Exploring Some Approaches to Privatize Data Releases for Training Recommendation Systems

May 16, 2019

Cynthia Du, Alexander Leighton, and Shwetark Patel

***Abstract*—This paper explores and evaluates the effectiveness of two methods for privatizing the "Netflix Competition" dataset. Narayanan and Shmatikov [1] shows how an attacker might reidentify the user ids in the dataset and gain sensitive information. We examine two methods for preventing such an attack. One method is a simple ϵ locally differentially private mechanism. The other method comes from a more recent work [2], taking a different approach to privacy entirely. While the *local model* proved not to be an effective mechanism for privatizing the "Netflix Competition" dataset, the approach of distance-based privacy seemed to reach a balance between privacy and model accuracy for some parameter sets. While the approaches tested in this paper had varying levels of success, it would still seems as though the best approach to privatizing this sort of dataset is an approach similar to the one taken in McSherry Mironov [3], which we have not explored in this paper.**

I. Introduction

In the last decade, technology has become increasingly integrated in our personal lives. With this integration of technology comes convenience. Who doesn't want everything from entertainment to food to be just a few taps away?

At the same time, the vast amount of content on the internet starts pushing companies to incorporate personalization into their strategy to get us more engaged and to cater to the needs of advertisers. However, this personalization comes at the expense of user privacy.

In this paper, we attempt to grapple with this trade-off between user privacy and effective personalization of movie recommendations, and we explore two methods for finding a middle ground between the two.

II. Background

Providing recommendations has become a key part of many services as the amount of information available on their platforms increases faster than what a human can browse through in a reasonable amount of time.

It's crucial for these services, in particular streaming services, to keep on improving quality of their recommendations to remain competitive because sometimes their users don't even know what they really want. By giving them great recommendations, these services can prevent the users from leaving the platform when they've exhausted the shows they thought they wanted to watch and keep the users on their platform for longer periods of time.

The most popular choice for finding recommendations is collaborative filtering. Collaborative filtering makes suggestions based on people who are similar to a user. The basic idea behind it is pretty simple: if you have similar taste as someone else, you have a greater chance of wanting to watch what that person has also watched.

The downfall of having a recommender system like this is that by definition a recommender system based on collaborative filtering reveals to one user some amount of information about other users, and this can be especially dangerous when one user is malicious.

One thing that's great about collaborative filtering is that it enables serendipity (ability to recommend something unexpected but desirable). Someone who only watches videos related to cats may be pleasantly surprised when the video streaming service recommends videos on piano, which he happens to also like. Here the collaborative filtering recommender system is able to do so because it recognizes other people on its service that have rated both cat videos and piano videos.

But by the same reason that a collaborative filtering can enable serendipity, an adversary can create multiple accounts, check what selections enable a certain recommendation, and from there identify this kind of straddlers across different domains. This is especially dangerous when the adversary also has access to queries to the data base, which is reasonable because such sites often provide statistical reporting or provide database to consultants. Back to the example, this adversary may use a query to verify that only a small amount of people liked both cat and piano videos, and use queries to identify attributes of this group of people, like "What is the average age of people who like both cat and piano videos?" to get information he otherwise wouldn't have gotten. [8]

We conduct an analysis on the trade-off between user privacy and how useful a recommender system is with a case study of the 2008 "Netflix Prize" competition.

In order to improve on its user-specific recommendation system, Netflix held a competition to outsource finding an effective prediction algorithm for movie recommendations. To help contestants train their models, Netflix chose a 10% sample of all users and divided the reviews of those users into a 100 million rating train set and a 3 million rating test set. To protect user privacy, Netflix removed the names of users from the training set, assigned customer ids that range from 1 to 2649429 (with gaps - there are 480189 users).

One team from AT&T, "BellKor's Prgramatic Chaos" took away the 1 million prize.

Along with that, though, came a class action lawsuit against Netflix, claiming Netflix had breached federal privacy law. Even though Netflix removed usernames and assigned gener-

ated user ids, researchers were able to de-anonymize by cross referencing Netflix's released data with IMDB data [1].

In the end, Netflix had to cancel the second prize competition it was going to have.

This raises a question: can companies ever release "real" sensitive training data? The answer is both yes and no. Any release of "real" data run the risk of reconstruction by a determined and informed adversary.

