Data Analytics Beyond OLAP

Prof. Yanlei Diao
OPERATIONAL DBs

- DB 1
- DB 2
- DB 3

EXTRACT
TRANSFORM
LOAD (ETL)

- METADATA STORE
- DATA WAREHOUSE

SUPPORTS

- OLAP
- DATA MINING
- INTERACTIVE DATA EXPLORATION
Overview of Topics

- **Data Mining in Databases**
  - Association rule mining
  - Interesting visualizations

- **Approximate Query Processing**
  - Online aggregation: group by aggregation, wander join
  - Interactive SQL

- **Interactive Data Exploration**
  - Faceted search
  - Semantic windows
  - Explore by example
1. Association Rule Mining

Fast Algorithms for Mining Association Rules

Rakesh Agrawal and Ramakrishnan Srikant
VLDB '94
Motivation

- **Example Rules:**
  - 98% of customers who purchase tires get automotive services done
  - Customers who buy mustard and ketchup also buy burgers
  - Goal: find these rules from transactional data

- **Rules help with decision making**
  - E.g., store layout, buying patterns, add-on sales
Association Mining

DB of "Basket Data"

<table>
<thead>
<tr>
<th>TID</th>
<th>items</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1 3 4</td>
</tr>
<tr>
<td>200</td>
<td>2 3 5</td>
</tr>
<tr>
<td>300</td>
<td>1 2 3 5</td>
</tr>
<tr>
<td>400</td>
<td>2 5</td>
</tr>
</tbody>
</table>

Association Rules

\{1\} \Rightarrow \{3\}
\{2,3\} \Rightarrow \{5\}
\{2,5\} \Rightarrow \{3\}

\cdots
Associate Rules

DB of "Basket Data"

<table>
<thead>
<tr>
<th>TID</th>
<th>items</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1 3 4</td>
</tr>
<tr>
<td>200</td>
<td>2 3 5</td>
</tr>
<tr>
<td>300</td>
<td>1 2 3 5</td>
</tr>
<tr>
<td>400</td>
<td>2 5</td>
</tr>
</tbody>
</table>

Association Rules

\{1\} \Rightarrow \{3\}
\{2,3\} \Rightarrow \{5\}
\{2,5\} \Rightarrow \{3\}

Association rule: X => Y

- X and Y are disjoint itemsets, called antecedent (LHS) and consequent (RHS)

- **Confidence**: c% of transactions that contain X also contain Y (rule-specific)
- **Support**: s% of all transactions contain both X and Y (relative to all data)

Goal: find all rules that satisfy the confidence and support thresholds.
Support Example

<table>
<thead>
<tr>
<th>TID</th>
<th>Cereal</th>
<th>Beer</th>
<th>Bread</th>
<th>Bananas</th>
<th>Milk</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>4</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>6</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Support(Cereal)  
\( \frac{4}{8} = 0.5 \)

Support(Cereal => Milk)  
\( \frac{3}{8} = 0.375 \)
## Confidence Example

<table>
<thead>
<tr>
<th>TID</th>
<th>Cereal</th>
<th>Beer</th>
<th>Bread</th>
<th>Bananas</th>
<th>Milk</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>4</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>6</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Confidence (Cereal => Milk)  
\[ \frac{3}{4} = 0.75 \]

Confidence (Bananas => Bread)  
\[ \frac{1}{3} = 0.33333... \]
Apriori Algorithm and Notation

- \{i_1, i_2, ..., i_m\} be the set of literals, known as items
- \{T_j\} is the set of transactions (database), where each transaction T_j is a set of items s.t.
  - Each transaction has a unique identifier TID
  - The size of an itemset is the number of items
  - Itemset of size k is a k-itemset
- Assume that items in an itemset are sorted in lexicographical order
Step I: Find all itemsets with minimum support \((\text{min\_sup})s\)

