Brussels Faculty of Engineering

Université Libre de Bruxelles

Vrije Universiteit van Brussel

Academic Year 2017-2018

Project Report

Data De-Anonymisation of the Netflix Prize dataset

Cédric Hannotier, Mathieu Petitjean, Hasan Can Yildirim

ELEC-Y591: Machine Learning and Big Data Processing

Contents

List of Figures

1 Introduction

Operators of social networks as well as companies are increasingly sharing information about their users. Would it be to support research or for commercial purposes, related data is typically protected by anonymization. Often, this "anonymization" is carried out by removing sensitive fields such as the name, address or Social Security Number of the user. Still, the scientific community has expressed doubt as to whether those methods guaranteed effective user privacy. Several successful attacks have been demonstrated, and this report aims to reproduce (with limitations such as reduced computing power capabilities) one of the most famous of those privacy breaches: the Netflix Prize dataset de-anonymization.

This report is structured as follows. First, in section 2, a state of the art of de-anonymization techniques is presented. It summarizes the major existing attacks, then focuses on the Netflix case. Secondly, in section 3, our approach is detailed and our choices are described. Eventually, in section 4, our results are shown, analyzed, and compared to the state of the art.

2 State of the art

2.1 De-anonymization attacks

One of the most mentioned de-anonymization deeds dates back to 2006, when New York Times journalists identified Thelma Arnold in the "anonymized" search queries released by AOL for research purposes [1]. By manually searching in the 20 millions search queries coming from 657,000 users, the reporters could tie Arnold's identity to some quite embarrassing queries.

The computer-aided attacks, being able to push such results to a much larger scale, can be separated in several categories depending on their approach [2]. The two most represented methods are:

- **Graph matching.** It is the most common approach in the case of social network de-anonymisation studies and is based on social graphs. One meaningful example is in [3], where Flicker and Twitter accounts were linked together with a 12% error rate. Several complex strategies can be used to improve graph matching, such as Seed & Grow [4] or Threading [5].
- **Similarity matching.** It is based on similar features between the target and auxiliary information. In [6], users were de-anonymised using the similarity between tweets and the content of their resume. In [7], victims of homicides were re-identified using "anonymous" homicides public records of Chicago and records in the Social Security Death Index.

2.2 The Netflix case

The attack that will be reproduced is the one presented in [8], an example of similarity matching. In this paper, researchers attacked a dataset released by Netflix in the context of a contest to improve their recommendation system. The 100 millions movie ratings by over 480,000 users were correlated to another movie rating database: the Internet Movie Database (IMDb). The goal was to link together private Netflix accounts and public IMDb accounts based on the published ratings. In a very small sample of the IMDb (50 users only), 2 users of the Netflix dataset were identified with statistical quasi-certainty. As the authors summarized, given a few of an user's reviews that he chose to make *public*, their algorithm is able to access all of his *private* Netflix ratings.

The algorithm is based on a similarity measure denoted Sim. It is defined, for two records r_1 and r_2, with $supp$ denoting the non-null attributes of a record:

$$Sim(r_1, r_2) = \frac{\sum Sim_{\cos}(r_{1i}, r_{2i})}{|supp(r_1) \cup supp(r_2)|} \tag{1}$$

With $Sim_{\cos}$ denoting the cosine similarity measure, the function Sim maps the records r_1 and r_2 to an interval $[0, 1]$, representing the notion of them being similar. This concept now needs to be adapted to the specific content of a movie review dataset. In particular, the scoring function needs to give higher importance to statistically rare attributes. Indeed, a review on "The Longest Most Meaningless Movie in the World[1]" helps identify a user much more than the knowledge of the fact that he liked the last episode of "Game of Thrones". Also, the two pieces of information that are available for a given review are the score given by the user and the time-stamp of the rating. The final scoring function that was used in [8] is:

$$Score(r, aux) = \sum_{i \in supp(aux)} \frac{1}{\log|supp(i)|} \left(e^{-\frac{|\rho_i - \rho'_i|}{\rho_0}} + e^{-\frac{|d_i - d'_i|}{d_0}} \right) \tag{2}$$

In the $Score$ function that compares a record r and auxiliary information aux, ρ and ρ' denote the score given to the same movie, while d and d' refer to the date of the rating. ρ_0 and d_0 are constants empirically determined to respectively 1.5 and 30. The closest the ratings and the timestamps are, the higher the scoring function will be.

