

CS 1699: Privacy in the Electronic Society  
*Project 3 – Data Anonymization*

Jonathan Dyer

April 22, 2018

## Contents

<b>Task W0</b>	<b>2</b>
Netflix Prize Data . . . . .	2
<b>Task W1</b>	<b>2</b>
Potentially Sensitive Information . . . . .	2
Re-identification through Quasi-Identifiers . . . . .	2
<b>Task C2</b>	<b>3</b>
<b>Task W3</b>	<b>3</b>
<b>Task C4</b>	<b>3</b>
<b>Task W5</b>	<b>4</b>
<b>Task C6</b>	<b>4</b>
<b>Task W7</b>	<b>4</b>
<b>References</b>	<b>5</b>

## Task W0

### Netflix Prize Data

For this project I selected the well-known Netflix prize dataset, acquired from Academic Torrents (see also [3]). This is a set of over 100 million movie ratings that was provided publicly to facilitate a contest hosted by Netflix to improve their movie recommendation system. The movie preferences of nearly 500 thousand Netflix users are represented, with the database including (for each of the 17770 movies):

1. The (randomly) assigned `CustomerIDs` of users who rated that movie.
2. The ratings (on a 5-star scale) of each of those users for that movie.
3. The date that rating was given on.

The data was collected from 1998 to 2005 and was released in 2006 with all personal identifying information having been purportedly removed [4].

## Task W1

This seemingly innocuous dataset actually has real implications for the privacy of those whose information is contained therein. What we are concerned with is *not* whether everyone in the dataset has their privacy at risk – that is almost certainly not the case. Rather we wish to see that *some* (or even one) of the individuals in the set may have their privacy violated by the data contained in this release. This violation has been demonstrated in several academic papers, including [1] and others.

### *Potentially Sensitive Information*

The privacy violation in question is the public disclosure of any Netflix user's viewing history, including that content which the user has not publicly avowed consuming or given a public opinion on. While at first this may seem not to qualify as sensitive, consider the inferences that may be made about someone's political opinions, religious stance, or sexual preferences based on their strong (ostensibly private) ratings of media content with those or related themes. In an individual's workplace or certain social settings, these would all be considered sensitive issues, making the possibility of their public disclosure a gross violation of privacy. In this particular data set that makes the UserID in some way the sensitive information, because linking an individual to that would reveal the viewing history as discussed herewith.

### *Re-identification through Quasi-Identifiers*

In this dataset, we consider that the following columns can act as quasi-identifiers:

- **Movie Title:** The knowledge that someone has viewed a particular film (out of thousands) is somewhat identifying, in that it certainly restricts the possible set of users to which they belong (since likely no one has viewed all 17770 films in the dataset). It is also relatively easy information for an adversary to acquire independently, especially if they are acquainted with the victim. In particular, it is worth noting that the less popular the movie (say, not in the top 100 or even 500), the greater value it is in helping to identify a viewer, since less people overall have viewed that movie.
- **User Rating:** The review a user has given to a particular movie can serve as a quasi-identifier, helping again to reduce the number of possible "anonymous" user IDs that might be associated with an individual. As with movie title, this information may be simple for an adversary to acquire (movie opinions are often freely shared, especially about popular or controversial films).
- **Date Reviewed:** Information about when a user gave a movie a particular rating could be useful in restricting the search once more, since some users frequently choose to rate movies soon after they view them on Netflix. This information could be garnered again in a casual conversation, and so could quite feasibly be available side knowledge to a potential attacker.

The above quasi-identifiers have been used, in conjunction with publicly available data on IMDb, to de-anonymize some viewing histories from this very dataset. See [2] for more information.

## Task C2

See the file `database_setup.py` for a script that transforms this dataset into an SQLite database. This is then easily imported into the ARX Data Anonymization program [6], where it is straightforward to implement  $k$ -anonymity for a given  $k$  along with  $l$ -diversity for desired  $l$ . This is done by indicating `UserID` as 'sensitive' and all other fields as 'quasi-identifying'. It is then possible to define hierarchies for each attribute so that they can be generalized during the anonymization process, and after selecting '3 anonymity' and 'distinct 2 diversity' as options we may anonymize the set, resulting in a set like the data in **W3**. See the next section for more details.

## Task W3

A sample of the dataset generated by **C2** is included. It satisfies 3-anonymity on the quasi-identifiers above along with 2-diversity on the "rating" attribute, meaning that for any unique set of quasi-identifiers matching a given individual in the set, there are a minimum of 2 *other* people that share those quasi-identifiers. Moreover, there are at least 2 different types of results in the "rating" column.

