Association Rules: Past, Present & Future

Ramakrishnan Srikant

www.almaden.ibm.com/cs/people/srikant/
Talk Outline

- Association Rules
  - Motivation & Definition
  - Most Popular Computation Approach
- Other Computation Approaches
- Extensions
- Interest Measures
- Future Directions
Motivation

- Many organizations have amassed massive data (running into several gigabytes and more)
  - Retailing: Sears, Safeway, K-Mart, Proctor & Gamble
  - Finance: American Express, Citicorp
  - Insurance: Prudential
  - Transportation: United Airlines
  - Hospitals

- Potential goldmine of valuable business information
Association Rules

• Given:
  – a database of transactions
  – each transaction is a set of items

• Example: 30% of transactions that contain beer also contain diapers; 5% of transactions contain these items
  – 30% : confidence of the rule
  – 5% : support of the rule

• Find all association rules that satisfy user-specified minimum support and minimum confidence constraints.

• We are interested in finding all rules rather than verifying if a rule holds.
Association Rules (cont.)

- Problem introduced in SIGMOD '93 paper, “Mining association rules between sets of items in large databases” by R. Agrawal, T. Imielinski, and A. Swami.

- Search on “association rules”:
  - 3369 papers in Citeseer.
  - 29,300 hits in Google.
Application Examples

- Market Basket Analysis
  - “* ⇒ Maintenance Agreement”
    What the store should do to boost Maintenance Agreement sales?
  - “Home Electronics ⇒ *”
    What other products should the store stock up on if the store has a sale on Home Electronics?

- HIC Australia “success story” (Nearhos et al., VLDB ’96)
  - associations between medical payment codes
  - saved $500,000 per year per state

- Reducing telecommunications order failures.
  - set of orders (transactions)
  - each order has around 3.5 sub-parts (USOCs), plus failure code (RMA) if automated processing failed
  - find sets of USOCs which lead to failure
  - only 2.5% of orders fail, 25 different failure codes
Problem Decomposition

1. Find all sets of items that have minimum support (frequent itemsets).

2. Use the frequent itemsets to generate the desired rules.
Problem Decomposition – Example

<table>
<thead>
<tr>
<th>Transaction ID</th>
<th>Items Bought</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Shoes, Shirt, Jacket</td>
</tr>
<tr>
<td>2</td>
<td>Shoes, Jacket</td>
</tr>
<tr>
<td>3</td>
<td>Shoes, Jeans</td>
</tr>
<tr>
<td>4</td>
<td>Shirt, Sweatshirt</td>
</tr>
</tbody>
</table>

For Minimum Support = 50% = 2 transactions, and Minimum Confidence = 50%:

<table>
<thead>
<tr>
<th>Frequent Itemset</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>{Shoes}</td>
<td>75%</td>
</tr>
<tr>
<td>{Shirt}</td>
<td>50%</td>
</tr>
<tr>
<td>{Jacket}</td>
<td>50%</td>
</tr>
<tr>
<td>{Shoes, Jacket}</td>
<td>50%</td>
</tr>
</tbody>
</table>

For the rule Shoes ⇒ Jacket:

- Support = Support({Shoes, Jacket}) = 50%
- Confidence = \( \frac{\text{Support}({\{\text{Shoes, Jacket}\}})}{\text{Support}({\{\text{Shoes}\}})} = \frac{50}{75} = 66.6\% \).

Jacket ⇒ Shoes has 50% support and 100% confidence.
Problem Statement

- $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$ : a set of literals, called items.
- Transaction $T$ : a set of items such that $T \subseteq \mathcal{I}$.
- Database $\mathcal{D}$ : a set of transactions.
- A transaction $T$ contains $X$, a set of some items in $\mathcal{I}$, if $X \subseteq T$.
- An association rule is an implication of the form $X \Rightarrow Y$, where $X, Y \subseteq \mathcal{I}$
- The rule $X \Rightarrow Y$ holds in the transaction set $\mathcal{D}$ with confidence $c$ if $c\%$ of transactions in $\mathcal{D}$ that contain $X$ also contain $Y$.
- The rule $X \Rightarrow Y$ has support $s$ in the transaction set $\mathcal{D}$ if $s\%$ of transactions in $\mathcal{D}$ contain $X \cup Y$.

