K-Anonimato

Presentazione elaborato k-anonymity - Molinari Lorenzo

Introduzione

MacroData

- Tabelle di conteggio
- Tabelle di grandezza

modalità pubblicazione:

1. Database a fini statistici
2. Dati a fini statistici

MicroData

Insieme di dati con informazioni specifiche di un certo gruppo di utenti

- maggiore flessibilità rispetto a dati statistici pre-computati
- problema legato privacy utenti e protezione delle informazioni

Attributi

Identificatori	Quasi-identifier	Confidenziali	Non Confidenziali
Identificano maniera univoca soggetto	insieme attributi che associati ad altri attributi permettono identificazione soggetti	attributi che contengono informazioni sensibili dei soggetti	attributi che non contengono informazioni sensibili o utili all'identificazione del soggetto

Problematiche di divulgazione

- **Identity Disclosure**
- **Attribute Disclosure**
- **Inferential Disclosure (importante per il k-anonimato e linking attack)**

K-anonimato

Problema dell’anonimato

Tecniche di mascheramento:

- non perturbative
- perturbative

Rilascio dati sintetici:

- completamente sintetici
- parzialmente sintetici

Per il K-anonimato vengono utilizzate tecniche non perturbative:

- Generalizzazione
- Soppressione

K-anonimato

Definizione 1: *“Requisito di k-anonimato per una release di dati deve essere tale che ogni combinazione di valori dei quasi-identifier (QI) può essere indistinguibile corrisposta ad almeno k altre identità.”*

Definizione 2*: “Data una tabella T(A1,...,An) e un attributo QI associato. T soddisfa la k-anonimità rispetto a QI se e solo se ogni sequenza di valori T[QI] appare almeno con k occorrenze in T[QI].”*

- K-anonimato
- L-diversità
- T-chiusura

Generalizzazione

Definizione 1: *“Date due tabelle Tx e Ty definite sullo stesso insieme degli attributi, Ty è una generalizzazione (con eventuale soppressione) della tabella Tx se:*

- *|Ty| <= |Tx|*
- *per ciascun attributo di Ty si ha un dominio uguale al dominio di partenza o una sua generalizzazione*
- *possibile definire funzione “f” iniettiva che associa ad elementi distinti del dominio elementi distinti del codominio che possono essere uguali o una generalizzazione degli attributi della tabella di partenza”*

- Generalizzazione sui domini
- Generalizzazione sui valori

Generalizzazione ottimale

Definizione K-minimal: “*Date due tabelle Ty e Tx tali per cui Ty <= Tx, viene definito Massimo Superiore (MaxSup) come soglia di soppressione accettabile tale per cui Ty è generalizzazione k-minimal della tabella Tx se:*

- *Ty soddisfa k-anonymity con soppressione minima*
- *|Tx|-|Ty| <= MaxSup*
- *non esiste un'ulteriore tabella che soddisfa le prime due condizioni con minore generalizzazione di Ty”*

Ne derivano le Generalizzazioni ottimale:

- distanza minima assoluta
- distanza minima relativa
- massima distribuzione
- minima soppressione

Generalizzazione

basata sulla gerarchia

Definizione di relazione di generalizzazione come <=_D che implica l'esistenza di una relazione gerarchica ordinata per ogni dominio di generalizzazione

La utilizzano:

- Samarati
- Incognito

basata sui record

Generalizzazione basata sulla generalizzazione del valore dei quasi-identificatori in intervalli, solitamente utilizzata run-time.

Richiede ordinamento dei valori degli attributi QI

La utilizza:

- Mondrian

Possibili attacchi

- Linking Attack
- Homogeneity Attack
- Background/External Attack
- Skewness Attack
- Similarity Attack
- Comparazioni su rilasci multipli

Algoritmi per il K-Anonimato

Tecniche per il K-anonimato

Generalizzazione	Soppressione			
	Tuple	Attributi	Celle	None
Attributi	AG_TS	AG_AS	AG_CS	AG_
Celle	CG_TS	CG_AS	CG_CS	CG_
None	_TS	_AS	_CS	-

Table 1: tecniche di k-anonimato[1]

Le più del rinforzo della generalizzazione più usate sono:

- AG_TS
- AG_ (AG_, AG_AS, _AS)
- CG_ (CG_, CG_CS)
- AG_CS (AG_CS, _CS)
- _TS

Algoritmi

AG e AG_TS

- Algoritmo di ricerca binaria K-minimal (Samarati)
- Algoritmo k-optimize
- Incognito
- Euristici

CG_CS e CG_

- Mondrian
- Approssimazione

Samarati - AG_ e AG_TS / basata sulle gerarchie

Algoritmo per l'implementazione del K-anonimato in una base di dati basato sulla definizione di K-Minimal.

