Privacy-Preserving Mining of Association Rules from Outsourced transaction Databases

F. Giannotti, L.V. Lakshmanan, **A. Monreale**, D .Pedreschi, W. Wang

SPCC 2010, Brussels, January 2010

Outline

- Introduction
- The problem
- Model Privacy
- Encryption/Decryption Schema
- Preliminary Experimental results
- Conclusion

Introduction

- Availability of large transactional database
- Data are an important resource for an organization if
  - Processed
  - Analyzed
  - Transformed in Knowledge by KDD techniques
- Mining the data requires
  - **Computational resources**
  - **In-house expertice** for data mining

Outsourcing of Data Mining

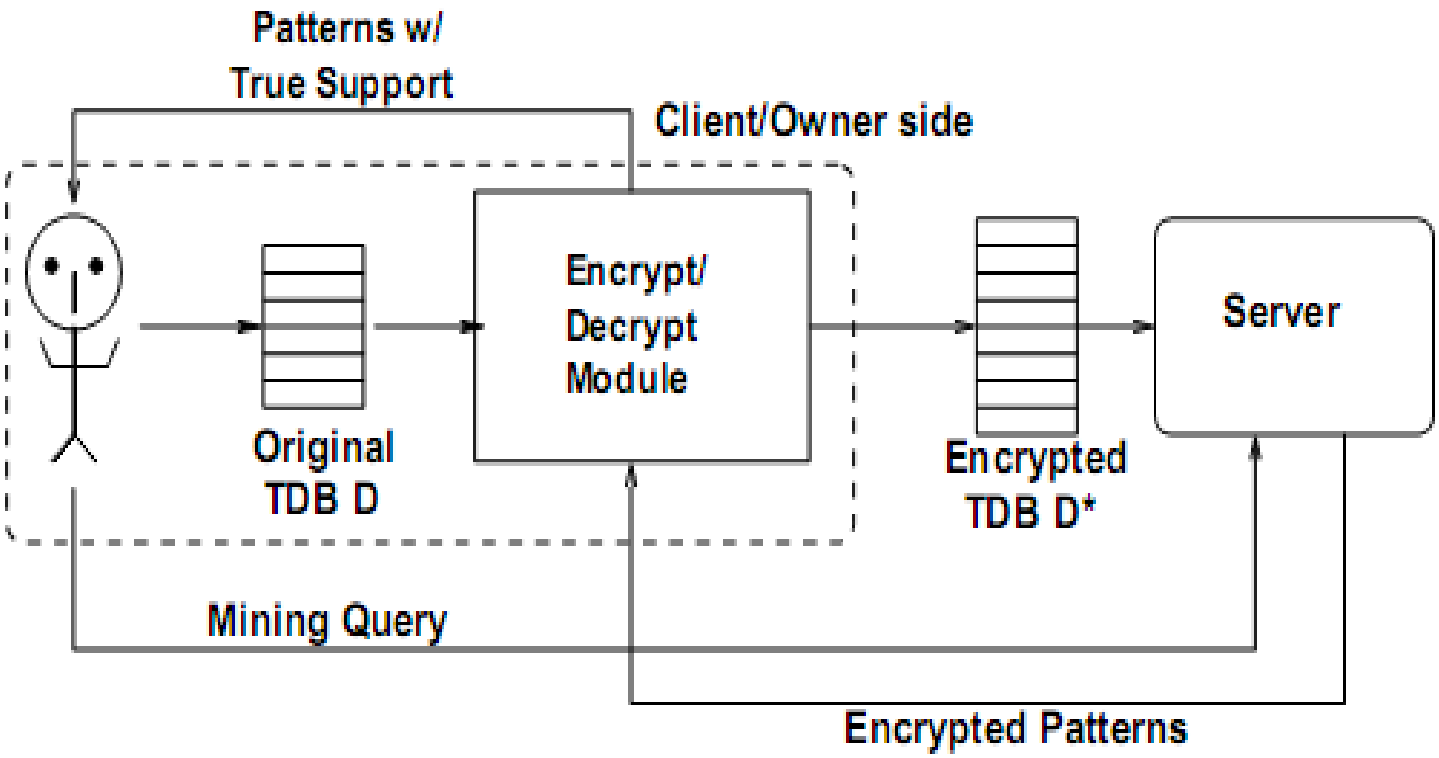
- Development of cloud computing vs the paradigm of **data mining as-a-service**