The fact whether a match has been found does not rely solely on the search for the entry in aux that has the highest score. Indeed, this only indicates which entry is the most similar but does not take into account how strong is the similarity. This is managed by imposing that two entries from r and aux are considered a match only if the difference between the best and second-best scores is higher than a threshold, referred to as the

[1] See its Wikipedia page: https://en.wikipedia.org/wiki/The_Longest_Most_Meaningless_Movie_in_the_World

eccentricity ϕ. Mathematically, it is defined as:

$$\phi = \frac{S_1 - S_2}{\sigma_S} \tag{3}$$

Where S_1 and S_2 denote respectively the best and second best score for a record r, and σ_S denotes the standard deviation of all the scores related to the record r. It was proposed in [8] that a match is considered to be found if $\phi > 1.5$. It is worth noting that the two matches with the 50 samples from IMDb had an eccentricity of respectively 28 and 15. These especially high numbers lead to the belief that two matches were found with statistical quasi-certainty.

In [9], theorems that formally demonstrate why the scoring expressed by Equation 2 works on the Netflix dataset were introduced. In addition to providing a mathematical framework, they propose another scoring algorithm, slightly less performing but based on more general assumptions.

In both [8] and [9], the matching algorithm is the same (only the metrics that are used differ) and it is summarized in Algorithm 1.

Algorithm 1 Matching algorithm based on weighted scale scoring.

Starting from datasets R and aux

for each record r_i in R **do**
    for each entry aux_i in aux **do**
        Compute $Score(r_i, aux_i)$
    end for
    Compute $\sigma_S =$ `stdev(`$Score$`)`
    Find $S_1 = \max(Score(r_i, aux))$
    Find $S_2 = \max(Score(r_i, aux) \backslash \{S_1\})$
    Compute $\phi = (S_1 - S_2)/\sigma_S$
    if $\phi > 1.5$ **then**
        Match found !
    end if
end for

3 Approach

3.1 Scope of the work

In this work, it is intended to reproduce the method proposed by the original Netflix attack. Hence, Algorithm 1 has been implemented in Python, using the scoring metric defined by Equation 2.

However, several differences with the original paper are to be highlighted. Firstly, it used 50 records from the IMDb. However, the datasets that are currently publicly available

from the IMDb are the ratings for each movie, but not the ratings from a given user. A solution would be to get this data directly from the IMDb website because the ratings of a user are public if he also wrote a review. A data miner that parses the content of random IMDb users could do the job, but there are two obstacles:

- it would be of significant complexity, and is out of scope of the project.
- the terms and conditions of IMDb prohibit the usage of "data mining, robots, screen scraping, or similar data gathering and extraction tools"[2]. It can be suspected that it is the reason only 50 entries we used in the original attack.

As a workaround, we propose to use the MovieLens dataset as auxiliary information. MovieLens is a web-based movie recommender system that makes its database available for research [10]. This database has already been used in privacy-related research, such as [11]. As opposed to the IMDb case, the "anonymous" user IDs are consistent across all the movie ratings that are registered, which makes it suitable for the user re-identification. Also, it is the occasion to test another dataset against the Netflix one.

After the raw implementation of the matching algorithm, a verification procedure is proposed to assess its robustness. Indeed, it is needed to validate the algorithm without knowing the ground truth of the matching users.

3.2 Data pre-processing

A significant amount of pre-processing was needed on both the MovieLens and Netflix datasets before being able to run the matching algorithm.

On one hand, the Netflix dataset contains over 100 million ratings from 480,000 users (around 5.5 GB of data). One folder contains 17,770 files (one per movie) filled with three columns: `userID, rating, date`. Another file maps each file to a movie name and specifies the movie release date. On the other hand, the MovieLens database has a different structure. It contains one file filled with four columns: `userID, movieID, rating, timestamp` as well a file mapping movieIDs to titles. Figure 1 depicts how the Netflix dataset was processed to give it the same structure as the MovieLens one.