Creazione di un albero con soluzioni della generalizzazione dove ogni percorso rappresenta una generalizzazione della tabella originale. Il nodo più basso di ogni percorso (che soddisfa il k-anonimato) è definito come generalizzazione minima locale (del ramo).

Algoritmo si basa sulla valutazione delle soluzioni ad altezze specifiche della gerarchia e continua fino a quando raggiunta altezza più bassa per la quale esiste un vettore di distanza (lattice) che soddisfa i vincoli del k-anonimato

Questo permette trovare una possibile soluzione, tra quelle locali (non garantisce quella ottimale)

Mondrian - _CS e CG_ / basata sui record

Basato sul concetto di multidimensionalità degli oggetti (record) salvati nelle tabelle, ogni QI rappresenta una dimensione e ogni tupla rappresenta un punto nello spazio (spazio definito dai QI)

Quando nell'analisi vengono trovate tuple con lo stesso valore per i QI vengono rappresentate come punti nello spazio e vengono associati valore che indica il numero di occorrenze che ne sono state trovate.

Dopo aver posto tutte le tuple nello spazio, viene suddiviso in modo tale che ogni area contenga almeno "k" occorrenze e tutti i punti che si trovano nello stesso cluster vengono generalizzati ad un valore univoco (loro generalizzazione).

Analisi applicativa

Dataset

Adult

Set di dati di UCI Machine Learning Repository, dataset benchmark più utilizzati per testare anonimizzazione in una base di dati.

QI: age, race, maritial_status, gender

Sensibile: occupation

Bank

Set di dati UCI Machine Learning Repository, per test su basi di dati più contenute.

QI: age, job

Sensibile: balance

Algoritmi

Per il test sono stati implementati due degli algoritmi più comunemente utilizzati: Samarati e Mondrian e per poter ottenere i risultati ottimali sono stati definiti i parametri di esecuzione per ciascun dataset:

- `python main.py --samarati --k 10 --maxsup 20`
- `python main.py --mondrian --k 10`
- `python main.py --samarati --k 3 --bank`
- `python main.py --mondrian --k 3 --bank`

Python main.py --samarati --k 10 --maxsup 20

```
configuration:
 {'k': 10, 'maxsup': 20, 'samarati': True, 'mondrian':
 ↪  False, 'bank': False, 'data': {'path':
 ↪  'data/adult.data', 'samarati_quasi_id': ['age',
 ↪  'gender', 'race', 'marital_status'],
 ↪  'mondrian_quasi_id': ['age', 'education_num'],
 ↪  'sensitive': 'occupation', 'columns': ['age',
 ↪  'work_class', 'final_weight', 'education',
 ↪  'education_num', 'marital_status', 'occupation',
 ↪  'relationship', 'race', 'gender', 'capital_gain',
 ↪  'capital_loss', 'hours_per_week', 'native_country',
 ↪  'class'], 'samarati_generalization_type': {'age':
 ↪  'range', 'gender': 'categorical', 'race':
 ↪  'categorical', 'marital_status': 'categorical'},
 ↪  'hierarchies': {'age': None, 'gender':
 ↪  'data/adult_gender.txt', 'race': 'data/adult_race.txt',
 ↪  'marital_status': 'data/adult_marital_status.txt'},
 ↪  'mondrian_generalization_type': {'age': 'numerical',
 ↪  'education_num': 'numerical'}}}

row count before sanitizing: 32561
row count sanitized: 30162
```

```
{'age': {'(35, 40)': 5, '(50, 55)': 5, '(25, 30)': 5, '(45,
 ↪  50)': 5, '(30, 35)': 5, '(40, 45)': 5, '(20, 25)': 5,
 ↪  '(55, 60)': 5, '(15, 20)': 3, '(75, 80)': 5, '(60,
 ↪  65)': 5, '(70, 75)': 5, '(65, 70)': 5, '(90, 95)': 1,
 ↪  '(80, 85)': 5, '(85, 90)': 3, '(30, 40)': 10, '(50,
 ↪  60)': 10, '(20, 30)': 10, '(40, 50)': 10, '(10, 20)':
 ↪  3, '(70, 80)': 10, '(60, 70)': 10, '(90, 100)': 1,
 ↪  '(80, 90)': 8, '(20, 40)': 20, '(40, 60)': 20, '(0,
 ↪  20)': 3, '(60, 80)': 20, '(80, 100)': 9, '*': 72},
 ↪  'gender': {'*': 2}, 'race': {'*': 5}, 'marital_status':
 ↪  {'*': 7, 'NM': 1, 'Married': 2, 'leave': 2, 'alone':
 ↪  2}}

metrica di loss: 2.0554451968758376

vettore delle generalizzazioni: (1, 0, 1, 2)

soppressione massima: 7
```