- **Idea**: Outsourcing of data mining to a service provider
  - specific human resources
  - technological resources

PP Outsourcing of Data Mining

- The server accesses to data of the owner
- Data owner has the property of both
  - **Data** can contain personal information about individuals
  - **Knowledge** extracted from data can provide competitive advantages
- **Solution**: Privacy-Preserving Outsourcing of data mining tasks to protect the corporate privacy
  - Association Rules
  - Classification
  - ....

Framework Architecture



The Problem

PROBLEM: *Given a plain database D, we want to construct a private encrypted database D* such that:*

- *all encrypted transactions in D* and items contained in it are private*
- *given any pattern mining query the server can compute the encrypted result*
- *encrypted pattern mining results are private*
- *the owner can decrypt the results and so, reconstruct the exact result*
- *the space and time incurred by the owner in the process has to be minimum*

Pattern Mining Task

- Given a Database of transactions

TDB
Bread
Milk Bread
Bread Milk
Water Milk
Bread Beer
Bread Eggs
Water

- The **support** of an item set (pattern) S is defined as

$$supp(S) = |\ \{t \in D \mid S \subseteq t\}\ |$$

Pattern Mining Task

- Given a Database of transactions D:

TDB
Bread
Milk Bread
Bread Milk
Water Milk
Bread Beer
Bread Eggs
Water

$supp(\{milk, bread\}) = 2$

- The **support** of an item set (pattern) S is defined as

$$supp(S) = |\{t \in D \mid S \subseteq t\}|$$

- **Pattern mining**: given D *and a support threshold* σ

$$FP = \{p \mid Supp(p) \geq \sigma\}$$

Pattern mining Task

- Given a Database of transactions D:

TDB
Bread
Milk Bread
Bread Milk
Water Milk
Bread Beer
Bread Eggs
Water

$\sigma = 2$

{milk, bread}
{milk}
{water}
{bread}

- The **support** of an item set (pattern) S is defined as

$$supp(S) = |\{t \in D \mid S \subseteq t\}|$$

- **Pattern mining**: given D *and a support threshold* σ

$$FP = \{p \mid Supp(p) \geq \sigma\}$$

Privacy Model

- Consider a ***conservative model***
- The attacker
  - knows the set of plain items and their true supports in D exactly
  - has access to the encrypted database D^*

Item	Sup
Bread	5
Milk	3
Water	2
Beer	1
Eggs	1

K-Anonymous TDB
e_4 e_1
e_4
e_3
e_2
e_2 e_4
e_4 e_2
e_4 e_5
e_2 e_1
e_2 e_3
e_5

Privacy Model

- Consider a ***conservative model***
- The attacker
  - knows the set of plain items and their true supports in D exactly
  - has access to the encrypted database D^*

- Two kinds of attacks
  - ***Item-based attack:*** *guessing the correct plain item corresponding to the cipher item* e *with probability* $prob(e)$
  - ***Itemset-based attack:*** *guessing the correct plain itemset corresponding to the cipher itemset* E *with probability* $prob(E)$

Goal and Ideal Solution

- **Goal:** minimize the probabilities of crack of
  - an item $prob(e)$
  - an itemset (transaction or pattern) $prob(E)$

- **Ideal Solution:**
  - every cipher item should have as candidates all the items in D
  - every cipher itemset should have as candidates all the itemset with same size in D