Find all rules that have support and confidence greater than user-specified minimum support and minimum confidence.
The Apriori Algorithm

- $L_k$: Set of frequent itemsets of size $k$ (those with minimum support).
- $C_k$: Set of candidate itemsets of size $k$ (potentially frequent itemsets)

$L_1 = \{\text{frequent items}\};$

\begin{verbatim}
for ( $k = 1; L_k \neq \emptyset; k++$ ) do
    begin
        $C_{k+1} = \text{New candidates generated from } L_k;$
        foreach transaction $t$ in the database do
            Increment the count of all candidates in $C_{k+1}$ that are contained in $t$.
        $L_{k+1} = \text{Candidates in } C_{k+1} \text{ with minimum support.}$
    end
end
Answer = $\bigcup_k L_k$;
\end{verbatim}

**Apriori – Example**

**Dataset \( \mathcal{D} \)**

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1 3 4</td>
</tr>
<tr>
<td>20</td>
<td>2 3 5</td>
</tr>
<tr>
<td>30</td>
<td>1 2 3 5</td>
</tr>
<tr>
<td>40</td>
<td>2 5</td>
</tr>
</tbody>
</table>

Minimum Support = 50% = 2 trans.

- **Scan** \( \mathcal{D} \)
  
  \( \rightarrow \)

**C\(_1\)**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {1} )</td>
<td>2</td>
</tr>
<tr>
<td>( {2} )</td>
<td>3</td>
</tr>
<tr>
<td>( {3} )</td>
<td>3</td>
</tr>
<tr>
<td>( {4} )</td>
<td>1</td>
</tr>
<tr>
<td>( {5} )</td>
<td>3</td>
</tr>
</tbody>
</table>

**L\(_1\)**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {1} )</td>
<td>2</td>
</tr>
<tr>
<td>( {2} )</td>
<td>3</td>
</tr>
<tr>
<td>( {3} )</td>
<td>3</td>
</tr>
<tr>
<td>( {5} )</td>
<td>3</td>
</tr>
</tbody>
</table>

**Scan** \( \mathcal{D} \)

\( \rightarrow \)

**C\(_2\)**

<table>
<thead>
<tr>
<th>Itemset</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {1 \ 2} )</td>
</tr>
<tr>
<td>( {1 \ 3} )</td>
</tr>
<tr>
<td>( {1 \ 5} )</td>
</tr>
<tr>
<td>( {2 \ 3} )</td>
</tr>
<tr>
<td>( {2 \ 5} )</td>
</tr>
<tr>
<td>( {3 \ 5} )</td>
</tr>
</tbody>
</table>

**Scan** \( \mathcal{D} \)

\( \rightarrow \)

**C\(_2\)**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {1 \ 2} )</td>
<td>1</td>
</tr>
<tr>
<td>( {1 \ 3} )</td>
<td>2</td>
</tr>
<tr>
<td>( {1 \ 5} )</td>
<td>1</td>
</tr>
<tr>
<td>( {2 \ 3} )</td>
<td>2</td>
</tr>
<tr>
<td>( {2 \ 5} )</td>
<td>3</td>
</tr>
<tr>
<td>( {3 \ 5} )</td>
<td>2</td>
</tr>
</tbody>
</table>

**L\(_2\)**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {1 \ 3} )</td>
<td>2</td>
</tr>
<tr>
<td>( {2 \ 3} )</td>
<td>2</td>
</tr>
<tr>
<td>( {2 \ 5} )</td>
<td>3</td>
</tr>
<tr>
<td>( {3 \ 5} )</td>
<td>2</td>
</tr>
</tbody>
</table>

**Scan** \( \mathcal{D} \)

\( \rightarrow \)

**C\(_3\)**

<table>
<thead>
<tr>
<th>Itemset</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {2 \ 3 \ 5} )</td>
</tr>
</tbody>
</table>

**Scan** \( \mathcal{D} \)

\( \rightarrow \)

**C\(_3\)**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {2 \ 3 \ 5} )</td>
<td>2</td>
</tr>
</tbody>
</table>

**L\(_3\)**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {2 \ 3 \ 5} )</td>
<td>2</td>
</tr>
</tbody>
</table>

R. Srikant
Apriori Candidate Generation

Monotonicity Property: All subsets of a frequent itemset are frequent.