<table>
<thead>
<tr>
<th>TID</th>
<th>items</th>
<th>support</th>
<th>itemsets</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1 3 4</td>
<td>0.25</td>
<td>{4}, {1,2}, {1,4}, {1,5}, {3,4},</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{1,3,4}, {1,2,3}, {1,2,5}, {1,3,5},</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{1,2,3,5}</td>
</tr>
<tr>
<td>200</td>
<td>2 3 5</td>
<td>0.5</td>
<td>{1}, {1,3}, {2,3}, {3,5}, {2,3,5}</td>
</tr>
<tr>
<td>300</td>
<td>1 2 3 5</td>
<td>0.75</td>
<td>{2}, {3}, {5}, {2,5}</td>
</tr>
</tbody>
</table>

Step II: Generate rules from \(\text{min\_sup}'ed\) itemsets

<table>
<thead>
<tr>
<th>support</th>
<th>confidence</th>
<th>rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>66%</td>
<td>{3}=&gt;{1}, {3}=&gt;{2}, {2}=&gt;{3}, {3}=&gt;{5},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{5}=&gt;{3}, {5}=&gt;{2,3}, {3}=&gt;{2,5},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{2}=&gt;{3,5}, {5,2}=&gt;{3}, {5,3}=&gt;{2}</td>
</tr>
<tr>
<td>0.5</td>
<td>100%</td>
<td>{1}=&gt;{3}, {5,3}=&gt;{2}, {2,3}=&gt;{5}</td>
</tr>
<tr>
<td>0.75</td>
<td>100%</td>
<td>{5}=&gt;{2}, {2}=&gt;{5}</td>
</tr>
</tbody>
</table>
Step I: Finding Minsup Itemsets

- What is the complexity of finding all subsets of item that satisfy the mini_sup s?
- The power set of the n literals!

- A new algorithmic framework based on anti-monotonicity:
  - For a frequent itemset, all of its subsets are also frequent
  - For an infrequent itemset, all of its supersets must be infrequent
  - Can be used to design efficient pruning of the search space.
Anti-monotonicity

- Adding items to an itemset never increases its support
- For a frequent itemset, all of its subsets are also frequent
- For an infrequent itemset, all of its supersets must be infrequent
Anti-monotonicity

- Adding items to an itemset never increases its support
- For a frequent itemset, all of its subsets are also frequent
- For an infrequent itemset, all of its supersets must be infrequent
**Anti-monotonicity**

- Adding items to an itemset never increases its support.
- For a frequent itemset, all of its subsets are also frequent.
- For an infrequent itemset, all of its supersets must be infrequent.
Step I: Finding Minsup Itemsets

- **Anti-monotonicity:**
  
  *Adding items to an itemset never increases its support*

- **Apriori Algorithm:** Proceed inductively on itemset size

  1) Base case: Begin with all minsup itemsets of size 1 ($L_1$)
  
  2) Without peeking at the DB, generate candidate itemsets of size $k$ ($C_k$) from $L_{k-1}$
  
  3) Remove candidate itemsets that contain unsupported subsets
  
  4) Further refine $C_k$ using the database to produce $L_k$

repeat
Task 2) Guess Itemsets

• Naïve way:
  – Extend all itemsets with all possible items

• Apriori:
  2) Join $L_{k-1}$ with itself, adding only a single, final item

  • e.g.: $\{1 \ 2 \ 3\}$, $\{1 \ 2 \ 4\}$, $\{1 \ 3 \ 4\}$, $\{1 \ 3 \ 5\}$, $\{2 \ 3 \ 4\}$ produces $\{1 \ 2 \ 3 \ 4\}$ and $\{1 \ 3 \ 4 \ 5\}$
Task 3) Filter Itemsets

