**DATA PUBBLICATION**

Protezione dei dati nelle cosenguenti release e pubblicazioni dei dati stessi.

Differenza tra statistical DBMS e statistical data:

DBMS = database che risponde solo a query statistiche, mentre statistical data sono dati che riguardano una specifica statistica.

**Microdata vs Macrodata**

I microdati sono dati che rappresentano un individuo specifico, possono essere utilizzati per maggiori attività rispetto a macrodati es: machine learning, di conseguenza possono avere un valore maggiore ma hanno anche più possibilità di essere non protetti es: linking attack.

I macrodati possono essere rappresentati da una count/frequency table o da una magnitude table:

Una count/freq table sono due tabelle differenti in cui ogni cella presenta un conteggio (count) oppure una percentuale (freq) degli individui relativa ad un certo attributo.

Una magnitude table è una tabella in cui le celle presentano un valore aggregato relativo ad un certo attributo es: la media .

**Disclosure**

Identity disclosure, attribute disclosure, inferencial disclosure:

Identity: consiste nel rivelare l’identità di un rispondente tramite la release dei dati,

Attribute: consiste nel rivelare dati sensibili relativi ad un rispondente tramite release dei dati

Inferencial: l’inferenza consiste nel rivelare dati di un soggetto non presente nella release dei dati ma comunque che possono essere ottenuti da essa es: forte correlazione tra income e prezzo di case, posso inferire l’income di altri rispondenti conoscendo il prezzo delle case che è un dato pubblico.

L’identity disclosure non è sempre un problema, per i macrodati non è motivo di preoccupazione a meno che porti alla scoperta di dati sensibili (attribute disclosure). Per i microdati è motivo di preoccupazione in quanto implica un’attribute disclosure.

L’attribute disclosure avviene sia se i dati sono rivelati precisamente o stimati quasi precisamente,

L’inferencial disclosure non è sempre un problema, avviene grazie all’utilizzo di proprietà statistiche e solitamente l’inferenza non riesce ad ottenere dati di singoli individui ma viene fatta tipicamente su quantità statistiche.

**Protezione**

Si può avere protezione avendo delle restrizioni sulle informazioni fornite dalle release dei dati o avendo delle restrizioni in accesso a questi dati o un insieme delle due.

La protezione di macrodati avviene tramite tecniche di sampling, special rules o threshold rules:

Sampling: viene preso una parte dei dati totali a disposizione,

Special rules: limito il dettaglio dell’informazione che può essere rivelato,

Threshold rules: pubblico solo certi dati i quali corrispondono a una certa quantità di rispondenti sopra una certa soglia.

Per la protezione di microdati ho masking techniques e synthetic data generation:

Masking: possono essere perturbative o non perturbative, le tecniche perturbative modificano i dati, mentre le tecniche non perturbative non modificano i dati ma li sopprimono o rimuovono prima delle release.

Synthetic: la generazione di dati sintetici può essere completa o parziale, una generazione di dati sintetici completa vuol dire che l’intera quantità di dati generati è completamente sintetica, mentre parziale significa che i dati vengono un po’ generati sinteticamente e il resto viene mantenuto come da originale.

**DeIdentification**

La deidentificazione non basta per rendere sicure le informazioni, gli attributi di un rispondente possono essere identificativi, quasi-identificativi confidenziali o non confidenziali:

Gli attributi identificativi permettono l’identificazione del rispondente con una granularità 1:1 sono unici per quel rispondente,

Gli attributi quasi-identificativi sono attributi che insieme rendono il rispondente fragile a linking attacks quindi aumentano la possibilità che il soggetto venga re-identificato grazie al collegamento con fonti esterni di dati.

Gli attributi confidenziali sono attributi che contengono dati sensibili,

Gli attributi non confidenziali non contengono dati sensibili e non comportano rischio di disclosure.