**Privacy Improvements** This dataset reveals less than the previous because a particular UserID may no longer be uniquely associated with a date and a particular rating on some movie. Now the exact rating is less certain and the date range is generalized, so that a user may fit into a larger category which reveals less potentially sensitive information about their entire viewing history.

**Remaining Disclosures** This is a simplified example, so in actuality the anonymity set for this data (when generalized as described) was much larger. However, there are still many ways that information may still be revealed, especially for a less popular film. Specifically, 2-diversity is not an especially strict privacy model to enforce. Just because there are two different sensitive values in some group of the dataset does not mean that it will be difficult to distinguish an individual from that group, especially if the ratings are very different and the adversary knows generally how the target feels about the movie in question. Consider the following example chart modeled after the transformed (anonymized) data. Although it technically satisfies the criteria above, it is clear that a casual conversation with an individual about the movie *Dinosaur Planet* (MovieID 1) could distinguish which of the three userIDs belongs to them.

UserID	MovieID	Date Reviewed	Ranking
1488844	1	2005	{3}
822109	1	2005	{4,5}
1009622	1	2005	{1,2}

Thus we see that this privacy model is not perfect. While it does obscure the ranking that a user gave to a particular film and the date it occurred, it does *not* hide the fact that the users with those IDs watched the movie, at least enough to want to *leave a rating*. So existence privacy (or **message flow privacy**) is not guaranteed, even if the user's opinions are kept private.

## Task C4

See the accompanying file `single_insight.py` for a script that indicates how many users disliked the movie *Dinosaur Planet* (MovieID 1). Note that this script supposes that `database_setup.py` has already been run on the appropriate dataset, and that relative pathnames are constant. This script accesses the SQLite database and queries it for the selected information. It is easy to modify to retrieve other information, or information about different movies as well. It returns an integer which answers the question "How many users gave the movie 'Dinosaur Planet' a rating of 1 star?" (The response for this script should be '28'.)

## Task W5

The above insight – a count of how many users hated *Dinosaur Planet* – does not satisfy differential privacy. Recall that for any two neighboring datasets  $D$  and  $D'$  (i.e.  $D'$  can be obtained from  $D$  by changing one single tuple), we have the following definition:

**Definition.** A randomized algorithm  $A$  satisfies  $\varepsilon$ -differential privacy, iff for any two neighboring datasets  $D$  and  $D'$  and for any output  $O$  of  $A$ ,

$$\frac{\Pr(A(D) = O)}{\Pr(A(D') = O)} \leq \exp(\varepsilon)$$

So we see that an attacker should not be able to tell the difference between these two datasets with probability above some certain bound, or put another way, the result of the query shouldn't depend too strongly on any particular tuple in the input [8]. The raw count data returned by the `single_insight.py` script is sensitive to the presence or absence of any single user. That is, for an attacker who has knowledge of the other rows in  $D$ , computing a sum on the database  $D'$  will reveal exactly what the extra (or missing) individual's data is. Thus, the left hand side of the equation above would not fit below any reasonably small parameter  $\varepsilon$ .

## Task C6

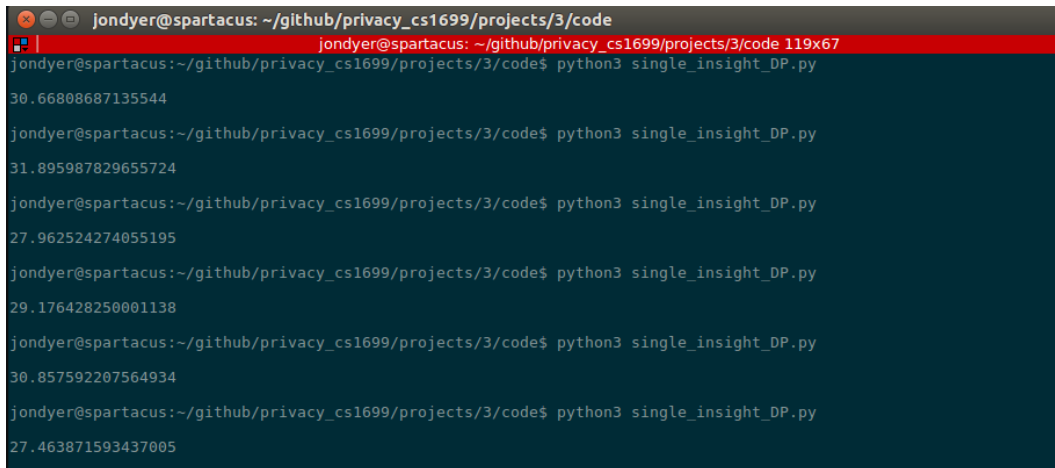
See the accompanying file `single_insight_DP.py` for a script that indicates how many users disliked the movie *Dinosaur Planet* (MovieID 1). Note that this script supposes that `database_setup.py` has already been run on the appropriate dataset, and that relative pathnames are constant.