- **Problem:** explosion in the computational effort required for mining patterns from D^*

k-Privacy

Definition 1 (Item k-anonymity). Let D be a transaction database and D^* its encrypted version. We say D^* satisfies the property of *item k-anonymity* provided for every cipher item $e \in \mathscr{E}$, there are at least $k-1$ other distinct cipher items $e_1, \ldots, e_{k-1} \in \mathscr{E}$ such that $supp_{D^*}(e) = supp_{D^*}(e_i)$, $1 \leq i \leq k-1$. $\square$

Definition 2 (k–Privacy). Given a database D and its encrypted version D^*, we say D^* is *k-private* if:

(1) for each cipher item $e \in D^*$, $prob(e) \leq 1/k$; and

(2) for each cipher itemset E with support $supp_{D^*}(E) > 0$, $prob(E) \leq 1/k$. $\square$

Encryption Schema

- Main steps of our Encryption schema:
  - 1-1 substitution ciphers for each plain item
  - k-grouping items
  - adding new fake transactions for having k-anonymity

TDB
Bread
Milk Bread
Bread Milk
Water Milk
Bread Beer
Bread Eggs
Water

TDB
e_2
e_2 e_4
e_4 e_2
e_4 e_5
e_2 e_1
e_2 e_3
e_5

+

Fake trans.
e_4 e_1
e_4
e_3

Robust K-Grouping Method

- Obtaining ***Robust k-groups*** that is unsupported in D

RobFrugal Grouping:

Given D and its item support table in decreasing order of support:

TDB
e_2
e_2 e_4
e_4 e_2
e_4 e_5
e_2 e_1
e_2 e_3
e_5

Robust K-Grouping Method

- Obtaining ***Robust k-groups*** that is unsupported in D

TDB
e_2
e_2 e_4
e_4 e_2
e_4 e_5
e_2 e_1
e_2 e_3
e_5

RobFrugal Grouping:

Given D and its item support table in decreasing order of support:

STEP1: grouping together cipher items into groups of k adjacent items starting from the most frequent item e_1, obtaining the grouping

$G = (G_1, \ldots, G_m)$

Item	Support
e_2	5
e_4	3
e_5	2
e_1	1
e_3	1

.

Robust K-Grouping Method

- Obtaining ***Robust k-groups*** that is unsupported in D

TDB
e_2
e_2 e_4
e_4 e_2
e_4 e_5
e_2 e_1
e_2 e_3
e_5

RobFrugal Grouping:

Given D and its item support table in decreasing order of support:

STEP1: grouping together cipher items into groups of k adjacent items starting from the most frequent item e_1, obtaining the grouping

$\mathbf{G = (G_1, \ldots, G_m)}$

Item	Support
e_2	5
e_4	3
e_5	2
e_1	1
e_3	1

STEP2: modifying the groups of G by repeating the following operations, until no group of items is supported in D:

1.Select the smallest j ≥ 1 such that $\mathbf{supp_D(G_j) > 0}$,

2.Find the most frequent item $\mathbf{i'} \notin \mathbf{G_j}$ such that, for the least frequent item $\mathbf{i}$ of $\mathbf{G_j}$:

1. $\mathbf{supp_D(G_j \setminus \{i\} \cup \{i'\}) = 0}$,
2. Swap $\mathbf{i}$ with $\mathbf{i'}$ in the grouping.

Item	Support
e_2	5
e_5	2
e_4	3
e_1	1
e_3	1

How create Fake Transactions?

- Output of the grouping step is a **Noise table**

Item	Support	Noise
e_2	5	0
e_5	2	3
e_4	3	0
e_1	1	2
e_3	1	2

How create Fake Transactions?

- Output of the grouping step is a **Noise table**

Item	Support	Noise
e_2	5	0
e_5	2	3
e_4	3	0
e_1	1	2
e_3	1	2

2-Anonymity

Item	Noise
e_5	3
e_3	2
e_1	2

Fake transactions

$\{e_5\}$

$\{e_5, e_3, e_1\}$

$\{e_5, e_3, e_1\}$

How create Fake Transactions?