However, there are steps companies like Netflix can take to mitigate these risks.We will provide a case study on two approaches to privatize the Netflix data. These approaches have differential privacy guarantees that prevent similar attacks.

III. Applying Differentially Private Mechanisms to the "Netflix Competition" Dataset

A. Differential Privacy

As a baseline for our subsequent methods and analysis, we revisit the definition of formal differential privacy.

Definition (ϵ-Differential privacy)**.** A mechanism M satisfies ϵ-differential privacy if for all adjacent datasets D_1 and D_2, and all $S \subseteq Range(R)$, the following inequality holds:

$$Pr[M(D_1) \in S] \leq e^{\epsilon} \cdot Pr[M(D_2) \in S]$$

Definition ((ϵ, δ)-Differential privacy)**.** A mechanism M satisfies (ϵ, δ)-differential privacy if for all adjacent datasets D_1 and D_2, and all $S \subseteq Range(R)$, the following inequality holds:

$$Pr[M(D_1) \in S] \leq e^{\epsilon} \cdot Pr[M(D_2) \in S] + \delta$$

Typically, and in our specific case of the "Netflix Competition" data, D_1 and D_2 can be thought of as databases where each row contains a given user's data.

Formal privacy of a mechanism M, guarantees us that the perturbed data output by M has a statistically similar chance of coming from D_1 as it does from an adjacent dataset, D_2.

Specifically to the "Netflix Competition" data, each row of D corresponds to a user and each column corresponds to a user's 1 through 5 rating of that movie or 0 if the user did not rate the movie. For obvious reasons this dataset is very sparse.

One of our objectives for this project was to test various mechanisms, M, taking D as an input and outputting a noisy version of D, $M(D) = \hat{D}$, such that M is differentially private and $\hat{D}$ is statistically similar to D.

Finding such a mechanism M would allow statistical analysis of the *real* data, while preserving users' privacy to within a certain guaranteed, statistical threshold.

B. Local Model

Definition (Local Model)**.** If every row in a release M of a database D is ϵ-Differentially private, then the entire release is ϵ-Differentially Private.

One way to make a one row database ϵ-differentially private is a randomized response mechanism with the total privacy budget distributed evenly across movies. Let$D_{i,j}$ be user i's review of movie j. We privatize reviews by substituting each $D_{i,j}$ with $M(D) = \hat{D}_{i,j}$ as follows:

$$\hat{D}_{i,j} = \begin{cases} D_{i,j} & \text{w.p. } \frac{e^{\epsilon_k}}{1+e^{\epsilon_k}} \\ \overline{D}_{i,j} & \text{w.p. } \frac{1}{1+e^{\epsilon_k}} \end{cases}$$

where

$$\overline{D}_{i,j} = \text{random}([0..5] \setminus D_{i,j})$$

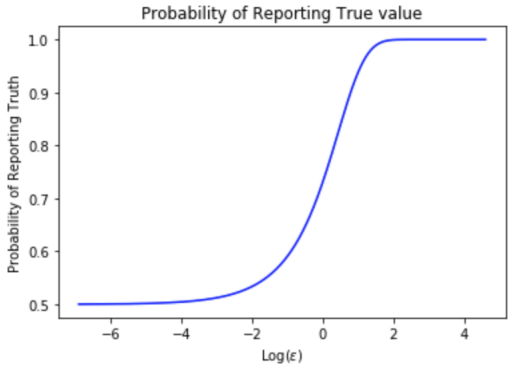
and,

$$\epsilon_k = \frac{\epsilon}{\#\text{movies}}$$

In the *local model* for privacy, the mechanism is applied to every row rather than the dataset as a whole, or specific algorithms on the data. Therefore, more noise is induced.

However, we reap the added benefit of privatizing the entire dataset, regardless of use case. We can allow blanket access to a dataset privatized in this way, and an attacker still would not be able to gain private information beyond some threshold.

The *local model* in this sense can be thought of as a sort of gold standard for this sort of data release: If the *local model* does not add too much noise to the data, then we release $\hat{D}$ and preserve privacy while giving full, unfettered access to the privatized data.



C. Distance-Based Differential Privacy

The standard differential privacy ensures that an adversary can't guess whether a user has rated a particular movie.

However, since a lot of movies are alike, we want to prevent the adversary from even guessing whether the user rated any movies that are similar to a particular movie. We realized that we needed a stronger notion of differential privacy[2]. Let $\mathcal{GRP}_{\lambda}(x)$ be the set of items that have distance less than λ with x,; and for a set S, let $\mathcal{GRP}_{\lambda}(S)$ be the union of $\mathcal{GRP}_{\lambda}(s)$ for elements $s \in S$.