La possibilità di essere linkati a fonti esterne dipende dalla quantità di fonti esterne, dalla granularità dell’informazione rappresentata nella release e dalla quantità di attributi in comune con la fonte esterna; è anche da considerare la possibilità che le fonti esterne non utilizzino lo stesso formato per la rappresentazione dei dati e inoltre i dati possono anche non essere aggiornati, di conseguenza più difficili da linkare perché rappresentano informazioni diverse dovute al dislivello temporale.

**K-anonimity**

Approccio per proteggere l’indentità dei rispondenti insieme a generalitazion e suppression.

I dati devono essere indistinguibili in relazione a un certo numero k di rispondenti, ogni tupla avrà questa caratteristica.

I concetti di generalizzazione (proteggo l’informazione utilizzando valori più generici) e soppressione (proteggo l’informazione eliminandola) sono fondamentali per ottenere una k-anonimity.

Domain generalization hierarchy != value generalitazion hierarchy.

Domain si basa sui domini e una relazione di generalizzazione sui domini del tipo Di <=d Dj significa che gli elementi in Dj sono generalizzazioni degli elementi in Di, quindi sono più generali.

La gerarchia riguarda i domini e non i valori dei domini, in questo caso la root è singleton.

Nella value generalization hierarchy non prendo in considerazione i domini ma i valori, l’elemento root apparterrà al dominio root, in questo caso la relazione Vi <=v Vj indica che ogni elemento in Vi viene generalizzato da un elemento in Vj.

Tabella generalizzata: Ti<= Tj:

Gerarchia = albero

Tj è una rabella più generale con soppressione se la cardinalità di Tj <= a quella di Ti, se gli attributi di Tj sono uguali o una generalizzazione degli attributi di Ti, e se è possibile definire una fuzione iniettiva dalle tuple di Tj a quelle di Ti in modo che quelle di Tj siano una generalizzazione o uguali a quelle di Ti.

Il distance vector

Date due tabelle Ti <= Tj il distance vector è un vettore Di,j = [d1 .. dn] dove ogni dz, z=1..n indica la distanza del path minimo dal dom(Az, Di) al dom(Az, Dj) nella domain generalization hierarchy.

K-minimal generalization

Devo iniziare a ragionare, ma quale parte della gerarchia è meglio utilizzare nel mio caso? Me lo faccio dire da questo concetto di DISTANCE VECTOR.

Se mi chiedono k-anom = 2 e generalizzo di + perdo di utilità perciò devo fare una k-anom di 2 che sarà la migliore. Devo cercare di mantenere l’utilità, NON STRAFARE NON FARE PIù PRIVACY DI QUELLA RICHIESTA. !!!

Se ti chiedo di k-anonim di 2 e arrivo a 2 non devo generalizzare di più, non esistono soluzioni specifiche che già hanno quella proprietà lì

Meglio sopprimere o generalizzare? Dipende, se ho pochi outliers sopprime se ne ho tanti generalizzo. Se posso fare entrambe, finchè rimango nella soglia del MaxSupp preferisco sopprimere dopo di chè sarebbe meglio generalizzare.

Proprietà:

1. Tj non sopprime più di quanto serve, (nel vettore di distanza), al vettore di distanza non sopprimo più di quelle necessarie per cancellare gli outliers
2. Non sopprimo più tuple di quelle che è possibile sopprimere ti – tj <= MaxSupp
3. Ce una soluzione che mi generalizza di meno e mi da la stessa protezione non oltrepassando la soglia di soppressione? Allora scelgo quella

In sostanza mi dice di scegliere il meno generale e che mi mantenga un certo livello di soppressione e di protezione.

K perché deve rispettare una certa k-anonimimty.

Preferred Generilization

Minimum absolute distance: preferisco la generalizzazione che mi permette di avere il minor numero di passi di generalizzazione, ovviamente sull’albero del dominio (distance vector).

Minimum relative distance: preferisco la generalizzazione che mi permette di avere il minor numero di passi relativi di generalizzazione, passo relativo significa uno step in avanti nella gerarchia diviso per l’altezza dell’albero.