- **Apriori:**
  2) Join $L_{k-1}$ with itself, adding only a single, final item
     - e.g.: \{1 \ 2 \ 3\}, \{1 \ 2 \ 4\}, \{1 \ 3 \ 4\}, \{1 \ 3 \ 5\}, \{2, \ 3, \ 4\} produces \{1 \ 2 \ 3 \ 4\} and \{1 \ 3 \ 4 \ 5\}
  3) Remove itemsets with an unsupported subset
     - e.g.: \{1 \ 3 \ 4 \ 5\} has an unsupported subset: \{1 \ 4 \ 5\} if minsup = 50%
Task 4) Finalize k-Itemsets

- **Apriori:**
  2) Join $L_{k-1}$ with itself, adding only a single, final item
     • E.g., \{1 2 3\}, \{1 2 4\}, \{1 3 4\}, \{1 3 5\}, \{2, 3, 4\} produces \{1 2 3 4\} and \{1 3 4 5\}
  3) Remove itemsets with an unsupported subset
     • E.g., \{1 3 4 5\} has an unsupported subset: \{1 4 5\} if minsup = 50%
  4) Use the database to further refine $C_k$
     • Count precisely the occurrence of each itemset in the dataset, to see if it is indeed larger than min_sup
Full Algorithm

Apriori{

L1 = {large 1-itemsets}

for (k = 2; Lk-1 != ∅ ; k++) { //Generate large k-itemsets

  //Step 1: generate candidate k-itemsets from large (k-1)-itemsets
  Ck = apriori-gen(Lk-1);

  //Step 2: count support of each candidate set
  forall transactions t in the database {

    Ct = subset(Ck, t); // Candidates from Ck contained in t
    forall candidates c in Ct

    c.count++;

  }

  Lk = {c in Ck | c.count >= minsup}

}

Answer = ∪k Lk

}
apriori-gen(Lk-1) {

    // Intuition: every subset of a large itemset must be large.  
    // So combine almost-matching pairs of large (k-1)-itemsets,  
    // and prune out those with non-large (k-1)-subsets.

    join:

    insert into Ck  
    select p.item1, ..., p.itemk-1, q.itemk-1  
    from Lk-1 p, Lk-1 q  
    where p.item1 = q.item1 and ... and p.itemk-2 = q.itemk-2 and p.itemk-1 < q.itemk-1;

    prune:

    // delete itemsets such that some (k-1)-subset is not in Lk-1  
    forall itemsets c in Ck  

            forall (k-1)-subsets s of c  

            if (s not in Lk-1) {  

                delete c from Ck;  
                break;

            }

    }
}
2. Interesting Visualizations

**SeeDB**: efficient data-driven visualization recommendations to support visual analytics

Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya Parameswaran, Neoklis Polyzotis
VLDB ’14
Visualization Recommendation

Given a dataset and a task, automatically produce a set of visualizations that are the most “interesting” given the task.
For simplicity, assume a single table (star schema)

Visualizations = agg. + grp. by queries

\[ V_i = \text{SELECT } d, f(m) \]
FROM table
WHERE ___
GROUP BY d

(d, m, f): dimension, measure, aggregate
Space of Visualizations

\[ V_i = \text{SELECT } d, f(m) \]
FROM table
WHERE __
GROUP BY d

(d, m, f):
dimension, measure, aggregate

\{d\} : race, work-type, sex etc.
\{m\} : capital-gain, capital-loss, hours-per-week
\{f\} : COUNT, SUM, AVG
Key Questions

I. *Interestingness*: How do we determine if a visualization is interesting?
   - Utility Metric

II. *Scale*: How to make recommendations efficiently and interactively?
   - Optimizations
Deviation-based Utility Metric

A visualization is interesting if it displays

* a large deviation from some reference

**Task:** compare unmarried adults with all adults

\[ V_1 = \text{SELECT } d, f(m) \text{ FROM table WHERE target GROUP BY } d \]
\[ V_2 = \text{SELECT } d, f(m) \text{ FROM table WHERE reference GROUP BY } d \]

Compare induced probability distributions!
Deviation-based Utility Metric

A visualization is interesting if it displays a large deviation from some reference

