COMP 430/530: Data Privacy and Security – Fall 2023

Homework Assignment #1

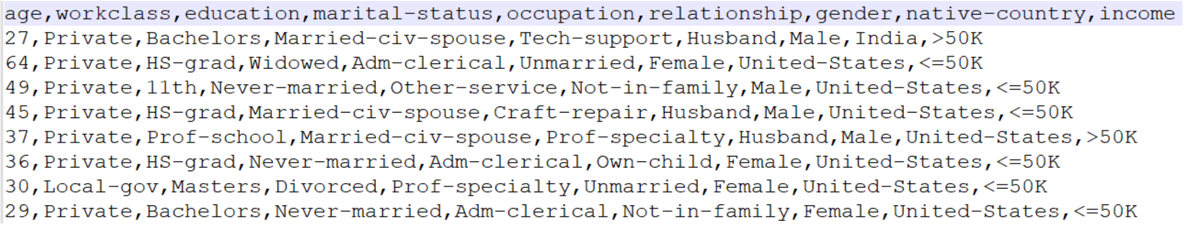
INTRODUCTION AND SETUP

In this assignment, you will implement k-anonymization algorithms in Python and compare their performance by executing your algorithms on a real dataset. You should use the skeleton Python file that we are providing as your starting point. (Submit **.py** only, we do not accept **.ipynb** extension.)

We also provided you a simplified and shortened version of the **Adult** dataset, which is a commonly used dataset in the literature (original version is here: http://archive.ics.uci.edu/ml/datasets/Adult, but you do not need the original version, you will be working with the version we provided).

Each row of **Adult** corresponds to one individual. Each column (attribute) contains information such as the individual's gender, education level, marital status, etc. The final column (***income***) is a binary attribute that tells whether the individual's yearly income is ≤ 50K or > 50K dollars. Throughout this assignment, you will treat every attribute other than ***income*** as a Quasi Identifier; and treat ***income*** as the Sensitive Attribute.

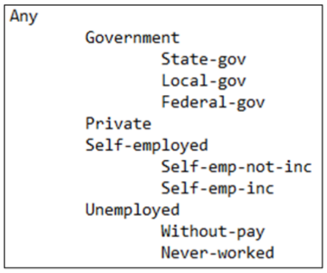
Here is a screenshot from the **Adult** dataset that is provided to you. Notice that the dataset is in **csv** (comma separated values) format and the first row contains attribute names.



```
age,workclass,education,marital-status,occupation,relationship,gender,native-country,income
27,Private,Bachelors,Married-civ-spouse,Tech-support,Husband,Male,India,>50K
64,Private,HS-grad,Widowed,Adm-clerical,Unmarried,Female,United-States,<=50K
49,Private,11th,Never-married,Other-service,Not-in-family,Male,United-States,<=50K
45,Private,HS-grad,Married-civ-spouse,Craft-repair,Husband,Male,United-States,<=50K
37,Private,Prof-school,Married-civ-spouse,Prof-specialty,Husband,Male,United-States,>50K
36,Private,HS-grad,Never-married,Adm-clerical,Own-child,Female,United-States,<=50K
30,Local-gov,Masters,Divorced,Prof-specialty,Unmarried,Female,United-States,<=50K
29,Private,Bachelors,Never-married,Adm-clerical,Not-in-family,Female,United-States,<=50K
```

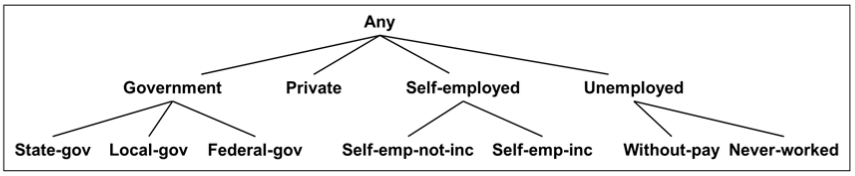
In addition to the **Adult** dataset, you are also given a folder named **DGHs**. Each file in the DGHs folder contains the domain generalization hierarchy of one of the QI attributes, e.g., **age.txt** contains the DGH for the **age** attribute, **education.txt** contains the DGH for the **education** attribute, and so forth.