Maximum distribution: preferisco la generalizzazione con il maggior numero di tuple diverse tra loro

Minimum suppression: preferisco la generalizzazione che sopprime il minor numero di tuple a.k.a. massima cardinalità

K-anonim techniques

Generalizzazione: un dato generalizzato è un dato che porta con sé meno informazione, la generalizzazione può essere applicata a livello di colonna o cella,

La soppressione: consiste nella protezione di un dato tramite eliminazione, si applica a livello di colonna, cella o riga

Algorithms for computing k-anonim

Gli algoritmi esatti proposti hanno sempre un tempo esponenziale rispetto al numero di attributi che compongono i quasi-Identifiers.

**Algorithms for AG\_TS and AG\_:**

Soluzione non minimale: c’è qualcuno che tu domini che è anch’esso una soluzione, che significa aver computato più passi di generalizzazione di quelli necessari mantenendo lo stesso livello di protezione e soppressione richiesto.

Soluzione localmente minima: soluzione che mi risolve il problema il prima possibile sul path di gerarchia, non è detto che sia minima assoluto perché è minima relativa ad un path ma non per forza minima relativamente a tutti i path.

K-minimal algorithm: prende i path dell’albero del dominio e cerca le soluzioni localmente minimo, se ne esiste almeno una allora procede con una ricerca binaria, cioè cerca la soluzione ad altezza ½ h, se la trova valuta altezza ¼ se non la trova valuta ¾ h e così via finchè non trova la soluzione di altezza più bassa possibile nell’albero di gerarchia.

Utilizza una distance vector Matrix, su righe e colonne sono rappresentate le tuple, sulle righe non sono presenti tuple che hanno già raggiunto un certo k-anonim. Richiesto.

Una cella nella matrice indica la distanza che bisogna percorrere in termini di step sull’albero di gerarchia per far sì che le tuple siano uguali.

K-Optimize:

Il cambio di dati per passare ad un problema successivo provoca un cambio del software questo perché il software sfruttava la monotonicità dei dati, perciò utilizzo il k-optimize in cui non ho una gerarchia predefinita ma la determino io, li metto insieme per risolvere il problema.

Posso mettere insieme solo elementi vicini, adiacenti, ogni nodo ha un proprio costo di generalizzazione e soppressione.

Parto dalla radice e faccio DFS, se in un nodo non ho una soluzione allora tagglio tutto il sotto albero, se ho soluzione vado più in basso. INTERESSA MODO DIVERSO FORMULAZIONE PROBLEMA.

Incognito:

Sapendo che prendendo un sottoinsieme di Q.I. la k-anonimity su quel sottoinsieme è condizione NECESSARIA per avere k-anonimity in relazione al gruppo totale di Q.I.

Condizione necessaria non sufficiente: dipende come li combino esempio z1,z2 r1,r2

Se li combino insieme male ottengo k anonm =1 invece nei sottoinsieme ho k anonim = 2. perciò non è condizione sufficiente per k-anonim generale.

Posso utilizzare un approccio bottom-up e prendere ad ogni iterazione le generalizzazioni che soddisfano k-anonimity e ad ogni iterazione successiva mettere insieme i passi precedenti fino ad arrivare alla soluzione alla iterazione |QI|.

Quindi iterazione 2 metterò insieme le coppie di attributi e scarto quelle che non raggiungono k-anonimity nell’iterazione 3 metterò insieme le triple ecc ecc..

Heuristic algorithms:

Se gli algoritmi esatti hanno un tempo di computazione esponenziali, questi euristici cercano di raggiungere un tempo di computazione minore basandosi su una euristica. Non assicurano una soluzione ottimale.

**Algorithms for \_CS and CG\_**

Mondrian:

Il più versatile di tutti, ogni attributo è una dimensione, ogni tupla è un punto nello spazio, tuple uguali rappresentano una molteplicità.

Questo algoritmo taglia lo spazio in modo da avere k elementi in ogni partizione e computare quindi la generalizzazione per avere un unico valore derivante dalla partizione, quindi rendere i k-elementi anonimi nella partizione.