Many metrics for computing distance between distributions

\[ D [P(V1), P(V2)] \]

- Earth mover’s distance
- L1, L2 distance
- K-L divergence

Any distance metric b/n distributions is OK!
Overview of Topics

- **Data Mining over Databases**
  - Association rule mining
  - Interesting visualizations

- **Approximate Query Processing**
  - Online aggregation: group by aggregation, wander join
  - Interactive SQL

- **Interactive Data Exploration**
  - Faceted search
  - Semantic windows
  - Explore by example
3. Interactive Data Exploration

**Explore-by-example**: an automatic query steering framework for interactive data exploration

Kyriaki Dimitriadou, Olga Papaemmanouil, Yanlei Diao
SIGMOD ’14
Web Applications: e.g., House Hunting

- 2+ bedrooms, ≤ 0.5 million
- 2+ bedrooms, price depending on the size
- 2+ bedrooms, price depending on the size, location
- 2+ bedrooms, price depending on the size, location, quietness (away from street, higher floor)
- 2+ bedrooms, price depending on the size, location, quietness, school district, building structure...
An Explore-by-Example Framework

- Sample Acquisition
- Exploration Queries
- Classification Model
- User Model
- Query Formulation
- Final Data Extraction Query

Relevance Feedback:
- Relevant examples
- Irrelevant examples

Initial $d$ attributes

Sample
How to produce initial examples that fall in the user interest area?

Analyze sampling methods in this new problem setting:

- Random sampling
- Equi-width stratified sampling
- Equi-depth stratified sampling
- Progressive sampling [Dimitriadou 2014]
Equi-Width Stratified Sampling

Given a database of $N$ data points in $D$-dim space, a true user interest in a $d^*$-dim subspace

- Start with a user-chosen $d$-dim subspace, $d^* \leq d << D$
- Consider generating $k$ samples
- Want to create $c$ buckets per dimension, $c^d \geq k$
- Generate 1 sample per cell, $[l^1, h^1] \times [l^2, h^2] \times \ldots \times [l^c, h^c]$

**Equi-width stratified sampling**

- $c$ buckets are of equal width
Equi-Depth Stratified Sampling

- Given a database of \( N \) data points in \( D \)-dim space, a true user interest in a \( d^* \)-dim subspace
  - Start with a user-chosen \( d \)-dim subspace, \( d^* \leq d \ll D \)
  - Consider generating \( k \) samples
  - Want to create \( c \) buckets per dimension, \( c^d \geq k \)
  - Generate 1 sample per cell, \([ l^1, h^1 ] \times [ l^2, h^2 ] \ldots \times [ l^c, h^c ]\)

Equi-depth stratified sampling
- \( c \) buckets have (roughly) equal numbers of data points
Progressive Sampling: dynamic # examples
Classification based on Decision Trees

<table>
<thead>
<tr>
<th>Sample</th>
<th>Red</th>
<th>Green</th>
<th>User Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object A</td>
<td>13.67</td>
<td>12.34</td>
<td>Yes</td>
</tr>
<tr>
<td>Object B</td>
<td>15.32</td>
<td>14.50</td>
<td>No</td>
</tr>
<tr>
<td>..</td>
<td>..</td>
<td>..</td>
<td>..</td>
</tr>
<tr>
<td>Object X</td>
<td>14.21</td>
<td>13.57</td>
<td>Yes</td>
</tr>
</tbody>
</table>

SELECT * FROM galaxy WHERE red <= 14.82 AND red >= 13.5 AND green <= 13.74
(2) Recover from Misclassified Samples

- False negative
- False positive
- Predicted Area
Handling Misclassified Samples

Red wavelength

Green Wavelength

Sampling Areas
Clustering-based Sampling

Red wavelength

Clustering-Sampling Areas

Green Wavelength
(3) Refine Boundaries of Relevant Areas

Red wavelength

Sampling Areas

Green Wavelength