Given $L_k$, generate $C_{k+1}$ in two steps:

$$L_3 = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 3, 5\}, \{2, 3, 4\}\}$$

1. Join Step: Join $L_k$ with $L_k$, with the join condition that the first $k - 1$ items should be the same and $l^1[k] < l^2[k]$.

   Now, $C_4 = \{\{1, 2, 3, 4\}, \{1, 3, 4, 5\}\}$.

2. Prune Step: Delete all candidates which have a non-frequent subset.

   Now, $C_4 = \{\{1, 2, 3, 4\}\}$. 

R. Srikant
On-The-Fly vs. Apriori Candidate Generation

\[ L_3 = \{ \{1 2 3\}, \{1 2 4\}, \{1 3 4\}, \{1 3 5\}, \{2 3 4\} \} \]

Transaction \( T : \{1 \ 2 \ 3 \ 4 \ 5\} \)

\( \{1 \ 2 \ 3\} \) contained in \( T \).

\( \Rightarrow \) Generate \( \{1 \ 2 \ 3 \ 4\} \) and \( \{1 \ 2 \ 3 \ 5\} \).

Similarly, generate \( \{1 \ 2 \ 4 \ 5\}, \{1 \ 3 \ 4 \ 5\} \) and \( \{2 \ 3 \ 4 \ 5\} \).
Finding Candidates Contained in a Transaction

Given

- a transaction \( T \) and
- a set of candidates \( C_k \),

find all members of \( C_k \) which are contained in \( T \).

\( C_2 : \{ \{1, 2\}, \{1, 3\}, \{3, 4\}, \{5, 6\} \} \)

\( T : \{3, 4, 6\} \)

Only check itemsets in buckets corresponding to 3, 4, and 6, i.e., \( \{3,4\} \).

- avg. number of items in trans. \( \ll \) total number of items
- generalized into a hash-tree
**Scaleup**

**Number of transactions scale-up:**

- 0.75% support
- similar results at other levels

**Transaction size scale-up:**

- Database size kept constant
Apriori vs. Earlier Algorithms

- Apriori 3 to 10 times faster than AIS.
- Apriori 3 to more than 100 times faster than SETM.
- Performance gap increases with problem size.
- Apriori scales linearly with the number of transactions.
- Apriori also has excellent scale-up properties with respect to the transaction size and the number of items in the database.
Talk Outline

• Association Rules

• Other Computation Approaches
  – Sampling
  – Transaction IDs
  – Maximal Associations
  – Data Projection
  – Constraints

• Extensions of the Concept

• Interest Measures

• Future Directions
Sampling

- Use sampling to reduce the number of passes

- Approach:
  1. Run algorithm on a sample \( D_S \) of the dataset \( D \).
  2. Find support in \( D \) for
     - “Expected Frequent”: all itemsets that were frequent in \( D_S \), plus
     - “Negative Border”: those itemsets that were not frequent in \( D_S \), but all of whose subsets were frequent in \( D_S \).
  3. If any of the itemsets in the negative border are frequent in \( D \), need mop-up pass for potentially frequent extensions of those itemsets.

- Typically run on \( D_S \) at a lower support to minimize size and number of mop-up passes.
Sampling (cont.)

- Pro: reduces the number of passes.
- Con: have to count significantly (or substantially) more candidates.

Transaction IDs

- For each itemset, keep list of transaction ids that support the itemset.

- Find support of “abc” by merging lists of “ab” and “ac”.

- Generate itemsets in depth-first manner (a, ab, ac, abc, ...) to minimize disk I/O.

