Building Decision Tree Classifier on Private Data

Author: Wenliang (Kevin) Du and
Zhijun Zhan

Center for Systems Assurance
Electrical Engineering and Computer Science
Syracuse University

1/3/2003

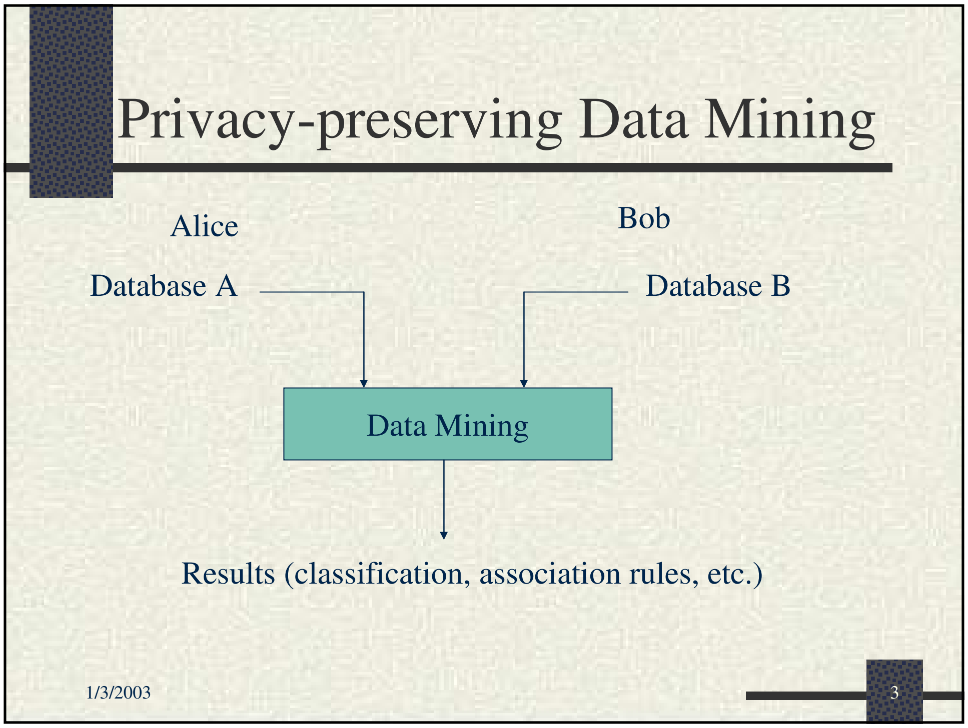
1

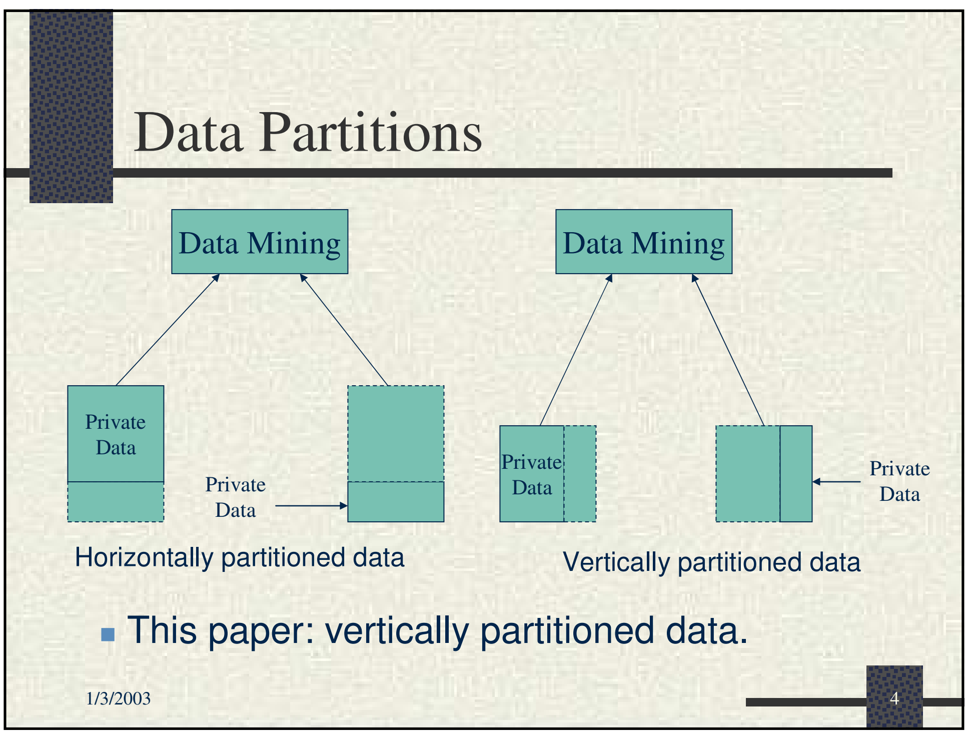
Overview

- Privacy-preserving Data Mining
- Problem Definition
- Building Block: Scalar Product Protocol
- Decision Tree Building
- Conclusion

1/3/2003

2


Privacy-preserving Data Mining
Alice
Bob
Database A
Database B
Data Mining
Results (classification, association rules, etc.)
1/3/2003
3

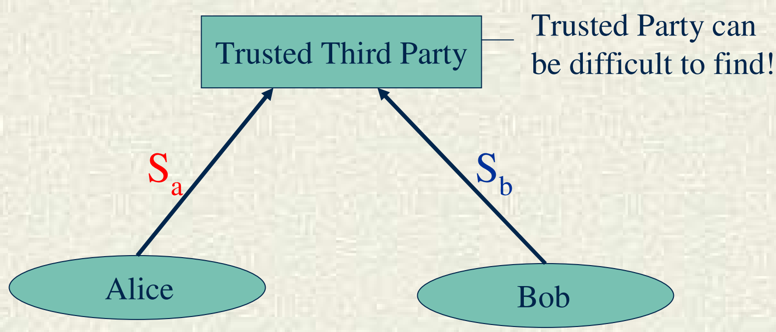

Data Partitions
Data Mining
Data Mining
Private Data
Private Data
Private Data
Private Data
Horizontally partitioned data
Vertically partitioned data
This paper: vertically partitioned data.
1/3/2003
4

Problem Definition

- Alice has a private data set S_a,
- Bob has a private data set S_b,
- They want to build a decision tree classifier on $[S_a \bowtie S_b]$ (Vertically Partitioned),
- Without disclosing their private data.