Python main.py --mondrian --k 10

```
configuration:
 {'k': 10, 'maxsup': 20, 'samarati': False, 'mondrian':
  ↪  True, 'bank': False, 'data': {'path':
  ↪  'data/adult.data', 'samarati_quasi_id': ['age',
  ↪  'gender', 'race', 'marital_status'],
  ↪  'mondrian_quasi_id': ['age', 'education_num'],
  ↪  'sensitive': 'occupation', 'columns': ['age',
  ↪  'work_class', 'final_weight', 'education',
  ↪  'education_num', 'marital_status', 'occupation',
  ↪  'relationship', 'race', 'gender', 'capital_gain',
  ↪  'capital_loss', 'hours_per_week', 'native_country',
  ↪  'class'], 'samarati_generalization_type': {'age':
  ↪  'range', 'gender': 'categorical', 'race':
  ↪  'categorical', 'marital_status': 'categorical'},
  ↪  'hierarchies': {'age': None, 'gender':
  ↪  'data/adult_gender.txt', 'race': 'data/adult_race.txt',
  ↪  'marital_status': 'data/adult_marital_status.txt'},
  ↪  'mondrian_generalization_type': {'age': 'numerical',
  ↪  'education_num': 'numerical'}}}

row count before sanitizing: 32561
row count sanitized: 30162

====================

metrica di loss: 0.2577096464479936
```

```
====================
Tabella:
          age education_num       occupation
209        17            3-7            Sales
262        17            3-7    Other-service
271        17            3-7    Other-service
335        17            3-7    Other-service
371        17            3-7    Other-service
...       ...            ...              ...
20483   74-90          15-16   Prof-specialty
21473   74-90          15-16   Prof-specialty
21835   74-90          15-16  Exec-managerial
23868   74-90          15-16  Exec-managerial
28176   74-90          15-16  Exec-managerial

[30162 rows x 3 columns]
====================
```

Python main.py --samarati --k 3 --bank

```
configuration:
 {'k': 3, 'maxsup': 10, 'samarati': True, 'mondrian': False,
  ↪  'bank': True, 'data': {'path': 'data/bank.csv',
  ↪  'samarati_quasi_id': ['age', 'job'],
  ↪  'mondrian_quasi_id': ['age', 'job'], 'sensitive':
  ↪  'balance', 'columns': ['age', 'job', 'marital',
  ↪  'education', 'default', 'balance', 'housing', 'loan',
  ↪  'contact', 'day', 'month', 'duration', 'campaign',
  ↪  'pdays', 'previous', 'poutcome', 'y'],
  ↪  'samarati_generalization_type': {'age': 'range', 'job':
  ↪  'categorical'}, 'hierarchies': {'age': None, 'job':
  ↪  'data/bank_job.txt'}, 'mondrian_generalization_type':
  ↪  {'age': 'numerical', 'job': 'categorical'}}}

row count before sanitizing: 4520
row count sanitized: 4520
```