- Pro: substantially faster for longer associations

- Con: substantially slower for shorter associations

Maximal Associations

- R.J. Bayardo, “Efficiently Mining Long Patterns from Databases”, SIGMOD ’98.
  - Finds only the maximal patterns.
  - Scales roughly linearly in the number of maximal patterns.

  - Fixed RHS
  - Finds all rules whose confidence is significantly higher than any of their simplifications.
  - Prune based on confidence.
Data Projection

- Generate associations in depth-first manner.
- Project database down the tree.
- Example: To count all itemsets starting with \( \{a, b\} \), with possible extensions \( \{d, f, g\} \):
  - Select transactions that contain \( a \) and \( b \).
  - Project only \( d, f \) and \( g \) from these transactions.
- Pro: substantially faster for longer associations.
- Cons: database cannot be much larger than memory, not much speedup if most patterns are short.
- J. Han, J. Pei and Y. Yin, “Mining Frequent Patterns without Candidate Generation”, SIGMOD 2000.
Item Constraints

• Users are often interested in a subset of rules.

• Can express constraints as boolean expressions over (the presence of) items.
  – (Shirts AND Shoes) OR (Outerwear AND NOT Hiking Boots)

**Approach Overview**

1. Find $L$, the set of all *frequent itemsets* (those with minimum support) satisfying the constraint $B$.

2. Count the support of subsets of the frequent itemsets in $L$.

3. Generate rules from the frequent itemsets in $L$.
   - $\text{support}(AB \Rightarrow CD) = \text{support}(ABCD)$
   - $\text{confidence}(AB \Rightarrow CD) = \frac{\text{support}(ABCD)}{\text{support}(AB)}$
Can we push the constraint?

For any frequent itemset with $k$ items that satisfies $\mathcal{B}$

- there is a subset with $k-1$ items that satisfies $\mathcal{B}$, unless
- the itemset corresponds to a disjunct in $\mathcal{B}$ with exactly $k$ non-negated terms.

Example:

- let $\mathcal{B} = (1 \land 2) \lor (4 \land \neg 5)$
- $\{1 \ 2 \ 4\}$ has a subset $\{1 \ 2\}$ that satisfies $\mathcal{B}$
- $\{1 \ 2\}$ does not have any subset that satisfies $\mathcal{B}$, but corresponds to the disjunct “1 $\land$ 2”
Constraint Transformation

1. Generate a set of selected items $S$ such that any itemset that satisfies the constraint $B$ will contain at least one selected item.

2. Generate only candidates that contain selected items.

3. Discard frequent itemsets that do not satisfy $B$.

Example:

- let the set of items be $\{1, 2, 3, 4, 5\}$
- if $B = (1 \land 2) \lor 3$
  - $S$ could be $\{1, 3\}$, $\{2, 3\}$ or $\{1, 2, 3, 4, 5\}$
- if $B = (1 \land 2) \lor \neg 3$
  - $S$ could be $\{1, 2, 4, 5\}$
Choosing Selected Items

- Assume that $B$ is in DNF (without loss of generality)
  - $B$: $D_1 \lor D_2 \lor \ldots \lor D_m$
  - $D_i$: $\alpha_{i1} \land \alpha_{i2} \land \ldots \land \alpha_{in_i}$
  - $\alpha_{ij}$: either $\langle \text{item} \rangle$ or NOT $\langle \text{item} \rangle$

- Choose one element from each disjunct in $B$
  - if $\langle \text{item} \rangle$, add the item to $S$.
  - if NOT $\langle \text{item} \rangle$, add all the other items (in the dataset) to $S$.

- Heuristic: minimize the sum of the supports of the elements in $S$. 

What Constraints can be Pushed?

- J. Pei and J. Han, “Can We Push More Constraints into Frequent Pattern Mining?”, KDD 2000.

- Classify constraints into succinct, anti-monotone, monotone, convertible and inconvertible.

- The first four classes can be pushed (to varying degrees).

