}$. Therefore, $\frac{1}{(\lambda_j + \mu_j)} \geq \frac{\phi(C,S_j^{(i)})^\epsilon}{2\phi(C,S_{j-1}^{(i)})^\epsilon} \geq 1/U_s \ \forall j \in \{1, 2, \cdots\}$ (by assumption). Also, $1/\lambda_0 = \frac{\phi(C,S_0^{(i)})^\epsilon}{\phi(C,S_1^{(i)})^\epsilon} \geq 2/U_s > 1/U_s$. Hence, $0 < \Delta t \leq 1/U_s$ satisfies the condition, Equation (36) $\forall j$. □

4.3 Characterizing $\phi(.,.)$

The choice of the function $\phi(.,.)$ is completely application specific in exponential mechanism. However, as mentioned in the definition of exponential mechanism, $\phi(.,.)$ has to satisfy the constraint:

$$|\log(\phi(C, S_j^{(i)})) - \log(\phi(C', S_j^{(i)}))| \leq |C \ominus C'| \ \forall j \text{ and } \forall C, C' \in \mathcal{C}. \tag{37}$$

Based on this constraint, we present the following proposition.

Proposition 4.2. *Let $\boldsymbol{r}_{ij}^{(k)}$ be the set of ratings corresponding to the k elements (items) in the set $S_j^{(i)}$ for the i^{th} user. $f : \mathcal{C} \times \mathbb{R}^k \to \mathbb{R}$ is a function that gives the sum of the ratings in the vector $\boldsymbol{r}_{ij}^{(k)}$ derived from some matrix $C \in \mathcal{C}$. A function $\phi(C, S_j^{(i)})) = exp(1/U_r) f(S_j^{(i)})$ satisfies the condition mentioned in Equation (37) where U_r is the magnitude of the maximum possible rating allowed in the matrix C or C'.*

Proof. $|\log(\phi(C, S_j^{(i)})) - \log(\phi(C', S_j^{(i)}))| = (1/U_r)|f(C, S_j^{(i)}) - f(C', S_j^{(i)})| \leq 1 = |C \ominus C'|$, when C and C' differ only by one rating in the i^{th} row. □

4.4 Generating Sets $\{S_j^{(i)}\}_j$

In Proposition 4.1, the condition $\phi(C, S_j^{(i)}) \geq \phi(C, S_{j+1}^{(i)}) \ \forall j$ implies that we are interested in generating sets with decreasing values of $f(.,.)$ (and therefore of $\phi(.,.)$). This not only ensures that the sampling is limited to sets with very high values of $f(.,.)$ (and hence incurs less accuracy loss), but also provides an upper bound on the ratios of $\phi(C, S_j^{(i)})/\phi(C, S_{j+1}^{(i)})$ that can be computed in practice. However, generating sets that follow a strictly decreasing order imposed by the function $f(.,.)$ is challenging. If we want to ensure that $\nexists \ell$ such that $f(C, S_j^{(i)}) > f(C, S_\ell^{(i)}) > f(C, S_{j+1}^{(i)}) \ \forall j$, then the complexity of the generating process is at least $O(N_{\mathcal{S}} \log N_{\mathcal{S}})$ with $N_{\mathcal{S}} =^M \mathbb{C}_k$.

To avoid this extra cost of computation, we pursue an approximate method for generating sets with decreasing values of $f(.,.)$ *i.e.* $f(C, S_j^{(i)}) \geq f(C, S_{j+1}^{(i)}) \ \forall j$. Let $\boldsymbol{r}_i$ be the sorted (in decreasing order) rating vector for i^{th} user of size M. We will index the components of this vector in defining the sets in the following.

The set with the largest value of f is the one with the largest k ratings and is denoted by $S_0^{(i)}$. The next best set is $S_1^{(i)} = \{1, 2, \cdots, k-1, k+1\}$ – *i.e.* the one for which the k^{th} element is left out and $(k+1)^{\text{th}}$ is, instead, included from $\boldsymbol{r}_i$. For the next j^{th} set, we have $S_{j+1}^{(i)} = \{1, 2, \cdots, k-j_1, k-j_1+2, \cdots, k, k+j_2\}$ with $j_1 = (\lfloor (j-1)/N_2 \rfloor + 1) - \lfloor (\lfloor (j-1)/N_2 \rfloor + 1)/N_1 \rfloor$ and $j_2 = \lfloor (j-1)/(N_1 N_2) \rfloor N_2 + (j - \lfloor j/N_2 \rfloor)$. In essence, we maintain a window of size N_1 on the left of the k^{th} element in $\boldsymbol{r}_i$ and a window of size N_2 to the right of the k^{th} element. The window on the right shifts by N_2 elements towards right after every $N_1 N_2$ number of sets have been generated. This process will ensure that at least $\hat{N}_{\mathcal{S}} = \lfloor (M-k)/(N_1 N_2) \rfloor (N_1 N_2)$ high scoring sets are generated with complexity $O(\hat{N}_{\mathcal{S}})$. Additionally, N_1 and N_2 can be tuned for better performances.

