**Progetto Data Protection & Privacy Francesco Calcagno: 4269014 Alessandro Vernazza: 4229315**

**Algoritmo CAHD per l’anonimizzazione di dati transazionali.**

Il nostro progetto presenta due implementazioni, una standard ed una da noi modificata per apportare e testare migliorie.

L’implementazione **standard** consiste dei seguenti file :

-Main.py : compie l’inizializzazione del programma, durante l’esecuzione richiama le classi BandMatrix e CAHDalgorithm ed esegue l’anonimizzazione del dataset specificato secondo i parametri di default. Al termine dell’esecuzione viene richiamata la classe OutputData per la stampa a video dei gruppi anonimizzati e per effettuare controlli su di essi.

-BandMatrix.py : riceve un .csv in ingresso e ne crea la matrice bandizzata necessaria per CAHDalgorithm.

-CAHDalgorithm.py: è il core del programma, riceve in ingresso la matrice bandizzata e ne crea i gruppi anonimizzati a seconda dei parametri forniti.

-Main\_KL\_Divergence.py : anche questo file svolge il calcolo della matrice bandizzata e ne fa l’anonimizzazione sempre con l’algoritmo CAHD andando successivamente a calcolare la KL-Divergence, richiamando KLDivergence, che rappresenta l’indice di ricostruibilità del Dataset anonimizzato rispetto a quello originale.

-KLDivergence.py: classe per il calcolo della KL-Divergence.

-OutputData.py: classe di supporto per la stampa a video dei gruppi e per effettuare un controllo sul fatto che l’anonimizzazione eseguita sia effettivamente corretta controllando la coerenza con i dati iniziali.

I file descritti a seguire sono invece relativi alla fase di testing, generazione e plotting dei dati:

-Main\_Privacy-KLD\_Analysis.py: programma che applica l’anonimizzazione e per ogni risultato ne calcola la KL-Divergence, genera un file utilizzato successivamente per il plotting nel quale sull’ascissa verranno rappresentati i diversi gradi di privacy tentati e sull’ordinata invece la KL-Divergence. Il tutto viene salvato in MainPlotData\0\.

-Main\_Sensitive-KLD\_Analysis.py: programma che applica l’anonimizzazione e per ogni risultato ne calcola la KL-Divergence, genera un file utilizzato successivamente per il plotting nel quale sull’ascissa verrà rappresentato il numero di righe contenenti dati sensibili e sull’ordinata invece la KL-Divergence. Il tutto viene salvato in MainPlotData\1\.

-Main\_R-KLD\_Analysis.py: programma che applica l’anonimizzazione e per ogni risultato ne calcola la KL-Divergence, genera un file utilizzato successivamente per il plotting nel quale sull’ascissa verrà rappresentato il numero di quasi-identifier su cui calcolare la divergenza e sull’ordinata invece la KL-Divergence stessa. Il tutto viene salvato in MainPlotData\2\.

-Main\_Privacy-Time\_Analysis.py: programma che applica l’anonimizzazione e per ogni risultato ne calcola la KL-Divergence, genera un file utilizzato successivamente per il plotting nel quale sull’ascissa verranno rappresentati i diversi gradi di privacy tentati e sull’ordinata invece il tempo necessario per l’anonimizzazione per ogni grado di privacy. Il tutto viene salvato in MainPlotData\3\.

-Main\_Plot\_Analysis.py: legge da MainPlotData\ i file contenenti i dati generati dai programmi precedentemente illustrati e genera un grafico per ognuno di essi.

*Utilizzo:*

*Per l’anonimizzazione standard utilizzare Main.py, se invece si volesse fare il plot dei dati generati dai file precedentemente descritti utilizzare Main\_Plot\_Analysis.py*

L’implementazione **alternativa** aggiunge un parametro nella scelta delle transazioni nella fase di creazione dei gruppi andando a dare priorità alle transazioni contenenti dati sensibili tramite il parametro **priority** per evitare la formazione di un ultimo gruppo molto più grande degli altri e quindi con transazioni molto dissimili fra loro.

-Main\_Priority.py: analizza dataset di diversa densità dalla cartella Dataset-Test-Priority con diversi valori di priority, calcola la KL-Divergence e stampa su file i risultati nelle cartelle TestDensi e TestSparsi. Utilizza BandMatrix\_Priority e CAHD\_Priority e KLDivergence(stesso file dell’implementazione standard).

-**BandMatrix\_Priority** e **CAHD\_Priority** sono solo implementazioni semplificate dei file sopra descritti.

-MainResults.py: analizza un singolo dataset con svariati valori di privacy e priority creando dei file nei quali vengono stampati i seguenti valori per ogni combinazione di privacy e priority per un numero prefissato di volte : KL-Divergence, il tempo impiegato per l’esecuzione, il numero di gruppi generati ed infine la coppia privacy-priority.

-MainPlotPriority: apre i file generati da MainResults e realizza i seguenti plot in funzione della suddetta coppia privacy-priority:

* Il valore della KL-Divergence prima tenendo conto dei valori più estremi e poi escludendoli
* Il tempo di esecuzione prima tenendo conto dei valori più estremi e poi escludendoli
* Il numero di gruppi formati per ogni grado di privacy. Quindi per ogni matrice bandizzata calcolata applichiamo quattro diversi valori di priority selezionati randomicamente in un intervallo [0,2 - 3] effettuando sempre un confronto con il caso a priority nulla. I colori di ogni campione nel grafico rappresentano la differenza tra il minor numero di gruppi generati da un valore di priority (i minimi vengono rappresentati con delle “x”) e il numero di gruppi generati per il dato campione. Ogni riga del grafico rappresenta un campione.