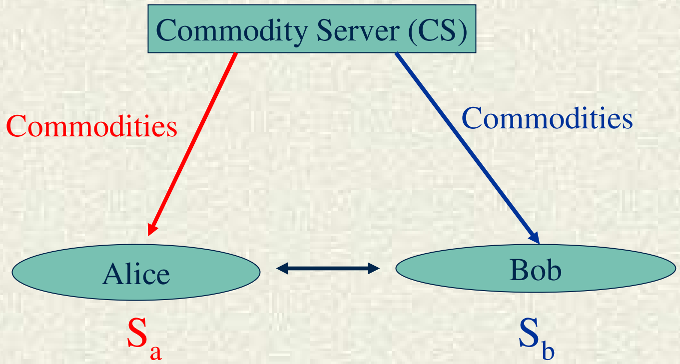
1/3/2003 5

Trusted Third Party Model



1/3/2003 6

Commodity Server (CS) Model



Our solution is based on commodity server model

1/3/2003 7

CS Properties

- Assumption: CS cannot collude with either Alice or Bob.
- CS is <u>not a trusted</u> party.
- CS doesn't participate in the computation.
- CS does not receive private data from Alice or Bob.
- The commodities from CS are independent from Alice and Bob's private data, so they can be generated offline (namely, CS can sell random data for profit ☺).

1/3/2003 8

Security Assumption

- In the Trusted 3rd Party model, Alice and Bob need to assume that the 3rd party does not abuse their private data.
- In the CS model, Alice and Bob need to assume that the CS does not collude with the other party. This security assumption is weaker than the Trusted 3rd Party model, so CS model is more realistic.

1/3/2003 9

The Origin of The CS Model.

- Commodity Server model was proposed by Beaver (1997).
- CS has been used for solving the Private Information Retrieval problem.