After the reshaping process, useless entries from both database were removed to make the score computation faster later on. Entries can be discarded for different reasons:

- Out-of-bounds timestamps: according to its ReadMe, the MovieLens database has records from ratings performed between January 09, 1995 and March 31, 2015. The Netflix database only ranges from October 1998 to December 2005, so that many ratings can be eliminated. The amount of distinct users from MovieLens drops from 138,493 to 52,875.
- Isolated movie: a movie rating is removed from one database if the movie is not present in the other database as it would not taken into account in the scoring

[2] see https://www.imdb.com/conditions

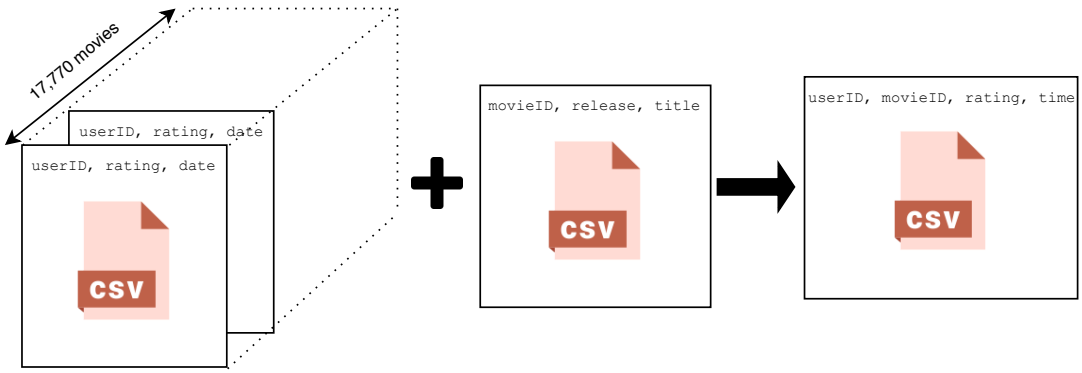

Figure 1: Data reshaping of the Netflix dataset

function. A movie was identified by its title and release date to filter only those appearing in both datasets[3]. There were around 5800 movies that were rated in both datasets (from the original 17,700 of Netflix).

Ultimately, the movie IDs needed to be made consistent between the two datasets to allow for proper implementation of the scoring function. Indeed, movies were not given the same `movieID` in both datasets.

Those lengthy data manipulations are not de-anonymization *per se* but are an unavoidable step needed to allow the implementation of the algorithm.

4 Results

4.1 Matching algorithm

The matching algorithm was run on the cleaned MovieLens dataset and on a subset of the Netflix dataset containing the ratings of FILL users. The algorithm found matches with an eccentricity higher than 1.5 for FILL of the entries, and an histogram of the results is depicted in Figure 2.

Unsurprisingly, most of the identified matches have a low ϕ regardless of being above the threshold. UPDATE amount of matches have an eccentricity $\phi < 2.5$, casting reasonable doubt about the fact that those are indeed matching users. However, FILL matches have an eccentricity of more than FILL, leading to the belief that de-anonymization was correctly carried out for those users.

4.2 Validation

In order to be able to conclude that de-anonymization is successful without the knowledge of the ground truth of matching users, a validation procedure is needed. It is proposed to

[3]Few special cases were also discarded, for example the fact that two movies named *Hamlet* were released in 2000, so that it was not possible to differentiate between them.