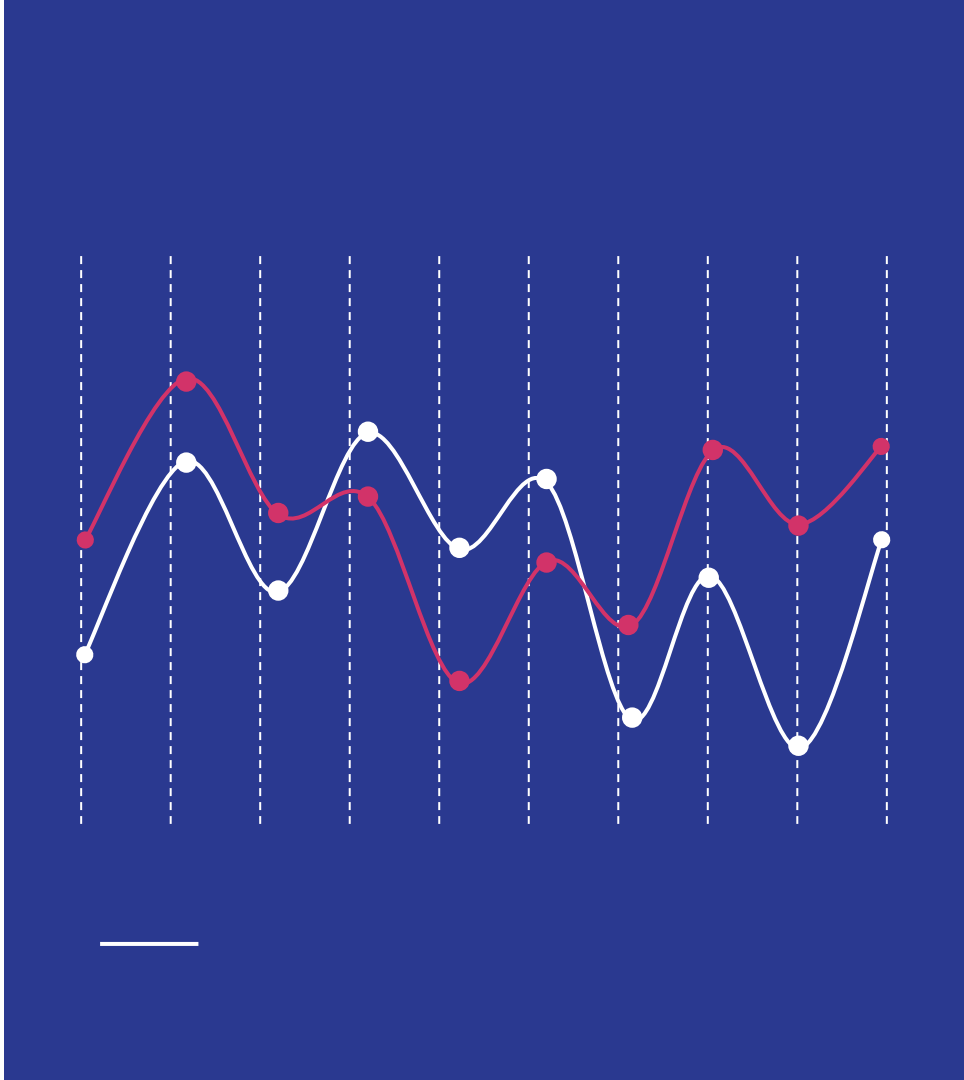
```
{'age': {'(30, 35)': 5, '(35, 40)': 5, '(55, 60)': 5, '(40,
 ↪  45)': 5, '(20, 25)': 5, '(25, 30)': 5, '(65, 70)': 5,
 ↪  '(50, 55)': 5, '(45, 50)': 5, '(75, 80)': 5, '(60,
 ↪  65)': 5, '(70, 75)': 5, '(15, 20)': 1, '(80, 85)': 4,
 ↪  '(85, 90)': 2, '(30, 40)': 10, '(50, 60)': 10, '(40,
 ↪  50)': 10, '(20, 30)': 10, '(60, 70)': 10, '(70, 80)':
 ↪  10, '(10, 20)': 1, '(80, 90)': 6, '(20, 40)': 20, '(40,
 ↪  60)': 20, '(60, 80)': 20, '(0, 20)': 1, '(80, 100)': 6,
 ↪  '*': 67}, 'job': {' employed': 2, 'employed': 1, '
 ↪  white-collar': 2, ' other': 4, ' blue-collar': 4, '
 ↪  self-employed': 2}}

 metrica di loss: 0.28933360150172577

 vettore delle generalizzazioni: (3, 0)

 soppressione massima: 6

 ====================
```