In the given version of this script, I use the  $\varepsilon$  value of  $\ln(3)$ , giving a scale parameter of  $\frac{1}{\varepsilon} = \frac{1}{\ln(3)}$  for the random Laplacian noise. This can of course be changed – I chose this value because it matches the differential privacy of a **randomized response** algorithm, and seemed like a good example value.

## Task W7

Several results of the improved DP script are shown below, satisfying  $\varepsilon$ -differential privacy for  $\varepsilon = \ln(3)$  in this case. This is achieved by first noticing that a function that returns the result of a counting query has  $l_1$ -sensitivity of 1 (as defined in the fantastic privacy resource [5]). This sensitivity indicates that adding noise scaled to  $\frac{1}{\varepsilon}$  to our result will provide the desired  $\varepsilon$ -differential privacy.

This method protects users more than **C4** because now for any given dataset  $D$  with some user in it, the probability that the query (i.e. algorithm  $A$  in our definition) will return some result is no more than the probability the dataset  $D'$  (without the user) will return the same result, scaled by a factor of  $\exp(\varepsilon)$ . In other words, it is much harder to tell (depending on the parameter  $\varepsilon$ ) whether or not a given individual is in a dataset or not, protecting that individual's privacy more than in the naive implementation.



```
jondyer@spartacus: ~/github/privacy_cs1699/projects/3/code
jondyer@spartacus: ~/github/privacy_cs1699/projects/3/code 119x67
jondyer@spartacus:~/github/privacy_cs1699/projects/3/code$ python3 single_insight_DP.py
30.66808687135544
jondyer@spartacus:~/github/privacy_cs1699/projects/3/code$ python3 single_insight_DP.py
31.895987829655724
jondyer@spartacus:~/github/privacy_cs1699/projects/3/code$ python3 single_insight_DP.py
27.962524274055195
jondyer@spartacus:~/github/privacy_cs1699/projects/3/code$ python3 single_insight_DP.py
29.176428250001138
jondyer@spartacus:~/github/privacy_cs1699/projects/3/code$ python3 single_insight_DP.py
30.857592207564934
jondyer@spartacus:~/github/privacy_cs1699/projects/3/code$ python3 single_insight_DP.py
27.463871593437005
```

Finally, it is worth noting that each time this query result is returned, *more information* is returned, and never will somehow *less* information be leaked in total (as noted very well in [7]). Thus, some value of  $\varepsilon$  that is robust or provides some differential privacy for a single query may **not** be sufficient for repeated queries of the same type.

## References

- [1] Arvind Narayanan and Vitaly Shmatikov. “How to break anonymity of the netflix prize dataset”. In: *arXiv preprint cs/0610105* (2006).
- [2] Arvind Narayanan and Vitaly Shmatikov. “Robust de-anonymization of large sparse datasets”. In: *Security and Privacy, 2008. SP 2008. IEEE Symposium on*. IEEE. 2008, pp. 111–125.
- [3] Netflix. “Netflix Prize Data Set”. In: (2009). URL: <http://archive.ics.uci.edu/ml/datasets/Netflix+Prize>.
- [4] Netflix. *Netflix Prize: FAQ*. 2009 (accessed April 21, 2018). URL: <https://netflixprize.com/faq.html>.
- [5] Cynthia Dwork, Aaron Roth, et al. “The algorithmic foundations of differential privacy”. In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.
- [6] Fabian Prasser et al. “Arx – A comprehensive tool for anonymizing biomedical data”. In: *AMIA Annual Symposium Proceedings*. American Medical Informatics Association. 2014, p. 984.
- [7] Matthew Green. “What is Differential Privacy?” In: *A Few Thoughts on Cryptographic Engineering*. 2016. URL: <https://blog.cryptographyengineering.com/2016/06/15/what-is-differential-privacy/>.
- [8] Xiaokui Xiao. “Privacy Preserving Data Publishing: From k-Anonymity to Differential Privacy”. In: *Tutorial at IEEE International Conference on Data Science and Cyberspace*. IEEE. 2016. URL: <http://www.itee.uq.edu.au/dke//filething/get/953/Xiao-privacy.pdf>.