1/3/2003 10

Building Block: Scalar Product

- Alice has a private vector A
- Bob has a private vector B
- They want to compute the scalar product of these two vectors

$$A \cdot B = \Sigma\ A(i)*B(i)$$

- Nobody wants to disclose its private data to the other party.

1/3/2003 11

Scalar Product Protocol using Commodity Server

CS

CS → Alice: R_a, r_a

CS → Bob: R_b, r_b

$r_a + r_b = R_a \cdot R_b$

Alice Bob

1. Alice → Bob: $A' = A + R_a$
2. Bob → Alice: $B' = B + R_b$
3. Bob → Alice: $A' \cdot B + r_b$

R_a, R_b: randomly generated vectors.

r_a, r_b: random numbers

Alice computes:

$(A' \cdot B + r_b) - (R_a \cdot B') + r_a$

$= A \cdot B + R_a \cdot B + r_b - R_a \cdot B - R_a \cdot R_b + r_a$

$= A \cdot B$

1/3/2003 12

Performance

- Efficiency
  - Communication cost: 2n
  - Computation cost: 2n
  - Very close to the optimal solution (the one without worrying about the security concerns)
    - Optimal:
      - communication cost: n
      - computation cost: n

1/3/2003 13

Comparing with Existing Work

- Scalar Product Protocols based on 2-party model were proposed by:
  - Du and Atallah (2001)
  - Vaidya and Clifton (2002)
- Advantage: our scheme is more efficient (close to optimal solutions)
- Disadvantage: the security assumption (about collusion) is stronger than the 2-party model

1/3/2003 14

Decision Tree Building

- Procedure:
  - Evaluate splits for each attribute,
  - Select the best split,
  - Create partitions using the best split.
- How to find the best split?

1/3/2003 15

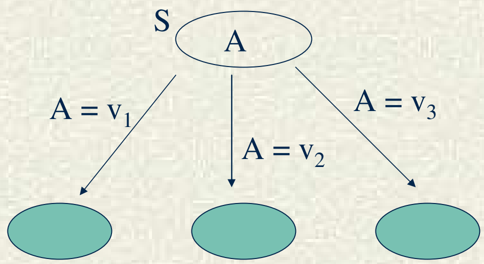
How to Find The Best Split?

- Splitting Index
  - Evaluate the “goodness” of a split
  - Several splitting schemes have been proposed:
    - Entropy
    - Gini Index

1/3/2003 16

Entropy-based Splitting

- We want to measure how good it is to use attribute A to partition S:



1/3/2003 17

Entropy-based Splitting

- Notation:
  - P_j : frequency of class j in S
  - S_v : subset of S for which attribute A = v
- Information Gain:

$$Entropy\ (S) = -\sum_{j=1}^{m} P_j \log P_j$$

$$Gain(S, A) = Entropy(S) - \sum_{v \in A} \left(\frac{|S_v|}{|S|} * Entropy(S_v)\right)$$

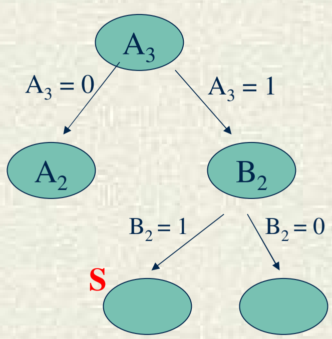
1/3/2003 18

Selecting the Splitting Attribute

- Compute *Gain(S, A)* for each attribute A.
- Select the attribute with the largest information gain as the best split for the current node.
- Partition S using the selected attribute.

1/3/2003 19

Challenges in Our Problem



- S is unknown
  - S contains data that satisfy (A_3=1) and (B_2=1)
  - A_3 is known to Alice only
  - B_2 is known to Bob only
- How to compute P_j, Entropy(S), Entropy(S_v), |S| and |S_v| without knowing what S contains?