Python main.py --mondrian --k 3 --bank

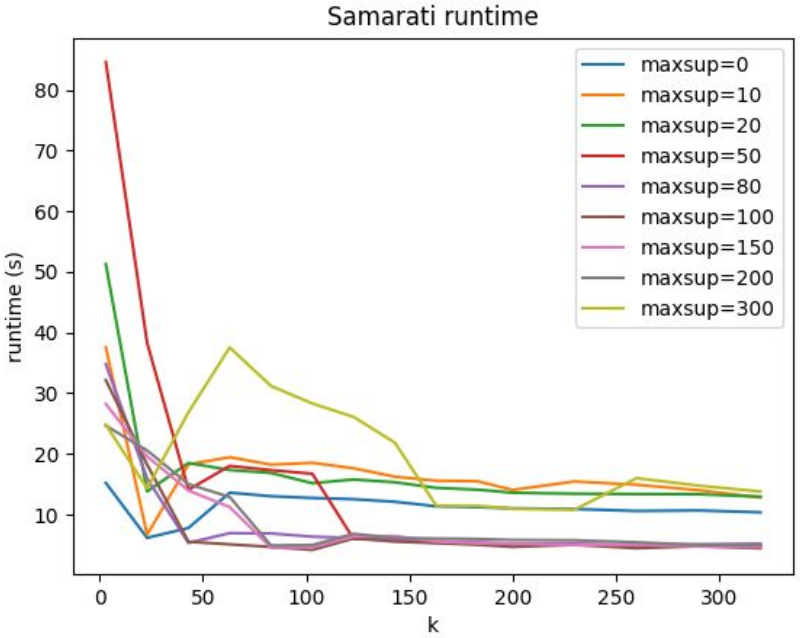
```
configuration:
 {'k': 10, 'maxsup': 20, 'samarati': False, 'mondrian':
 ↪  True, 'bank': True, 'data': {'path': 'data/bank.csv',
 ↪  'samarati_quasi_id': ['age', 'job'],
 ↪  'mondrian_quasi_id': ['age', 'job'], 'sensitive':
 ↪  'balance', 'columns': ['age', 'job', 'marital',
 ↪  'education', 'default', 'balance', 'housing', 'loan',
 ↪  'contact', 'day', 'month', 'duration', 'campaign',
 ↪  'pdays', 'previous', 'poutcome', 'y'],
 ↪  'samarati_generalization_type': {'age': 'range', 'job':
 ↪  'categorical'}, 'hierarchies': {'age': None, 'job':
 ↪  'data/bank_job.txt'}, 'mondrian_generalization_type':
 ↪  {'age': 'numerical', 'job': 'categorical'}}}

row count before sanitizing: 4520
row count sanitized: 4520

====================

metrica di loss: 0.16187029719369647

====================
Tabella:
         age    job  balance
116    22-24    0-1     4111
270    22-24    0-1      174
444    22-24    0-1     1222
773    22-24    0-1      111
1003   22-24    0-1      204
...      ...    ...      ...
```

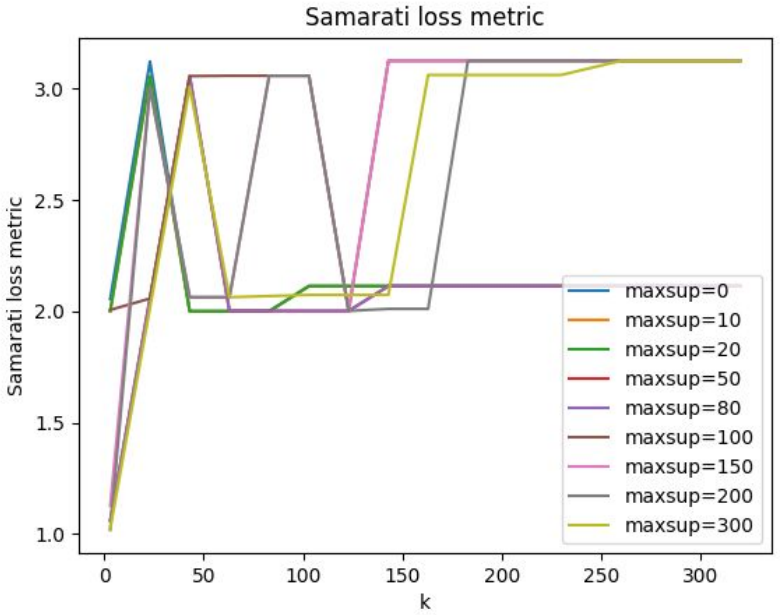
Plotting dei risultati

visualizzazione dei risultati in termini di tempi di esecuzione e metrica di perdita