Definition (Distance-based differential privacy)**.** For any two adjacent profile sets D_1 and D_2, where $\mathcal{U}$ denotes any arbitrary user and S denotes any possible subset of elements, then any mechanism $\mathcal{R}$ is (ϵ, λ)-private if the following inequality holds:

$$\frac{Pr[\mathcal{R}(D_1, \mathcal{U}) \in \mathcal{GRP}_{\lambda}(S)]}{Pr[\mathcal{R}(D_2, \mathcal{U}) \in \mathcal{GRP}_{\lambda}(S)]} \leq e^{\epsilon}$$

Distance-based DP [2] comes to rescue.It creates *AlterEgo* profiles of users that still promise security while ensuring the quality of a collaborative filtering recommender system.

Our differential privacy guarantee will depend on λ, which we can alter to balance privacy and accuracy.

A notion of distance between two movies is used to measure how similar they are.

1) First, we form a group G_i for item i that consists of all other items j such that the distance from i to j is less than λ (For distance, we can use many different metrics, including cosine distance and squared Cartesian distance. Personally, we used squared distance in our algorithm).

2) In the selector stage, for each item i, we replace it with a possible item at random with probability $1-p$ and replace it with a randomly chosen item from group G_i with probability p.

3) In the profiler stage, we build the *AlterEgo* profile. In this stage, we replace an item specified in selector stage with probability $1-p^*$ and keep the item as is with probability p^*.

In order to apply this methodology to Netflix data, we found for each movie the set of similar movies to it using a similarity function (squared Cartesian distance, though other functions could also be applied with different λ values) and a constant λ.

Then, we selected a representative for the movie by selecting from the set of similar movies with probability p and from the set of all movies with probability $1-p$.

Finally, for each movie-user pair, we replaced the movie in that movie-user pair with its representative with probability $1-p^*$ and kept it the same with probability p^*.

IV. Analysis

A. Dataset

Due to the original "Netflix Competition" dataset being so large and time constraints, in our analysis we consider a representative subset of the data.

We subset as follows: Keep the 1000 movies with the most nonzero ratings and ignore the rest. Keep the ratings for the 1000 users who have left nonzero ratings for the most of these 1000 movies, and ignore the rest. This reduces our data matrix D to 1k by 1k, instead of 480k by 17k, something necessary due to time constraints and lack of access to high power machines used by groups competing in the original competition.

We note that in our approach we reduce the sparsity of D; however, we deemed the benefit of capturing the maximum possible amount of data in our sub-sample to outweigh the cost of changing the structure of the data.

B. Finding recommendations

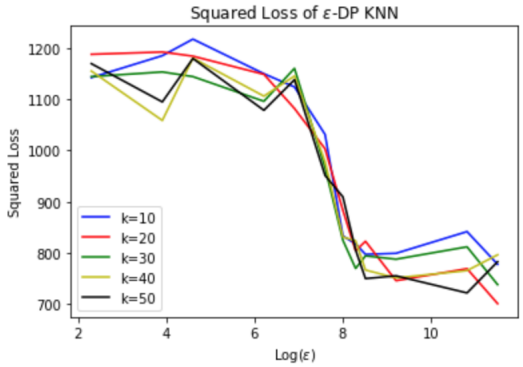
In order to test accuracy of the data after we privatize it, we will use the K-Nearest Neighbors approach [**?**]. We test accuracy by, for each user, selecting a random movie and attempting to predict the user's rating for that movie by looking at other users.

We go through all other users and compare their ratings to the user's for all movies but the randomly selected one using a similarity function. We then determine the K most similar users (i.e. the K ones with the best similarity metrics with the similarity function) and average their ratings on the randomly selected movie to predict the user's vote on the movie.

Finally, we choose squared loss between the predicted rating and the actual rating of the movie to evaluate the preformance of our algorithm.

C. Local Model

Our analysis show the *local model* adds an unacceptable amount of noise to the "Netflix Competition" data. While there are some values of ϵ for which the level of noise is feasible and the definition of ϵ-differential privacy is met, those values of ϵ are too large to guarantee any sort of practical privacy.