When you study the contents of the DGH files, you will see that they follow a pattern such that the tabs indicate parent-child relationships in a DGH. For example, here is how **workclass.txt** represents the visual DGH of the **workclass** attribute:



```
Any
	Government
		State-gov
		Local-gov
		Federal-gov
	Private
	Self-employed
		Self-emp-not-inc
		Self-emp-inc
	Unemployed
		Without-pay
		Never-worked
```

DGH in workclass.txt



Visual representation of DGH for workclass

Your first goal should be to read datasets and DGHs into your program's memory. It is important to note that the **Adult** dataset and its DGHs are provided to you as samples, to help you in development and testing. Your code should not be specific to these inputs – **it should work for any dataset and any set of DGHs!** Hence, do not hardcode any paths, filenames, DGHs, etc. We may test your code with arbitrary datasets and DGHs, and your code should still work.

You can assume that the following conditions will always hold:

- The dataset will always be a csv file and its first row will contain the names of attributes.
- For a dataset containing N columns, 1 of those columns will be the SA and the rest will be QIs.
- All DGHs will be given to you in one folder containing multiple files (one file for each DGH).
- All necessary DGHs will be present and all DGHs will be complete (i.e., no need to check if there is a missing DGH file or if the DGH covers all possible values for an attribute – it does).

The homework assignment consists of several parts. In each part, we give you the names and parameters of the functions you need to implement. **Do not modify function names or parameters (e.g., do not add or remove parameters)!** When grading, we will call exactly these functions with different inputs – if you modify function names or parameters, we won't be able to call your functions the way we are expecting to, and you will lose points.

In addition to the functions you are asked to implement, you are welcome to implement as many additional or helper functions as you like. Try to make your code modular, organized and easy-to-follow. This may help us in giving partial credit.

As you go through the assignment, you will find out that there are multiple ways to implement a certain operation. We strongly recommend that you take into account **efficiency** (both time-efficiency and memory-efficiency). An inefficient implementation can take several hours on a small dataset, whereas an efficient implementation can finish in a few minutes.

GOOD LUCK!

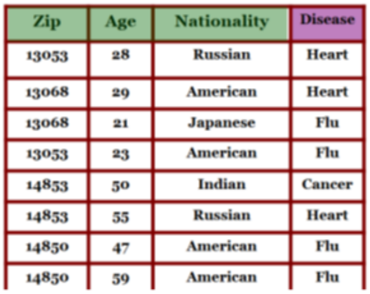
PART 1: ANONYMIZATION COSTS (20 pts)

In the lectures, we covered two metrics to measure costs of anonymization: Distortion Metric (MD) and Loss Metric (LM). In this part, you will implement these metrics.

cost_MD (*raw_dataset_file*, *anonymized_dataset_file*, *DGH_folder*)

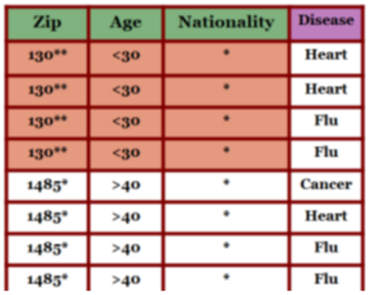
- *raw_dataset_file* is a string containing the path of the raw dataset file, e.g., *"adult-hw1.csv".*
- *anonymized_dataset_file* is the path of the anonymized dataset, e.g., *"adult-anonymized.csv".*
- *DGH_folder* is the directory containing all DGH files.