Talk Outline

- Association Rules
- Other Computation Approaches
- Extensions
  - Taxonomies
  - Quantitative Association Rules
  - Sequential Patterns
- Interest Measures
- Future Directions
Generalized Association Rules

- Given a taxonomy $\mathcal{T}$:

\[
\begin{align*}
\text{Clothes} & \quad \text{Outerwear} & \quad \text{Shirts} & \quad \text{Footwear} \\
\quad & \quad \text{Jackets} & \quad \text{Ski Pants} & \quad \text{Shoes} & \quad \text{Hiking Boots}
\end{align*}
\]

- Find associations between items at any level of the taxonomy

- A transaction $\{\text{Jacket, Shoes}\}$ supports the rules
  - $\text{Jacket} \Rightarrow \text{Shoes}$,
  - $\text{Outerwear} \Rightarrow \text{Footwear}$,
  - $\text{Clothes} \Rightarrow \text{Shoes}$, etc.

Motivation:

- Rules at lower levels may not have minimum support.
- Replace many specialized rules with one general rule.
- Use taxonomy information to identify interesting rules.
Generalized Association Rules (cont.)

```
  Clothes
     /\    /
    /   \  /   \
  Outerwear Shirts Footwear
    /\   /\   /\ \\
   Jackets Ski Pants Shoes Hiking Boots
```

- “Outerwear $\Rightarrow$ Hiking Boots” may be a valid rule, even if
  - “Jackets $\Rightarrow$ Hiking Boots” doesn’t have min. support.
  - “Clothes $\Rightarrow$ Hiking Boots” doesn’t have min. confidence.

- $\text{support(“Outerwear } \Rightarrow \text{ Hiking Boots”)}$ is not equal to $\text{support(“Jackets } \Rightarrow \text{ Hiking Boots”) + support(“Ski Pants } \Rightarrow \text{ Hiking Boots“).}$
Approach

- Add all ancestors of each item in a transaction to the transaction.
  - *Example*: A transaction \{Jackets, Shoes\} is replaced with \{Jackets, Outerwear, Clothes, Shoes, Footwear\}.
- Run the algorithm for mining association rules over these “extended transactions”.

```
 Clothes
  /   \
Outerwear  Shirts  Footwear
    /   \
Jackets  Ski Pants  Shoes  Hiking Boots
```
Enhancements

- Pre-compute ancestors.
- Only add relevant ancestors to transactions.
  - If “Jacket” is present in a transaction $T$, but “Clothes” is not in any of the candidates, don’t add “Clothes” to $T$.
  - Combine with pre-computing ancestors.
- Prune candidate itemsets that contain an item and its ancestor.
  - $\text{support}([\text{Jacket, Clothes}]) = \text{support}([\text{Jacket}])$.
  - Pruning $C_2$ is sufficient.

![Diagram of Clothing Items]

- J. Han and Y. Fu, “Discovery of Multiple-Level Association Rules from Large Databases”, VLDB ’95.
Talk Outline

• Association Rules

• Other Computation Approaches

• Extensions
  – Taxonomies
  – Quantitative Association Rules
  – Sequential Patterns

• Interest Measures

• Future Directions
Quantitative Associations

- Given a relational table:
  - a set of records
  - each record has categorical & quantitative attributes

- Example: “30% of married people between age 45 and 60 have at least 2 cars; 5% of records have these properties”

- Find all association rules that satisfy user-specified minimum support and minimum confidence constraints.

- Can we map this problem to boolean associations?
### Mapping to Boolean Associations

<table>
<thead>
<tr>
<th>RecID</th>
<th>Age</th>
<th>Married</th>
<th>#Cars</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>23</td>
<td>No</td>
<td>1</td>
</tr>
<tr>
<td>200</td>
<td>25</td>
<td>Yes</td>
<td>1</td>
</tr>
<tr>
<td>300</td>
<td>29</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>400</td>
<td>34</td>
<td>Yes</td>
<td>2</td>
</tr>
<tr>
<td>500</td>
<td>38</td>
<td>Yes</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RecID</th>
<th>Age: 20..29</th>
<th>Age: 30..39</th>
<th>Married: Yes</th>
<th>Married: No</th>
<th>#Cars: 0</th>
<th>#Cars: 1</th>
<th>#Cars: 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>200</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>300</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>400</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>500</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Rule:** \( \langle \text{Age: 30..39} \rangle \text{ and } \langle \text{Married: Yes} \rangle \Rightarrow \langle \text{NumCars: 2} \rangle \)