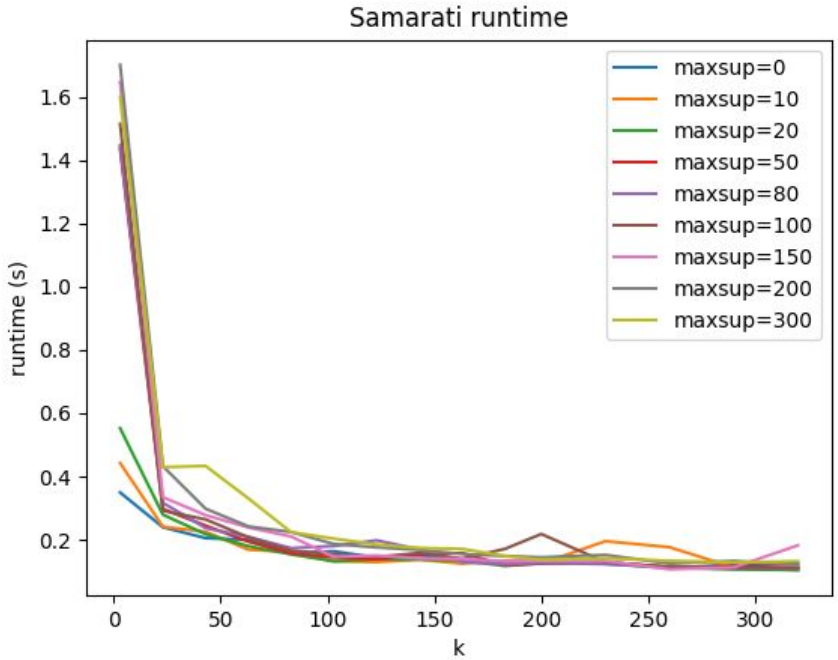


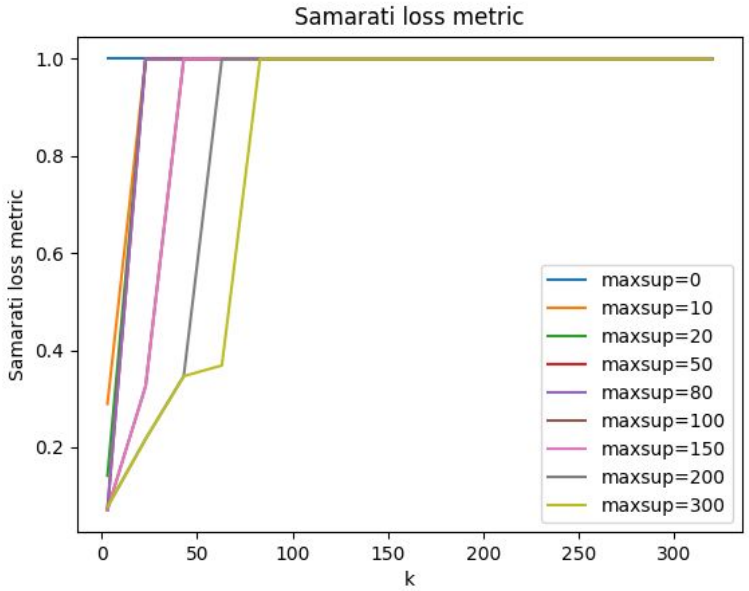
Samarati Adult database



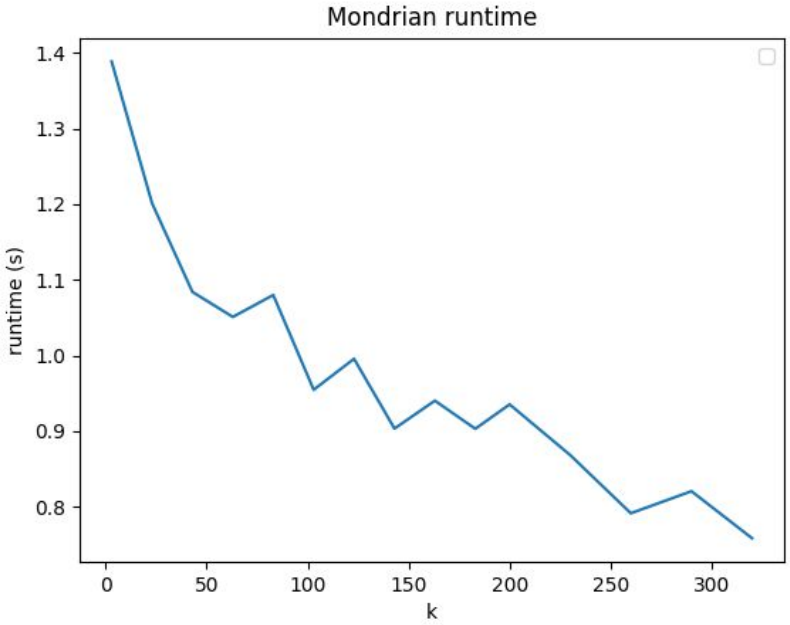


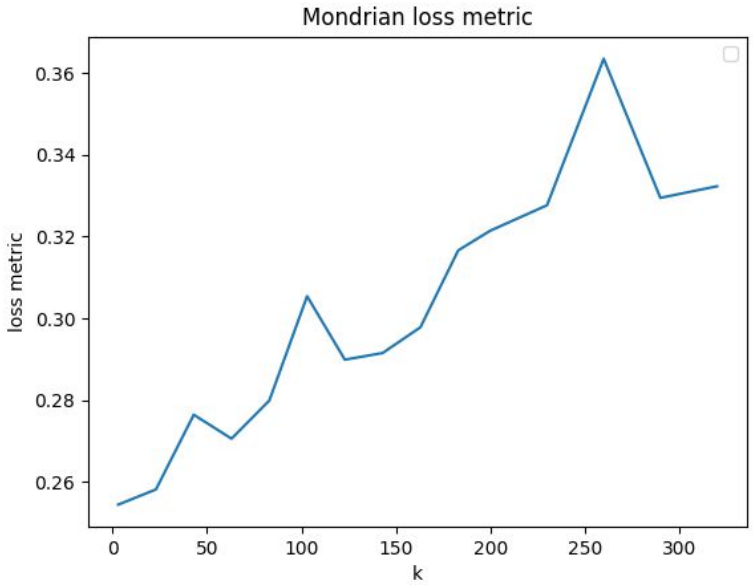
Samarati Bank database



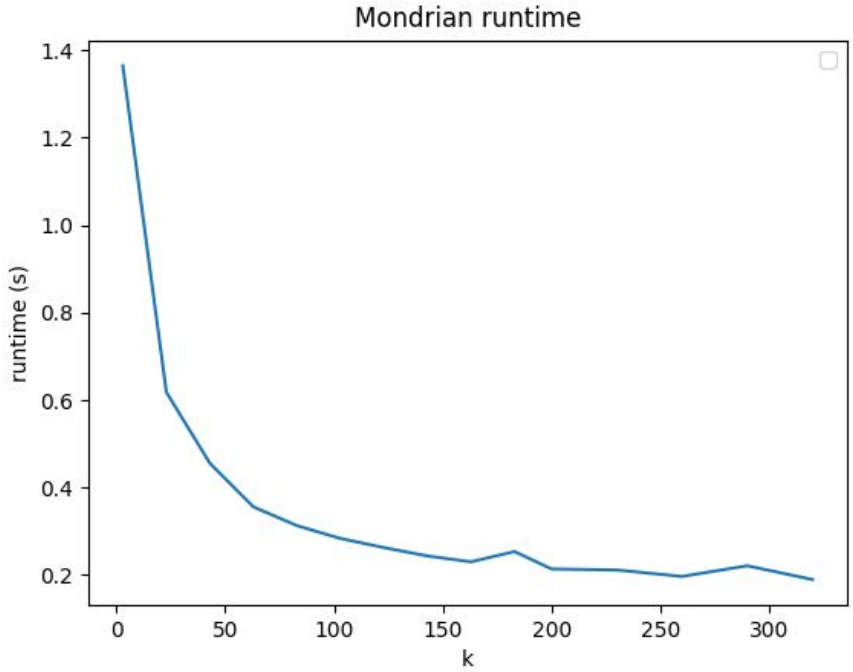


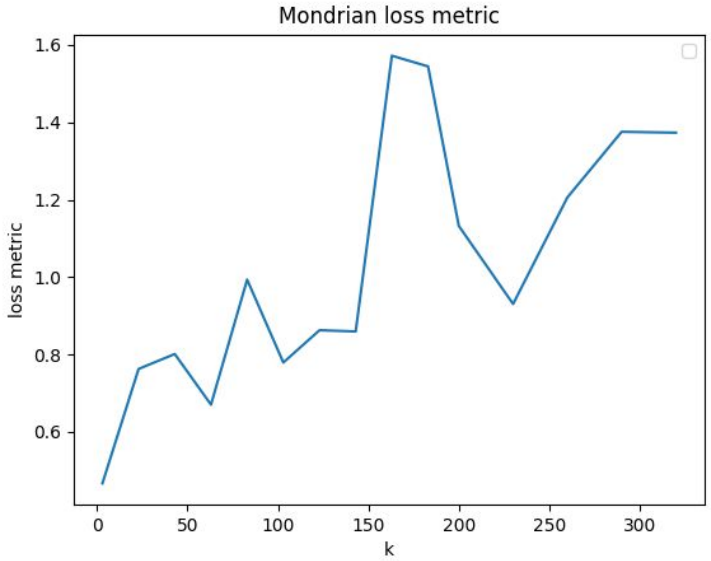
Mondrian Adult database





Mondrian Bank database





Analisi dei risultati

L’esecuzione degli algoritmi sui dataset Adult e Bank dimostra come generalmente entrambi i database funzionino bene nell’anonimizzazione dei dati nei vincoli del k-anonimato.

Valutando i parametri si può notare che la scelta di questi ultimi (k e maxsup) influenzi significativamente il trade-off tra anonimato e perdita di informazioni, in particolare: valori più alti di “k” migliorano l’anonimato ma portano una maggiore perdita dei dati (viceversa diminuendo “k”)

Generalmente si potrebbe dire che in dataset con attributi molti distintivi è necessario un approccio più prudente per evitare una perdita di dati (ad esempio nel dataset adult) mentre in database più piccoli (come bank) o in cui vi è già una minima anonimizzazione è possibile utilizzare una configurazione meno aggressiva.