This function should read the raw dataset and anonymized dataset from the corresponding files and calculate the cost of anonymization using the Distortion Metric. You can assume that the order of the rows and columns are the same in the raw and anonymized datasets. That is: (i) both datasets contain the same attributes in the same order, (ii) N'th row in the anonymized dataset is the generalized version of the N'th row in the raw dataset. For example:



Zip	Age	Nationality	Disease
13053	28	Russian	Heart
13068	29	American	Heart
13068	21	Japanese	Flu
13053	23	American	Flu
14853	50	Indian	Cancer
14853	55	Russian	Heart
14850	47	American	Flu
14850	59	American	Flu

Raw dataset



Zip	Age	Nationality	Disease
130**	<30	*	Heart
130**	<30	*	Heart
130**	<30	*	Flu
130**	<30	*	Flu
1485*	>40	*	Cancer
1485*	>40	*	Heart
1485*	>40	*	Flu
1485*	>40	*	Flu

Anonymized dataset

In this example, it also happens to be that consecutive records belong to the same equivalence class. This need not always be the case. For example, in the files given to you, rows 1-3-8 could constitute one equivalence class, rows 2-4-5 could constitute another equivalence class, etc.

The return value of the **cost_MD** function should be the MD cost.

cost_LM(*raw_dataset_file*, *anonymized_dataset_file*, *DGH_folder*):

This function follows the same parameters and assumptions as **cost_MD**. But instead of calculating MD cost, it calculates and returns the Loss Metric (LM) cost. When calculating LM cost, assume that the weights of all attributes in the dataset are identical, i.e., for a dataset with M quasi-identifier attributes, $w_1 = w_2 = \ldots = w_M = 1/M$.

PART 2: RANDOM ANONYMIZER (15 pts)

In this part, you will implement a randomized algorithm for k-anonymizing a dataset. The algorithm works as follows:

1. Given dataset ***D*** and k-anonymity parameter ***k***, the algorithm randomly shuffles the dataset to randomize the order of the rows.
2. The first k records are put into EC1, the next k records are put into EC2, the next k records are put into EC3, … and so forth.
    a. If the size of the dataset is a perfect multiple of ***k***, you'll have exactly **|*D*|/*k*** ECs.
    b. Otherwise, the very last EC will contain between ***k+1*** and ***2k-1*** records, so that all records can end up belonging to an EC that contains at least ***k*** records.
3. For each EC, make sure that QI-wise equality is achieved through generalizations. While doing so, perform the minimum amount of generalization necessary to achieve k-anonymity for that EC. **No redundant generalizations or over-generalizations!**
4. Finally, the impact of the shuffling performed in step #1 is reversed, so that the N'th row in the anonymized dataset will be the generalized version of the N'th row in the raw dataset when we write the anonymized dataset to an output file.

In the skeleton code provided to you, steps 1 and 4 are already taken care of. Do not modify these parts. We did this so that when we call your function with a fixed seed (for the random shuffler), we'll get the same results each time. This makes grading and debugging easier on our end. If you modify these parts, your output may not match what we are expecting, and you may lose points.

Your task is to add the code necessary to perform steps 2 and 3.

random_anonymizer (*raw_dataset_file*, *DGH_folder, k, output_file, s*):

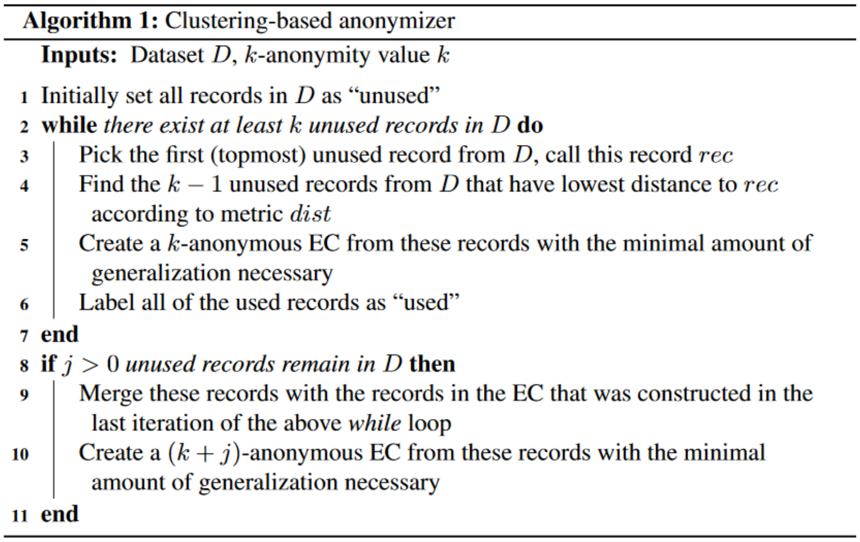
- *raw_dataset_file* is a string containing the path of the raw dataset file, e.g., *"adult-hw1.csv".*
- *DGH_folder* is the directory containing all DGH files.
- *k* is the k-anonymity parameter.
- *s* is the seed for the random shuffler. You can call your function with different seeds to observe different results.
- The function has no return value, but it should write the anonymized dataset to a file. The name of this file is passed in a parameter called *output_file*, e.g., *output_file* = *"anon.csv"*. N'th row in the anonymized dataset will be the generalized version of the N'th row in the raw dataset.