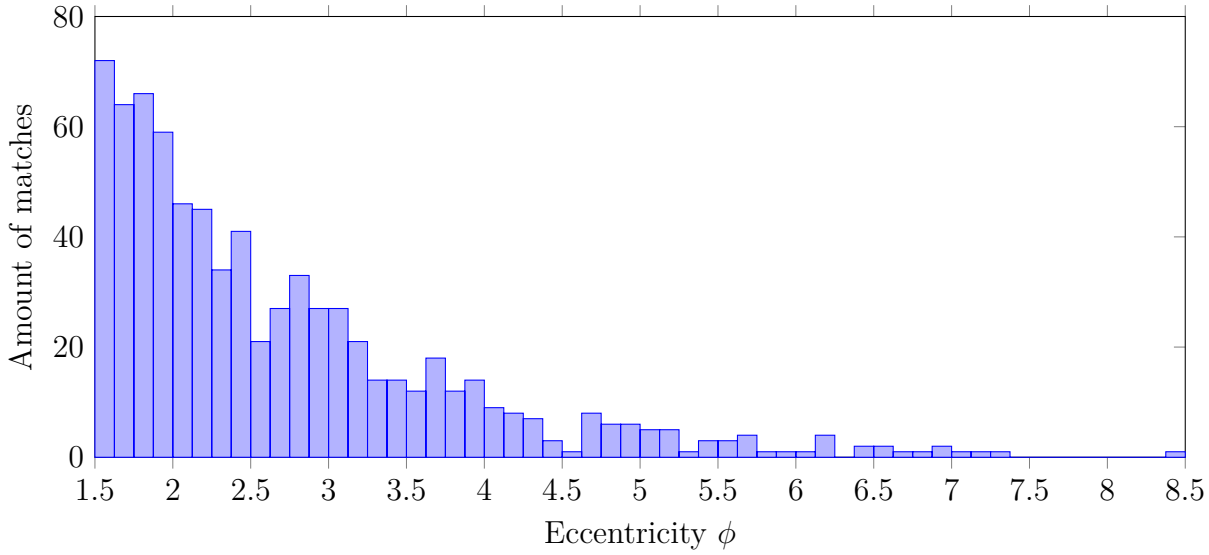

Figure 2: Histogram of the found matches and their corresponding eccentricity.

include a dummy user in both database, with the same ratings and same timestamp for 30 movies. A functioning algorithm should identify a very strong match between the two entries.

This was done for our algorithm and the movies were randomly chosen (the popularity of the involved movies has an impact on the scoring function). The results were averaged over 100 realizations and the found match had a mean eccentricity of 60.47. This confirms that entries that match perfectly yield a very high eccentricity. However, such a perfect situation is not expected. Some variability is present in the data, would it be due to user behavior or to noise voluntarily added in the datasets for anonymity purposes. Hence, the robustness to noise is studied later.

An interesting factor influencing the scoring function is the amount of movies that are common for users in both databases. In the validation step, 30 common movies were injected in the data. However, this is a purely arbitrary value. The impact of the number of movies is depicted in Figure 3.

From this analysis, it can be seen that a single rating that has exactly the same values for ρ and d is enough to exceed the threshold $\phi = 1.5$. However, it is intuitively not enough to conclude that the accounts can be linked together, confirming the hypothesis that the previously found matches with σ around 2 are not to be trusted.

4.3 Robustness analysis

Because noise is present in the data, the same dummy user is used to assess noise robustness. Instead of including exactly matching records, the ratings and timestamps will be perturbed with uniformly distributed noise. If a movie rating in one database is ρ, the

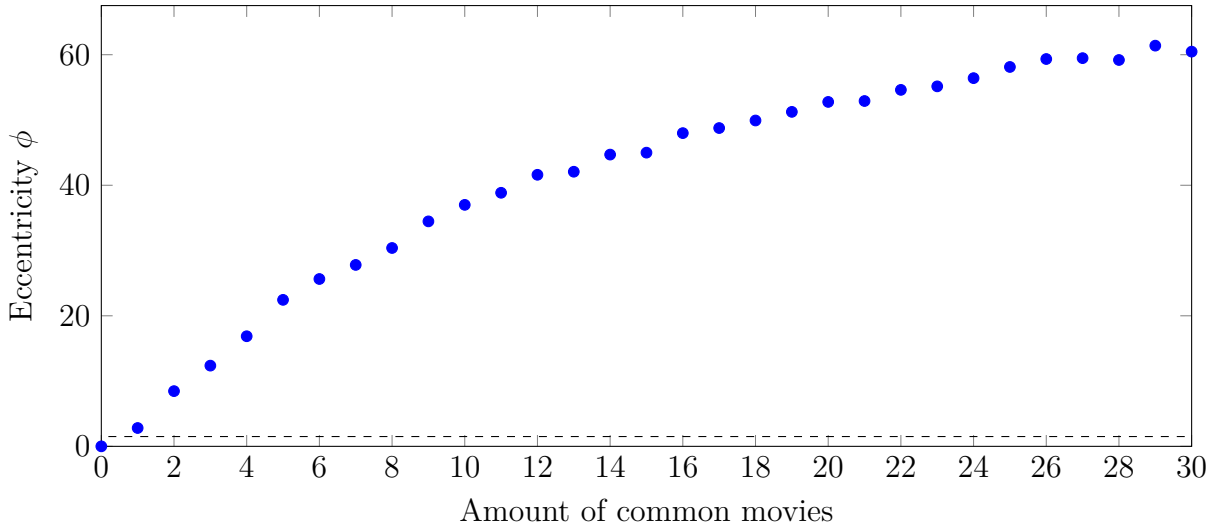

Figure 3: Influence of the amount of common movies on the eccentricity of a match. The dashed line highlights the $\phi = 1.5$ threshold.