One general comment on the entire process of approximating the true distribution is that, in limit, the sampling process is going to give us the true distribution. However, it is impossible to ensure this convergence with finite number of samples. Therefore, there will always be some gap between the true distribution P given in Equation (28) and the approximate distribution $\hat{P}$ so that $\delta = \text{KL}(P||\hat{P}) > 0$. Therefore, following definition, we will never be able to get exactly ϵ-differential privacy but some $(\epsilon + \Delta\epsilon)$-differential privacy where $\Delta\epsilon$ accounts for the gap $\delta > 0$. However, from our perspective, this is an issue of how privacy is defined and not an issue in practice. By drawing enough number of samples, we can get arbitrarily close to the true distribution. Also, following this mechanism, we never need to enumerate the sets in advance. A set is generated only when the sampled point resides in the last set generated so far.

4.5 Experimental Results

Figures 4 (a) and (b) show the results on the MovieLens dataset for the proposed exponential mechanism to recommend top-10 items. The sampling process described in Section 4.4 ensures that only one item gets changed in all the sets generated from the optimal set. So, the lower bound in accuracy is always going to be 90% for lower values of ϵ which is important from privacy perspective. As ϵ increases, the system becomes more and more non-private and the accuracy hits 100%. A computationally more expensive sampling algorithm, however, could have generated sets with varying number of overlaps with items in the optimal set and a smoother accuracy vs. ϵ curve could have been observed. We chose $N_1 = 2$ and $N_2 = 3$ in the results displayed.

In Figure 4 (b), we plot the KL divergence between the sampled distribution and the true distribution of the top $\hat{k}$ states. $\hat{k}$ here is the minimum between the value 10 and the number of sets explored (as the number of sets explored can be less than 10 for larger values of ϵ). It is clear, that the distribution converges towards zero. Particularly, for $\epsilon = 100$, the divergence is 0 since the beginning. In all of these experiments, we used

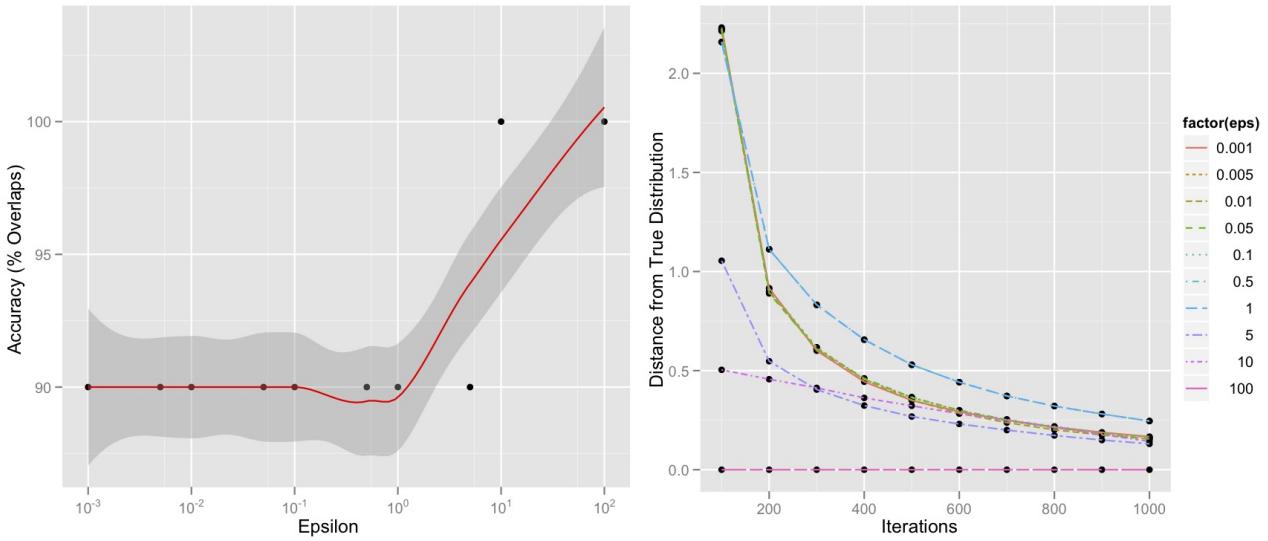

Figure 4: Differentially Private User-to-item recommendation system. (a) Accuracy (% overlaps) vs. ϵ (left) and (b) KL-divergence vs. Number of iterations (right).

$\Delta t = 1/U_s = 1/20$ except for the one with $\epsilon = 100$ where $\Delta t = 10^{-8}$ was used. In practice, the value of U_s can be set at $10\frac{\phi(C,S_0^{(i)})}{\phi(C,S_1^{(i)})}$. However, any arbitrary large value of U_s cannot be chosen as that would imply $p_{0,0} \to 1$ and no new set will be sampled for a significant amount of time prolonging the convergence of the sampling process to the true distribution. One, therefore, has to be careful in choosing the value of this parameter.