- Output of the grouping step is a **Noise table**

Item	Support	Noise
e_2	5	0
e_5	2	3
e_4	3	0
e_1	1	2
e_3	1	2

2-Anonymity

Item	Noise
e_5	3
e_3	2
e_1	2

Fake transactions

$\{e_5\}$

$\{e_5, e_3, e_1\}$

$\{e_5, e_3, e_1\}$

$L > L_{max}$

How create Fake Transactions?

- Output of the grouping step is a **Noise table**

Item	Support	Noise
e_2	5	0
e_5	2	3
e_4	3	0
e_1	1	2
e_3	1	2

2-Anonymity

Item	Noise
e_5	3
e_3	2
e_1	2

Fake transactions

$\{e_5\}$

$\{e_1\}$ $\{e_1\}$

$\{e_5, e_3\}$ $\{e_5, e_3\}$

Synopsis in side-client

- The noise table provides a compact ***synopsis***
  - used for decryption to compute the true support of a pattern
  - represents the fake transactions
- Hash table created with a ***minimal perfect hash function***

Item	Noise
e_5	3
e_3	2
e_1	2

#NoiseItems / L_{max} = 2

HASH TABLES

	Table1
0	$\langle e_5, 1, 2 \rangle$
1	$\langle e_3, 2, 0 \rangle$

	Table2
0	$\langle e_1, 2, 0 \rangle$

Synopsis in side-client

- The noise table provides a compact ***synopsis***
  - used for decryption to compute the true support of a pattern
  - represents the fake transactions
- Hash table created with a ***minimal perfect hash function***

Item	Noise
e_5	3
e_3	2
e_1	2

e_5 = item
1 = $\{e_5\}$ occurs 1 times
2 = $\{e_5, e_3\}$ occurs 2 times

	Table1
0	$\langle e_5, 1, 2 \rangle$
1	$\langle e_3, 2, 0 \rangle$

	Table2
0	$\langle e_1, 2, 0 \rangle$

Decryption: How use the synopsis?

- The client receives frequent patterns mined over D*
- Synopsis allows computing the actual support of every pattern

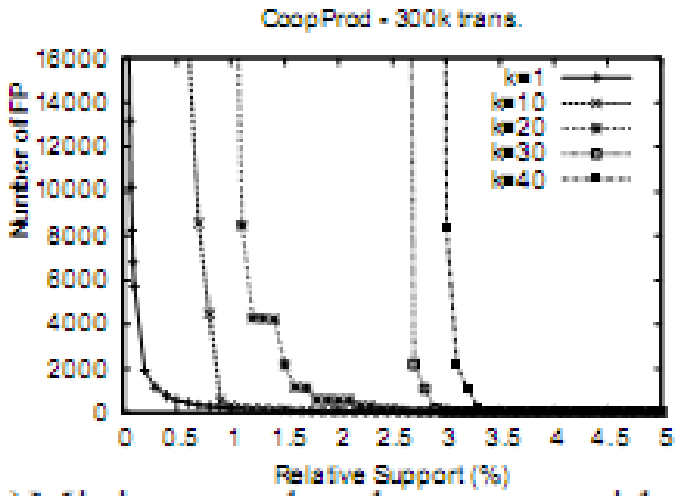
Item	Support	Noise
e_2	5	0
e_5	2	3
e_4	3	0
e_1	1	2
e_3	1	2

