Graduate Algorithms Project: Privacy-Preserving Data Sampling

1. Background

In many real-world scenarios, a data owner may wish to share data with data analysts for research or model training. However, user consent often restricts access to only part of the dataset, typically, the non-sensitive attributes. This project explores how to generate a representative sample from the non-sensitive subset that both preserves the statistical characteristics of the full dataset and enables the training of machine learning models whose performance closely approximates those trained on the original data. The challenge lies in balancing privacy protection and data utility.

2. Objectives

Your group will design and implement an algorithm to select a representative sample from non-sensitive data, minimizing information leakage while maintaining model quality. By the end of this project, you should be able to:

- Apply algorithmic thinking to data privacy challenges.
- Use mutual information or other statistical metrics to quantify data similarity.
- Evaluate algorithm efficiency and effectiveness both theoretically and empirically.

3. Project Tasks

Step 1: Dataset Selection

• Choose a publicly available dataset from Kaggle or the UCI Machine Learning Repository.

• Identify a subset of columns as sensitive attributes (e.g., in a customer dataset, 'education-level' or 'salary' might be sensitive).

• Ensure the dataset is suitable for a classification task.

Step 2: Sampling Algorithm Design

Let the original dataset D contain n records, with both sensitive and non-sensitive columns. Design an algorithm to select m records from only the non-sensitive columns, forming a representative sample S. The goal is to minimize the mutual information (MI) between S and D, subject to the sample size constraint m. You are encouraged to research the definition and computation of mutual information, explore heuristic, search, or sorting algorithms discussed in this course, and describe your algorithm's time complexity.

Step 3: Implementation

Implement your algorithm in Python. Train a classification model (e.g., decision tree, logistic regression, SVM, etc.) on your selected sample S. Compare its performance to a model trained on the original dataset D.

4. Evaluation and Future Work

In the final phase (after November 20), we will:

• Examine whether mutual information can be replaced by alternative similarity or distance metrics.

• Evaluate your algorithm based on computational efficiency (linear, sub-linear, or

polynomial time) and model performance (how closely the classifier trained on S approximates the one trained on D).

5. Timeline

Phase	Dates	Description
Phase 1	Nov 3 – Nov 7	Literature review, dataset selection, define sensitive attributes
Phase 2	Nov 7 – Nov 20	Algorithm design and Python implementation
Phase 3	After Nov 20	Feedback, analysis, and alternative metric exploration

6. Deliverables

Each group must submit the following:

1. Code files implementing your algorithm and model training.
2. A technical report (3–5 pages) including:
    - Dataset description and justification of chosen sensitive attributes
    - Explanation of your algorithm (with pseudocode and complexity analysis)
    - Experimental results and evaluation
    - Discussion of limitations and possible improvements

7. Notes

• All groups are assigned the same question but are expected to propose unique approaches.

• Collaboration within your group only is permitted.

• Use of generative AI tools must follow the academic integrity policies of your institution.