CSC 533: Privacy in the Digital Age (Fall 2023)
Home Assignment #1
Assigned: Friday, Sept. 01, 2023, Due: Thursday, Sept. 14, 2023

Instruction: Completed homework should be typed (e.g., using LaTeX or word document) or hand-written clearly and scanned, and uploaded into Moodle. In coding is involved provide the part of the output that is asked in each question within the submitted PDF. Other than discussion, no other form of collaboration is permitted on this assignment.

Q1. Additional files required for this homework is available inside the zipped file. In this question, we are going to work through how deanonymization works on databases. We will work with a subset of the original Netflix database. You are given a set of 15 movies in the folder titled **movies**. The identifiers for these movies are [03124, 06315, 07242, 16944, 17113, 10935, 11977, 03276, 14199, 08191, 06004, 01292, 15267, 03768, 02137]. The movies folder contains csv files for each movie. Each line of the csv file has three entries: a user id, the date of rating, and the rating provided.

Learning Objectives: The problems in this part are based on the paper 'Provable Deanonymization of Large Datasets with Sparse Dimensions" [1]. We will refer to the paper through all problems in this part. We will perform an attack along the lines of the original Netflix-IMDB deanonymization attack. In particular, we will learn how to identify a user by utilizing noisy and incomplete auxiliary information.

Starter Code: You may use the provided starter code (in Python) for this homework. The script reads each file from the movies folder and populates the database *db*. *db* is a Python dictionary. Dictionaries consist of pairs (called items) of keys and their corresponding values. Each element of *db* is the tuple *<user-id, movie-dict>*. *movie-dict* is also a dictionary representing the user's ratings for various movies, with each item being the tuple *<movie-id, rating>*. It is not compulsory that you use the starter code provided. You can use any programming language of your choice (among C/C++/Java/Python), but you must provide a README file with the proper step by step process for running your code.

Db = {user-id, {<movie-id, rating>}} → *the value of the dict is another dict*

Starter code for this problem is provided in *link.py*. You are given the auxiliary information for **one user** (*aux*). The auxiliary information contains noisy ratings given by the user for 11 of the 15 movies. You can think of these being perturbed ratings given by a user on IMDB. This auxiliary information is provided in Table 1 and in the variable aux in *link.py*. **You, as the attacker, want to identify the user id for whom the auxiliary information is provided**.

Movie	Rating	Movie	Rating	Movie	Rating	Movie	Rating
03124	3.5	10935	2	14199	3.5	103768	1.5
16944	2.5	11977	2	06004	1.5	02137	1.5
17113	4	03276	2.5	01292	2		

Table 1: Auxiliary information for the target user

(a) Using the following definition of the weigh function compute *w()* and compute the weights of each movie. Tabulate the weights obtained for each movie. This should be a table with 15 movie-ids and their corresponding weights. **[points 10]**

$w(i) = \frac{1}{\log_{10}(|supp(i)|}$, $|supp(i)|$ *is the number of times the* $i-th$ *movie is rated by all users in the whole database*

(b) Now complete the function *score()* and compute the similarity score of the auxiliary information with respect to a user's ratings in the released database (here, 'r' represents a single user in the DB). Print out the **top five** similar candidate user IDs for 'aux' (i.e., which five user IDs are most similar to 'aux'). **[points 20]**

$$score(aux, r) = \sum_{i \in supp(aux)} w(i) * \frac{T(aux(i), r(i))}{|supp(aux)|}$$

Where $|supp(aux)| = no.of\ non\ null\ attributes\ (i.e., movie\ ratings)\ in\ aux$, i.e., only consider the movies rated by 'aux' when computing the similarity score.

[1]. http://www.andrew.cmu.edu/user/divyasha/dss-post12.pdf

CSC 533: Privacy in the Digital Age (Fall 2023)
Home Assignment #1
Assigned: Friday, Sept. 01, 2023, Due: Thursday, Sept. 14, 2023

$$T\big(aux(i), r(i)\big) = 1 - \frac{|aux(i) - r(i)|}{p(i)},$$

Where *aux(i)* and *r(i)* is the rating of movie '*i'* and *p(i)* is the maximum possible difference in rating for movie '*i'* in the whole database (i.e., *max* **difference is rating** possible for a given movie even considering values from *aux).* Here, the difference in movie rating between *aux* and *r* is scaled (normalized) by *p(i),* so that the value for T lies in the interval [0, 1].

(c) Compute the followings:

- What is the user-id of the user with the highest score with *aux*? **[points 5]**
- Write out the movie ratings of this user from the database, side-by-side with the ratings from the auxiliary data. Comment on how similar/dissimilar they are. **[points 5]**

(d) Compute the followings:

- What is the difference between the highest and second highest similarity score? [**points 5**]
- Assume the algorithm accepts the top candidate if the difference between the highest and second highest similarity score > γ*M, where $M = \sum_{i \in supp(aux)} \frac{w(i)}{|supp(aux)|}$ is the scaled sum of weights of attributes in *aux* and γ is some constant.
  a) if γ = 0.1 would you accept the top candidate? [**points 2.5**]
  b) If γ = 0.05 would you accept the top candidate? [**points 2.5**]

Q2. For the following questions consider table below-

ID	ZIP code	Age	Salary	Disease
1	47677	29	3K	gastric ulcer
2	47602	22	4K	gastritis
3	47678	27	5K	stomach cancer
4	47905	43	6K	gastritis
5	47605	30	7K	pneumonia
6	47906	47	8K	bronchitis
7	47674	36	9K	bronchitis
8	47607	32	10K	pneumonia
9	47909	42	11K	stomach cancer

(a) What are the quasi-identifiers and sensitive attributes in this table? **[points 5]**

(b) Your task is to compose a **3-anonymous, 3-diverse** table.

- Define the generalization hierarchy for each of your quasi-identifiers. **[points 5]**
- Next, using **incognito algorithm** draw the generalization lattice (remember to label or color each node as to whether they satisfy 3-anonymity, 3 diverse or not). **[points 10]**
- Identify the node in the lattice that you are using for your final solution and write out the final anonymized table of your choice. **[points 10]**

(c) Compute the followings:

- Compute the **t-closeness** of your selected solution (from part **b**) with respect to **only** 'salary'. **[points 20]**

Submission:

You have to submit **three** files:
1. Merge all the written answers into a single pdf file named <your unity id>_HW1.pdf. **Even if you print out the answers in your code, please copy them in the PDF. We want all the answers in one place.**
2. Rename the program file you used as <your unity id>_HW1_Qx.extension (e.g., .c/.cpp/.java/.py).
3. Add a README file regarding how to run your code. Remember to describe any dependencies.

Don't submit ipython notebooks. We want to run scripts. Zip all files into <your unity id>_HW1.zip and submit the zip file on Moodle.

[1]. http://www.andrew.cmu.edu/user/divyasha/dss-post12.pdf