1/3/2003 20

Overview of Our Solution (1)

- Assume records in S must satisfy requirements R.
- Divide R into 2 parts: R = (R_a and R_b)
  - R_a contains Alice's attributes
  - R_b contains Bob's attributes
- Alice computes vector $\mathbf{V_a}$:
  - $\mathbf{V_a}(i) = 1$ if the i^{th} record satisfies R_a; $\mathbf{V_a}(i)=0$ otherwise
- Bob computes vector $\mathbf{V_b}$:
  - $\mathbf{V_b}(i) = 1$ if the i^{th} record satisfies R_b; $\mathbf{V_b}(i)=0$ otherwise
- $|S| = V_a \cdot V_b$: use our scalar product protocol!

1/3/2003 21

Overview of Our Solution (2)

- Similarly we can compute the following using scalar product protocol (assume the candidate attribute A belongs to Alice):
  - $|S_v|$: R = (**R_a and A=v**) and **R_b**
  - P_j : R = (**R_a and Class=j**) and **R_b**

    or R = R_a and (**R_b and Class=j**)
- Finally we can compute *Entropy(S)*, *Entropy(S_v)*, and *Gain(S, A)*.

1/3/2003 22

Example

Alice

Day	Outlook	Play Ball
D1	Sunny	No
D2	Sunny	No
D3	Rain	Yes
D4	Rain	Yes
D5	Rain	No

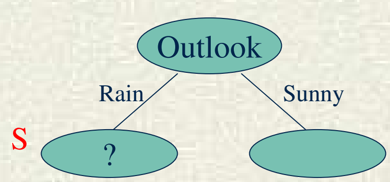
Bob

Day	Humidity	Wind	Play Ball
D1	High	Weak	No
D2	High	Strong	No
D3	High	Weak	Yes
D4	Normal	Weak	Yes
D5	Normal	Strong	No

1/3/2003 23

How to Find the Best Split

- Assuming we already find the the root, which is attribute Outlook.
- We will show how to obtain the best split for the left child of the attribute Outlook.
- We use ***Gain(S, Humidity)*** as an example.



1/3/2003 24

Alice's Vectors

- Alice computes:
  - V_a(outlook=*Rain*) = (0,0,1,1,1)
    - 1 means (outlook=*Rain*).
  - V_a(outlook=*Rain*, playball=*No*) = (0,0,0,0,1)
    - 1 means (outlook=*Rain*) and (playball=*No*).
  - V_a(outlook=*Rain*, playball=*Yes*) = (0,0,1,1,0):
    - 1 means (outlook=*Rain*) and (playball=*Yes*).

1/3/2003 25

Bob's Vectors

- Bob computes:
  - V_b(humidity=*High*) = (1,1,1,0,0)
    - 1 means (Humidity=*High*).
  - V_b(humidity=*Normal*) = (0,0,0,1,1)
    - 1 means (Humidity=*Normal*).

1/3/2003 26

Compute Entropy($S_{v=High}$) (Example)

- $|S_{v=High}| = V_a(\text{outlook}=Rain) \cdot V_b(\text{humidity}=High)$
- $P_0 = P(\text{outlook}=Rain, \text{playball}=No, \text{humidity}=High)$
  $= V_a(\text{outlook}=Rain, \text{playball}=No) \cdot V_b(\text{humidity}=High)$
- $P_1 = P(\text{outlook}=Rain, \text{playball}=Yes, \text{humidity}=High)$
  $= V_a(\text{outlook}=Rain, \text{playball}=Yes) \cdot V_b(\text{humidity}=High)$
- $\text{Entropy}(S_{v=High}) = -(p_0/|S_{v=High}|)\log(p_0/|S_{v=High}|) - (p_1/|S_{v=High}|)\log(p_1/|S_{v=High}|)$
- Similarly, we can compute Entropy($S_{v=Normal}$), and finally *Gain(S, Humidity).*

1/3/2003 27

Security Analysis

- Information Disclosure
  - From the algorithm
  - From the results (the final decision tree)
- Information Disclosure From our Algorithm
  - The results of the scalar product can disclose some information
  - Better protection scheme can be used (details in the paper)
- Information Disclosure from the final decision tree
  - How much can be disclosed needs more study

1/3/2003 28

Conclusion

- We presented a solution for building a decision tree classifier on vertically partitioned private data.
- The scalar product protocol with a commodity server is used as a basic tool.
- Future studies will focus on developing more efficient solutions.

1/3/2003 29