5 Discussion

In this project, we explored two major differentially private recommendation frameworks: (i) item-to-item and (ii) user-to-item collaborative filterings:

In the item-to-item case, we derived the sensitivities of the output functions, then added calibrated Laplace noise to the outputs. Even though our sensitivity analyses are confined in two similarity measures, which are (i) the Slope One and (ii) cosine similarities, we think the similar approaches can be applied to other similarity measures. For a more generalized differentially item-to-item recommendation system, the sensitivity bounds for general similarity metrics should be analyzed and we leave this as our future work. To visualize the gain and loss in this setting, we introduced ϵ-ideal enemies and ϵ-Linearly Ideal Enemy (LIA), and we presented the experimental results using the MovieLens dataset. The trade-off between utility and risk in the privacy preserving recommendation systems is successfully visualized, and we also hope this methodology would help to determine the optimal value of ϵ in real applications.

In the user-to-item case, instead of focusing on learning algorithms, we approached from the output domain. We categorized possible privacy attacks into three cases, then we propose a solution to the dynamic attack case. To sample from the exponentially large recommendation sets, an efficient MCMC-based sampling scheme is derived and its asymptotic behavior was analyzed. The proposed MCMC sampling mechanism has a wide range of applications in other non-numeric differentially private output formats e.g. English words, and we will try to generalize this framework in our future work. The experimental results using the same MovieLens dataset is provided to show the performance of the proposed sampling scheme. In our future work, a general defense mechanism for other kinds of attacks will be studied with (ϵ, δ)-differential privacy, which is a slightly relaxed version of ϵ-differential privacy.

Recommendation systems are now prevalent in various domains such as web commerce, search engines (e.g. related searches), financial solutions, etc. However, due to rapidly growing usage of sensitive information in them, their privacy aspects should not be overlooked. Even though we only presented two major settings of recommendation systems in this project, we believe their coverage of applications is large and general. For more sophisticated algorithms, which use multiple datasets or sources, we plan to investigate generalized output perturbation algorithms as an extension of this project.

Bibliography

[1] Fidel Cacheda, Victor Carneiro, Diego Fernandez, and Vreixo Formoso. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transactions on the Web*, 1(2), 2011.

[2] Joseph A. Calandrino, Ann Kilzer, Arvind Narayanan, Edward W. Felten, and Vitaly Shmatikov. You might also like: Privacy risks of collaborative filtering. In *IEEE Symposium on Security and Privacy*, pages 231–246, 2011.

[3] Joseph A. Calandrino, Arvind Narayanan, Edward W. Felten, and Vitaly Shmatikov. Don't review that book: Privacy risks of collaborative filtering. In *Manuscript*, 2009.

[4] Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems*, 2008.

[5] Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. Differentially private empirical risk minimization. *JMLR*, 2011.

[6] Cynthia Dwork. Differential privacy. In *in ICALP*, pages 1–12. Springer, 2006.

[7] Cynthia Dwork. Differential privacy: a survey of results. In *Proceedings of the 5th international conference on Theory and applications of models of computation*, TAMC'08, pages 1–19, Berlin, Heidelberg, 2008. Springer-Verlag.

[8] Cynthia Dwork, Frank Mcsherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *In Proceedings of the 3rd Theory of Cryptography Conference*, pages 265–284. Springer, 2006.

[9] Cynthia Dwork and Adam Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2):135–154, 2009.

[10] Arik Friedman and Assaf Schuster. Data mining with differential privacy. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 493–502, New York, NY, USA, 2010. ACM.

[11] Geoffrey Grimmett and David Stirzaker. *Probability and Random Processes, Chapter Six.* Oxford, 2001.

[12] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, August 2009.

[13] Danieal Lemire and Anna Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *SIAM Data Mining (SDM)*, 2005.

[14] Greg Linden, Brent Smith, and Jeremy York. Amzaon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[15] Frank Mcsherry. Mechanism design via differential privacy. In *Proceedings of the 48th Annual Symposium on Foundations of Computer Science*, 2007.

[16] Frank McSherry and Ilya Mironov. Differentially private recommender systems: building privacy into the net. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 627–636, New York, NY, USA, 2009. ACM.

[17] Arvind Narayanan, Elaine Shi, and Benjamin I. P. Rubinstein. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *IJCNN*, 2011.

[18] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 111–125, 2008.

[19] Arvind Narayanan and Vitaly Shmatikov. De-anonymizing social networks. In *IEEE Symposium on Security and Privacy*, 2009.

[20] Latanya Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10:557–570, October 2002.

[21] Oliver Williams and Frank McSherry. Probabilistic inference and differential privacy. In *Advances in Neural Information Processing Systems*, pages 2451–2459, 2010.