(40% support, 100% confidence)
Mapping Woes

- “MinSup”: small intervals ⇒ miss rules because of low support
- “MinConf”: large intervals ⇒ miss rules because of low confidence

<table>
<thead>
<tr>
<th>RecID</th>
<th>Age</th>
<th>Married</th>
<th>NumCars</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>23</td>
<td>No</td>
<td>1</td>
</tr>
<tr>
<td>200</td>
<td>25</td>
<td>Yes</td>
<td>1</td>
</tr>
<tr>
<td>300</td>
<td>29</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>400</td>
<td>34</td>
<td>Yes</td>
<td>2</td>
</tr>
<tr>
<td>500</td>
<td>38</td>
<td>Yes</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rule</th>
<th>Support</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{NumCars: 0} ) (\Rightarrow) (\text{Married: No})</td>
<td>20%</td>
<td>100%</td>
</tr>
<tr>
<td>(\text{NumCars: 0..1} ) (\Rightarrow) (\text{Married: No})</td>
<td>40%</td>
<td>66%</td>
</tr>
</tbody>
</table>
Mapping Woes: Solution

Solution:

- use small intervals, and
- combine adjacent intervals.

But...

- execution time high
- many similar rules

Note: Not meaningful to combine categorical attribute values unless a taxonomy is present.
Approach

- How do we reduce execution time?
  - *maxsupport* limit for combining adjacent intervals.

- Should we partition a quantitative attribute?
  If so, how many partitions?
  - Partial completeness measure.

- How do we deal with similar rules?
  - “Greater-than-expected-value” interest measure.

- How do we compute the rules?
  - Extend algorithm for boolean associations.

- R. Srikant and R. Agrawal, “Mining Quantitative Association Rules in Large Relational Tables”, SIGMOD ’96

Talk Outline

- Association Rules
- Other Computation Approaches
- Extensions
  - Taxonomies
  - Quantitative Association Rules
  - Sequential Patterns
- Interest Measures
- Future Directions
Sequential Patterns

- Given:
  - A database of customer transactions
  - Each transaction is a set of items

- Example: 10% of customers bought “shirts” and “jackets” in one transaction, followed by “shoes” in another transaction.
  - 10% is called the support of the pattern

- Find all sequential patterns supported by more than a user-specified percentage of customers.

- Constraints
  - max/min time gap between elements
  - “sliding window” transactions

- Applications:
  - attached mailing
  - customer satisfaction
  - medical research

Talk Outline

- Association Rules
- Other Computation Approaches
- Extensions
- Interest Measures
- Future Directions
“Simple” Interest Measures

- Statistical Measures, e.g., p-value of independence test.
- “Lift”: ratio of support to expected support assuming independence.
- These measures are not based on what the other rules are.
Interest Measures based on Other Rules

- Clothes $\Rightarrow$ Shoes
  - 8% support, 70% confidence
- Quarter of sales of Clothes are Jackets
- Jackets $\Rightarrow$ Shoes
  - expect 2% support, 70% confidence
- Interesting rule if support/confidence is greater from “expected” value.
- User-specified “interest level”
Interest Measures based on Other Rules (cont.)

- Jackets $\Rightarrow$ Shoes [2% support, 70% confidence]
- Jackets and Shirts $\Rightarrow$ Shoes [1.5% support, 71% confidence]
- Second rule not very useful.
- Can take one step further & consider only the direction of the correlation (positive, negative or neutral).
- Extension of a rule is interesting only if its direction is different.
Future Directions

- Faster computation (still an active area!)
- What are the interesting rules?
- Privacy
  - Can we find association rules while preserving privacy at the individual transaction level?