PART 3: CLUSTERING-BASED ANONYMIZER (20 pts)

Now, let us implement a more intelligent k-anonymization algorithm based on the idea of clustering similar records together. Recall from the lecture that we can define a distance metric **dist(r1,r2)** between two records r1 and r2 as the **total LM cost of hypothetically placing those two records in one equivalence class (EC) with the minimum amount of generalization necessary**. When calculating the LM cost, assume that the weights of all attributes in a dataset are identical, i.e., for a dataset with M quasi-identifier attributes, $w_1 = w_2 = \ldots = w_M = 1/M$.

[Hint: You may want to re-use some code that you wrote for the cost_LM function in Part 1.]

Given dataset ***D*** and parameter ***k***, your clustering-based anonymizer should work as follows:



```
Algorithm 1: Clustering-based anonymizer
   Inputs: Dataset $D$, $k$-anonymity value $k$
1  Initially set all records in $D$ as "unused"
2  while there exist at least $k$ unused records in $D$ do
3      Pick the first (topmost) unused record from $D$, call this record $rec$
4      Find the $k-1$ unused records from $D$ that have lowest distance to $rec$
         according to metric $dist$
5      Create a $k$-anonymous EC from these records with the minimal amount of
         generalization necessary
6      Label all of the used records as "used"
7  end
8  if $j > 0$ unused records remain in $D$ then
9      Merge these records with the records in the EC that was constructed in the
         last iteration of the above while loop
10     Create a $(k+j)$-anonymous EC from these records with the minimal
         amount of generalization necessary
11 end
```

Some notes and clarifications:

- The terminology "used" and "unused" records indicate whether that record was previously used in the construction of an EC. If a record has previously been used to construct one EC, we label it as "used" so that it is not re-used in a later iteration in another EC.
- On line 3, "first (topmost) unused record from D" means the unused record with lowest index in the dataset D.
- The purpose of lines 8-11 is to handle datasets with cardinality other than perfect multiple of **k**.
- On line 4, you may need to perform tie-breaking when there are multiple records with the exact same distance to the current record. In such cases, always prefer the topmost records in the dataset (i.e., records which have the lowest index). Notice that the clustering-based anonymizer becomes deterministic using this tie-breaking strategy.

The signature of the function you need to implement is:

clustering_anonymizer *(raw_dataset_file, DGH_folder, k, output_file)*

The parameters have the same meaning as in **random_anonymizer**. Similar to random_anonymizer, clustering_anonymizer does not have a return value. Instead, it should write the anonymized dataset to *output_file*. N'th row in the anonymized dataset should be the generalized version of the N'th row in the raw dataset.

PART 4: TOP-DOWN ANONYMIZER (30 pts)

Recall the “top-down” anonymization approach from our lectures. In the top-down approach, all records in the dataset are initially assumed to be maximally generalized at the root node of the specialization tree. Then, each node of the tree is recursively split into its children via specializations.

The split criteria, i.e., which specialization to perform at each node, is a key factor. If no specializations can be performed at a certain node because all specializations lead to violations of k-anonymity or there is no possible specialization left to perform according to DGHs, then that node stops splitting.

In this part, you will implement the top-down anonymization approach with the following split criteria.