Indeed, the values of ϵ for which KNN squared loss starts to near that of the baseline are the ones where hardly any of the entries in the database are being modified, and, therefore, the mechanism is not protecting users privacy.

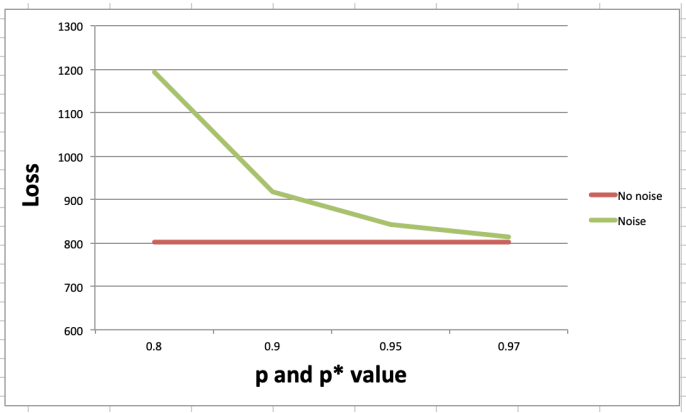
D. Distance based

In order to determine which parameters were best suited to ensure that the Netflix data was both accurate enough and retained reasonable levels of differential privacy, we chose to vary the value of K (the number of nearest neighbors selected in the recommender algorithm), p, and $p*$.

We found that varying K had almost no effect on accuracy. Meanwhile, varying p and p* had significant effects on accuracy, as shown in the following table:

p	p*	Loss of baseline	Loss of DP data with noise	Percent increase in loss
0.8	0.8	800.47	1193.31	49.1%
0.90	0.90	800.47	917.44	14.6%
0.95	0.95	800.47	843.11	5.06%
0.97	0.97	800.47	813.75	1.66%

As well as the following plot:



Here, loss is measure by taking the squared distance between the actual user votes vector and predicted votes vector (which was initially generated using KNN).We define the decrease in accuracy from adding noise to be the increase in loss from the baseline data (i.e. no noise) to the DP data (the baseline data with noise added).

Note that noise is added to approximately $100 \times (1 - p*)$ percent of the data. Thus, with $p* = 0.8$, noise is added to approximately 20% of the data while with $p* = 0.97$ noise is added to approximately 3%.

Note that with $p* = 0.8$, accuracy is affected greatly (it takes a 49.1% hit), while with $p* = 0.97$, accuracy is barely affected. This represents the trade-off between differential privacy and accuracy: Adding more noise, and thus making the data more differentially private, lowers accuracy.

This idea is further emphasized by the data in the table, as we can clearly see accuracy decreasing as more noise is added.

In terms of optimal p and $p*$ values, we would recommend using p and $p*$ both between 0.90 and 0.95, simply because larger p and $p*$ risk compromising differential privacy while smaller p and $p*$ compromise accuracy too much.

However, the best combination of accuracy and differential privacy is subjective and we could see other choices of p and $p*$ being used (for example, if Netflix doesn't care much about accuracy, they could use $p = p* = 0.8$).

V. Discussion

A. Limitations of Local Model

The disadvantages of the local model when compared with other models such as the one employed by McSherry and Mirvonov [3] is that by privatizing the dataset itself, rather than an algorithm, it necessarily induces more noise.

The advantage of local differential privacy, however, is data scientists can work hands on with a noisy version of the data, without needing any special understanding of privacy.

More effective approaches than the local model would likely take into account the whole of the data structure, and not attempt to privatize individually each row of a database with many thousands of rows.

However, this naive approach shows us the sensitivity of classifiers like KNN to even the slightest addition of noise.

As seen with the local model, to achieve a reasonable ϵ we almost completely randomize our data. And likewise, to achieve reasonable results in our recommender, we must increase ϵ to the point where we have an incredibly weak privacy guarantee.

For these reasons, the local model does not seem to be an effective privacy mechanism for the "Netflix Competition" data, or realistically, releasing any sort of large dataset. The reason for this being we can get far tighter bounds on privacy using other approaches that scale much better with the size of the data than does the local model.

B. Limitations of Distance-Based DP

While Distance-Based DP seems like an effective approach to this specific problem, we must consider possible ramifications of using an approach which does not guarantee formal privacy.

In the case of the "Netflix Competition", Distance-Based DP seems a reasonable metric to counter the immediate threat of identifying whether a user has viewed a certain category of movie, but it is not the solve-all-solution that is (ϵ, δ)-Differential Privacy.