	Table1
0	$\langle e_5, 1, 2 \rangle$
1	$\langle e_3, 2, 0 \rangle$

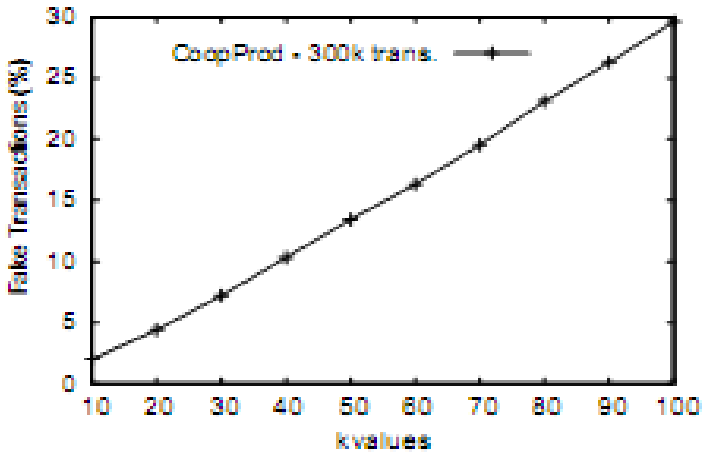
	Table2
0	$\langle e_1, 2, 0 \rangle$

- $RS(\{\mathbf{e_5}\}) = supp_{D^*} - supp_{D^* \setminus D} = 5 - (1 + 2) = 2$
- $RS(\{\mathbf{e_5}, \mathbf{e_3}\}) = supp_{D^*} - supp_{D^* \setminus D} = 2 - (2 + 0) = 0$

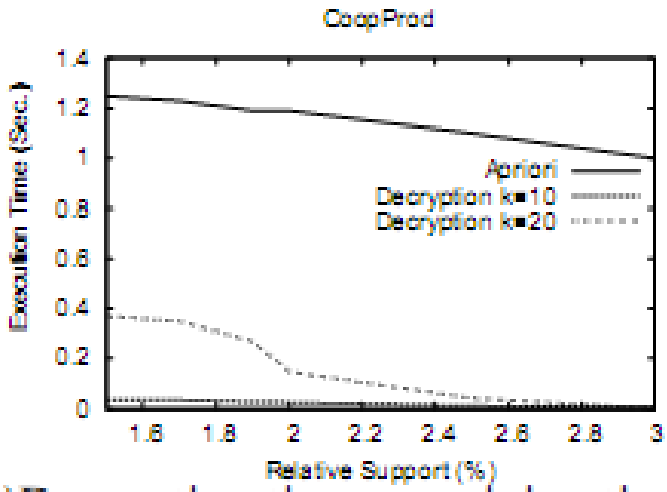
Client & Server Overhead



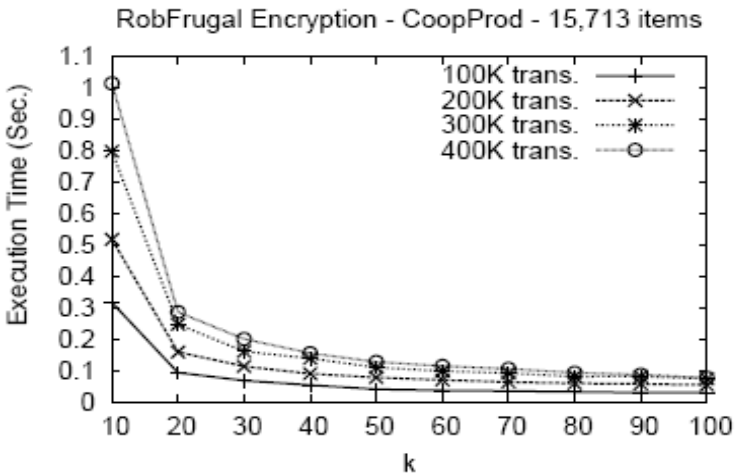
(a)Mining overhead at server side



(b)Fraction of fake transactions



(c)Decryption time vs. mining time



Encryption overhead on CoopProd

Conclusion & Future work

- An Enryption/Decryption Schema for privacy-preserving outsourcing of association rules mining
- Preliminary experiments on large real database
- Issues to be addressed:
  - Complexity Analysis
  - Privacy analysis to prove that the crack probability can be controlled
  - Strategy for incrementally maintaining the synopsis