Test di performance

Volto a determinare le tempistiche di accesso dei dati paragonandole tra tabella originale e tabella anonimizzate (attraverso Mondrian e Samarati)

Il test viene eseguito eseguendo il database tramite SQL lite in memoria locale sia per il dataset Adult che per il dataset Bank, eseguendo una query di ricerca con attributo “age” e valore generico o il suo intervallo di riferimento

Primo test adult

Query Eseguite

1. **Tabella originale:**

```
SELECT * FROM original WHERE age = '39'
```

2. **Tabella anonimizzata (Samarati):**

```
SELECT * FROM anonymizedS WHERE "age" = "(35, 40)"
```

3. **Tabella anonimizzata (Mondrian):**

```
SELECT * FROM anonymizedM WHERE "age" = "39" OR "age" = "38-47" OR "age" = "38-48"
```

Risultati del Test Tempi medi di accesso ai dati:

- Tabella originale: 0.0068 secondi
- Tabella anonimizzata con Samarati: 0.0167 secondi
- Tabella anonimizzata con Mondrian: 0.0066 secondi

Numero di tuple restituite:

- Tabella originale: 786
- Tabella anonimizzata con Samarati: 4085
- Tabella anonimizzata con Mondrian: 956

Secondo test bank

Query Eseguite

1. **Tabella originale:**

```
SELECT * FROM original WHERE age = '33'
```

2. **Tabella anonimizzata (Samarati):**

```
SELECT * FROM anonymizedS WHERE "age" = "*"
```

3. **Tabella anonimizzata (Mondrian):**

```
SELECT * FROM anonymizedM WHERE "age" = "33-35"
```

Risultati del Test Tempi medi di accesso ai dati:

- Tabella originale: 0.0009 secondi
- Tabella anonimizzata con Samarati: 0.0163 secondi
- Tabella anonimizzata con Mondrian: 0.0015 secondi

Numero di tuple restituite:

- Tabella originale: 186
- Tabella anonimizzata con Samarati: 4520
- Tabella anonimizzata con Mondrian: 415

Analisi

L'analisi comparativa ha riportato una grande differenza riguardo l'efficienza nell'accesso dei dati da parte delle tabelle anonimizzate tramite i due algoritmi, vedendo un aumento di circa 2 volte e mezzo del tempo di accesso nella tabella anonimizzata tramite samarati rispetto alla tabella originale, mentre per quanto riguarda l'algoritmo mondrian ha mantenuto i tempi di accesso paragonabili alla tabella non anonimizzata.

Anche le tuple riportate dagli algoritmi hanno evidenziato delle differenze sempre a favore dell'algoritmo Mondrian che garantisce una migliore qualità dei dati riuscendo a generalizzare meglio i QI tramite il clustering.

In generale il differente approccio dei due algoritmi porta ad avere dei migliori risultati nell'accesso ai dati tramite l'algoritmo di anonimizzazione mondrian che quindi dimostra in generale un migliore bilanciamento tra tempi di accesso, qualità dei dati restituiti, tempo di esecuzione e metrica di perdita.

Scenari applicativi

Social Network

Solitamente database di tipo “grafo” (grafo sociale), anonimato non sempre garantito, per il k-anonimato essenziale rendere indistinguibili nodi del grafo da almeno altri “k” nodi

Data Mining / Big Data

1. Anonyme and Mine -> approccio conservativo, anonimizzare preventivamente i dati, poco elevato per costo computazionale alto.
2. Mine and Anonymize -> analisi dei dati eseguita da soggetto di fiducia, dati anonimizzati dopo l’analisi, prima della pubblicazione

Dati di localizzazione

- Privacy di locazione (posizione precisa) posizione di un utente in cluster di posizioni con almeno altri “k” utenti
- Privacy della traiettoria (percorso) modifica e mescolamento dei percorsi in modo che sia condiviso da almeno “k” utenti

Conclusioni

Conclusioni

K-anonimato rappresenta approccio fondamentale nella protezione dei dati sensibili.

Test degli algoritmi:

- Samarati: tecnica classica del k-anonimato garantisce il k-anonimato sia per grandi che per piccole basi di dati in modo abbastanza efficiente, a livello di velocità di accesso ai dati presenta alcune criticità, in particolar modo quando la generalizzazione diventa più aggressiva.
- Mondrian: tecnica di clustering multidimensionale per il k-anonimato, molto efficiente sia per quanto riguardo tempistiche di esecuzione sia per metriche di perdita. Ottima in particolar modo anche per l'accesso ai dati mantenendo tempi di accesso e tuple analizzate paragonabili alla tabella originale.