Ogni passo fa un taglio nello spazio

----------------------------------------------------------------------------------------------------------------

**K-anonimity rivisited**

Ho k-anonim rivisited se ho un mapping dalla tabella originale a quella generalizzata e viceversa che mi permette di avere k elementi nel mapping.

Ovvero quando faccio mapping da una tabella all’altra, prendendo un elemento da una ottengo k elementi nell’altra e viceversa.

-----------------------------------------------------------------------------------------------------------------

**L-diversity**

Un q-block (tuple che hanno lo stesso valore per i QI) è l-diverso se possiede almeno L valori “well-represented” per ogni attributo sensibile.

Well-represented: definizione basata sulla ricorsione, un blocco è l-diverso se togliendo un valore sensibile rimane l-1 diverso

Una tabella è l-diversa se tutti i suoi q-block sono l diversi.

Il problema della l-diversity è che protegge da certi attacchi basati su external knoledge ma rende più comuni attacchi basati sulla distribuzione quindi skewness e similarity.

Skewness: la distribuzione in un q-block non rispecchia la distribuzione nella realtà.

Similarity: i valori sensibili sono diversi ma semanticamente uguali.

T-Closeness: la differenza tra la distribuzione nel q-block e la distribuzione reale è minore di una certa soglia t. Una tabella è t-close se tutti i q-block sono t-close.

External Knowledge

Può riguardare un individuo, gli altri soggetti oppure un gruppo di individui con le stesse caratteristiche sensibili (same-value family)

**Multiple release**

È importante che release differenti non siano indipendenti altrimenti si incorre nell’intersection attack, cioè mettendo insieme le informazioni di release differenti riesco ad ottenere delle informazioni utili rispretto a dei rispondenti.

M-invariance

Release multiple rispettano la m-invariance se e solo se:

1. Ogni classe di equivalenza contiene almeno m tuple
2. I valori sensibili non appaiono più di una volta in ogni classe di eq.
3. Per ogni tupla t: per ogni classe di equivalenza a cui t appartiene sono presenti sempre gli stessi insiemi di valori sensibili

Se queste condizioni sono rispettate non è possibile associare meno di m valori sensibili ad ogni rispondente

**Proteggere privacy in applicazioni location based ed esempi di re identificazione sono discorsivi.**

**Differential privacy**

La differential privacy punta a far in modo che la presenza o l’assenza di un individuo all’interno di un dataset non faccia differenza, evita che un attaccante scopra la presenza o meno di un individuo all’interno del dataset.

Se due dataset applicano questo principio, la stessa query sui due dataset ritornerà probabilmente lo stesso risultato.

Viene implementata aggiungendo del random noise e viene impiegata in due scenari:

1. Interactive scenario: valutazione a run-time di queries
2. Non-interactive scenario: rilascio di tabelle di macro-dati

Sono presenti delle varianti per diminuire il noise nei risultati.

Come la k-anonimity si è sviluppata in diversi domini es social networks.

Limitare inferenza rispetto alla presenza di individuo è diverso dal limitarne la partecipazione nella generazione dei dati, infatti in un social network la partecipazione ad esso genera link diversi causati dalle amicizie, perciò se anche l’individuo è coperto dalla differntial privacy potrebbe essere rivelato comunque a causa di questi link che vengono generati dalla partecipazione e non dalla presenza.

**Altre forme di privacy:**

Sensitive value distributions possono comportare leakage di dati che non potrebbero essere inferiti da singole tuple, ma vengono inferiti dalla distribuzione delle tuple.

Es: camping dei soldati, viene inferito dall’età dei soldati che si trovano tutti nello stesso posto.

Età giovane = campo allenamento, età anziana = headquarters.

Bisogna combattere l’inferenza capendo qual è l’informazione sensibile da proteggere.

Genomic data

Riguarda dati relativi ai geni degli umani, una rivelazione di essi porta all’identificazione del proprietario nonché alla rivelazione di caratteristiche fisiche, predisposizioni a malattie, e discendenza/parenti del proprietario.

Inferenza dal data mining: shopping, social networks ecc...