Say that you are considering splitting node **n** of the specialization tree. For **n**, among the specializations that are possible according to DGHs and do not lead to a violation of k-anonymity, you should pick the specialization that minimizes the number of child nodes. If there are multiple options to choose from (i.e., min number of child nodes is 3 and there are multiple possible specializations that lead to 3 child nodes), you should pick the specialization that makes the distribution of resulting records in child nodes closest to the uniform distribution. For instance:

- Let’s say according to DGHs and without violating k-anonymity, the minimum possible number of child nodes is found to be **3**.
- The uniform distribution with 3 child nodes is: **[0.33, 0.33, 0.33]**.
- Assume that there are two possible specializations that can be performed which yield 3 child nodes. Specialization **s1** yields a distribution of **[0.30, 0.36, 0.34]** and **s2** yields **[0.10, 0.60, 0.30]**. To calculate the distance between two distributions we use the **L1** norm.
    - The L1 distance between the distribution resulting from **s1** and uniform distribution is: **|0.33 - 0.30| + |0.33 - 0.36| + |0.33 - 0.34| = 0.07**.
    - The **L1** distance between the distribution resulting from **s2** and uniform distribution is: **|0.10-0.33| + |0.60-0.33| + |0.30-0.33| = 0.53**.
    - Therefore, **s1** specialization is chosen rather than **s2**.

Important: Each node in the specialization tree makes its own decision regarding which specialization to perform. Nodes in the same tree depth do not have to apply the same specialization.

topdown_anonymizer (*raw_dataset_file*, *DGH_folder, k, output_file*):

- Implement your top-down anonymizer under this function.
- The parameters of this function are identical to those functions in Parts 2 and 3.
- Again, preserve the order of the rows when writing the anonymized dataset to *output_file*.

PART 5: MINI REPORT (15 pts)

Submit a **max** one-page report (hard limit!) containing your experiments and analysis of the different anonymization algorithms you implemented in Parts 2, 3 and 4. Your report should be in pdf format.

First, run the 3 anonymizers on the Adult dataset for different values of k: k = 4, 8, 16, 32, 64, 128, 256. In each run, measure the anonymizer's ***time cost*** (how long does it take to execute), ***LM cost***, and ***MD cost***. When applicable, we recommend that you repeat each experiment a few times and average the results in order to improve statistical significance.

- In Part 2, you can change the seed value to achieve randomization.
- In Part 3, you can shuffle the data (do it before calling clustering_anonymizer) to achieve randomization.

Next, report your experiment results in tables and/or graphs. You should choose which tables or graphs are most suitable for your report. (Hint: **LM cost vs k**, **time vs k, MD cost vs k**).

Finally, briefly discuss your observations and take-away messages. For example, what trade-offs exist? Which anonymizer is fastest? Which one has the lowest utility loss? Which anonymizer would you prefer under what settings? Do the results fit your expectations? What did you learn from this assignment?

When grading your report, we will pay attention to how you constructed your tables/graphs, their readability and quality, as well as the quality of your analysis and discussion.

SUBMISSION

When you are finished, submit your assignment via Blackboard:

- Move all of your Python files and your report into a folder named **`your KU ID`.**
- **Compress this folder into a single zip file**. (Don't use compression methods other than zip.)
- Upload your zip file to Blackboard.

Notes and reminders:

- After submitting, download your submission and double-check that: (i) your files are not corrupted, (ii) your submission contains all the files you intended to submit, including all of your source code and your report. If we cannot run your code because some of your code files are missing, we cannot give you points!
- You must upload your code in py files**, we do not accept Python notebooks (.ipynb extension)**.
- This homework is an **individual assignment**. All work needs to be your own. Submissions will be checked for plagiarism (including comparing to previous years' assignments).
- **Your report should be a pdf file.** Do not submit Word files or others which may only be opened on Windows or Mac (or opening them may remove table/figure formatting).
- Only Blackboard submissions are allowed. Do not e-mail your assignment to the instructor or TAs.
- **Do not submit any data files** (such as the Adult dataset, DGHs, or your anonymized datasets).
- Do not change the names or parameters of the functions that we will grade.
- If your code does not run (e.g., syntax errors) or takes an extremely long amount of time (e.g., it takes multiple hours whereas it should take 2-3 minutes), you may get 0 for the corresponding part.