corresponding rating in the other database is:

$$\rho_{\text{noisy}} = \rho + \mathcal{U}\left[-\sigma_\rho, \sigma_\rho\right] \tag{4}$$

Where $\mathcal{U}\left[-\sigma_\rho, \sigma_\rho\right]$ is a random variable uniformly distributed between $-\sigma_\rho$ and σ_ρ. Similarly, noise on the rating timestamp is introduced as:

$$d_{\text{noisy}} = d + \mathcal{U}\left[-\sigma_d, \sigma_d\right] \tag{5}$$

The impact of the rating spread σ_ρ on the eccentricity of the dummy user is shown in Figure 4.

Figure 4: Influence of noisy movie ratings on the eccentricity of a match.

The impact of the timestamp spread σ_d on the eccentricity of the dummy user is shown in Figure 5.

Figure 5: Influence of noisy time stamps on the eccentricity of a match.

5 Conclusion

The famous Netflix attack was reproduced on a subset of the Netflix and the MovieLens datasets. Several matches with an eccentricity higher than FILL were found, which are lower values than was found in the original attack.

To validate the method, a fake entry was added in both datasets and the corresponding eccentricity was FILL.

References

[1] The New York Times. *A Face Is Exposed for AOL Searcher No. 4417749*. [Online, last accessed 9 April 2018]. URL: https://www.nytimes.com/2006/08/09/technology/09aol.html.

[2] Dalal Al-Azizy et al. "A Literature Survey and Classifications on Data Deanonymisation". In: *Risks and Security of Internet and Systems*. Ed. by Costas Lambrinoudakis and Alban Gabillon. Cham: Springer International Publishing, 2016, pp. 36–51. ISBN: 978-3-319-31811-0.

[3] A. Narayanan and V. Shmatikov. "De-anonymizing Social Networks". In: *2009 30th IEEE Symposium on Security and Privacy*. 2009, pp. 173–187. DOI: 10.1109/SP.2009.22.

[4] A. Narayanan, E. Shi, and B. I. P. Rubinstein. "Link prediction by de-anonymization: How We Won the Kaggle Social Network Challenge". In: *The 2011 International Joint Conference on Neural Networks*. 2011, pp. 1825–1834. DOI: 10.1109/IJCNN.2011.6033446.

[5] Xuan Ding et al. *De-Anonymizing Dynamic Social Networks*. DOI: 10.1109/GLOCOM.2011.6133607.

[6] T. Okuno et al. "Content-Based De-anonymisation of Tweets". In: *2011 Seventh International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. 2011, pp. 53–56. DOI: 10.1109/IIHMSP.2011.57.

[7] Salvador Ochoa et al. *Reidentification of Individuals in Chicago's Homicide Database: A Technical and Legal Study*. 2001.

[8] A. Narayanan and V. Shmatikov. "Robust De-anonymization of Large Sparse Datasets". In: *2008 IEEE Symposium on Security and Privacy (sp 2008)*. 2008, pp. 111–125. DOI: 10.1109/SP.2008.33.

[9] Anupam Datta, Divya Sharma, and Arunesh Sinha. "Provable De-anonymization of Large Datasets with Sparse Dimensions". In: *Principles of Security and Trust*. Ed. by Pierpaolo Degano and Joshua D. Guttman. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 229–248. ISBN: 978-3-642-28641-4.

[10] F. Maxwell Harper and Joseph A. Konstan. "The MovieLens Datasets: History and Context". In: *ACM Trans. Interact. Intell. Syst.* 5.4 (Dec. 2015), 19:1–19:19. ISSN: 2160-6455. DOI: 10.1145/2827872. URL: http://doi.acm.org/10.1145/2827872.

[11] Dan Frankowski et al. "You Are What You Say: Privacy Risks of Public Mentions". In: *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '06. Seattle, Washington, USA: ACM, 2006, pp. 565–572. ISBN: 1-59593-369-7. DOI: 10.1145/1148170.1148267. URL: http://doi.acm.org/10.1145/1148170.1148267.