In our opinion this is where Distance-Based DP fails. While Distance-Based DP does protect specifically against the de-anonymization attack shown in Narayanan and Shmatikov [1], it gives us weaker blanket guarantees against unforseen, future attacks than does (ϵ, δ)-DP.

C. Future work

In this project, we evaluated the accuracy of data with randomized response differential privacy and with distance-based differential privacy using K-Nearest-Neighbor collaborative filtering.

When evaluating similarity between movies and similarity between users, we used the normal cartesian distance. Possible future work could include testing a variety of distance metrics, such as Pearson correlation or Cosine similarity instead. In all cases, as we did with cartesian distance, each user (or movie) is represented by a vector.

Pearson correlation method finds the linear correlation between two vectors, which ranges from -1 to 1. A Pearson correlation close to 1 means there is a high positive correlation between the two items.

Cosine similarity measure the similarity by finding the cosine of the angle between the two vectors.

Note, however, these three vector-based methods don't quite capture similarity between users quite well when a user just comes on board - we get a cold start problem where the user's vector would be close to the null vector [10].

In addition, it would be interesting to see how the two approaches would affect the accuracy of the collaboration filtering method if, instead of using K-Nearest -Neighbor which is based on the similarity among users, we use item-based collaborative filtering - another memory-based techniques, or even use model-based collaborative filtering like Bayesian belief nets CF or clustering CF that improve prediction performance [9].

Furthermore, applying these techniques to other datasets released by companies and checking whether the resulting data assists in creating differential privacy without impacting accuracy would be interesting. Another extension is to explore which values of p, p^*, and K are optimal for these other datasets and check whether there are universally optimal parameters for the distance based differential privacy algorithm.

VI. Conclusions

Using differential privacy is important for companies like Netflix that want to protect user's privacy while providing good recommendation system or releasing user data (even anonymized ones) and aggregate statistics. In this project, we explored two different approaches for privatizing data: Local-DP and distance based.

For an ϵ-differentially private local-DP approach, we see the squared loss drop dramatically as the log of ϵ increases. However, before we can achiever reasonable accuracy with this approach, we must decrease the privacy parameter ϵ to a level which would be unacceptable for any practical application.

While distance-based privacy uses a notion of differential privacy that prevents an adversary from guessing movies that are some distance away from a certain movie of a user profile, we still have some skepticism of it as a metric for entry based privacy, rather than user based privacy: in other words, it protects netflix users from people finding out they have watched a given movie, or one like it, but not from someone finding out they watch netflix or that they were part of the dataset. While the former definition might be the "problem" from Netflix's perspective, ultimately as students of differential privacy, we would like to see the sort of user level privacy detailed in McSherry and Mironov [3].

Overall, it became clear to us how one trades privacy for accuracy when considering how much noise to add. While it is a great tool, we suggest anyone who's adding noise to data with differential privacy guarantee to carefully consider the trade-off between the two.

References

[1] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. pp 111–125. In SP, 2008.

[2] Guerraoui, Rachid & Kermarrec, Anne-Marie & Patra, Rhicheek & Taziki, Mahsa. (2015). D2P: Distance-Based Differential Privacy in Recommenders.

[3] F. McSherry and I. Mironov. Differentially private recommender systems: Building privacy into the net. pages 627–636. In SIGKDD, 2009.

[4] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, Theory ofCryptography Conference—TCC 2006, volume 3876 of Lecture Notes in Computer Science, pages 265–284. Springer, 2006.

[5] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science, 9(3–4):211–407, 2014

[6] Netflix Prize Rules: https://www.netflixprize.com/ rules.html. Accessed on May 1, 2019.

[7] Netflix Prize Dataset: https://www.kaggle.com/ netflix-inc/netflix-prize-data. Downloaded on April 27, 2019.

[8] N. Ramakrishnan, B. J. Keller, B. J. Mirza, A. Y. Grama, and G. Karypis. Privacy risks in recommender systems. pages 54–62. In Internet Computing, 2001.

[9] X. Su, T. M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques. 2009.

[10] A.Agarwal, M.Chauhan. Similarity Measures used in Recommender Systems: A Study

[11] M. Archie, S. Gershon, A. Katcoff, A. Zeng, Who's Watching? De-anonymization of Netflix Reviews using Amazon Reviews.