Data Mining Techniques
CS 6220 - Section 3 - Fall 2016

Lecture 15: Association Rules

Jan-Willem van de Meent
(credit: Yijun Zhao, Tan et al., Leskovec et al.)
Association Rule Discovery

Market-basket model:

- **Goal**: Identify items that are bought together by sufficiently many customers

- **Approach**: Process the sales data to find dependencies among items

- **A classic rule**:
  - If someone buys diaper and milk, then he/she is likely to buy beer
  - Don’t be surprised if you find six-packs next to diapers!
The Market-Basket Model

- A large set of **items**
  - e.g., things sold in a supermarket
- A large set of **baskets**
- Each basket is a small subset of items
  - e.g., the things one customer buys on one day
- Want to discover association rules
  - People who bought \{x,y,z\} tend to buy \{v,w\}

**Input:**

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bread, Coke, Milk</td>
</tr>
<tr>
<td>2</td>
<td>Beer, Bread</td>
</tr>
<tr>
<td>3</td>
<td>Beer, Coke, Diaper, Milk</td>
</tr>
<tr>
<td>4</td>
<td>Beer, Bread, Diaper, Milk</td>
</tr>
<tr>
<td>5</td>
<td>Coke, Diaper, Milk</td>
</tr>
</tbody>
</table>

**Output:**

**Rules Discovered:**

- \{Milk\} --> \{Coke\}
- \{Diaper, Milk\} --> \{Beer\}
Applications – (1)

- **Items** = products; **Baskets** = sets of products someone bought in one trip to the store
- **Real market baskets**: Chain stores keep TBs of data about what customers buy together
  - Tells how typical customers navigate stores, lets them position tempting items
  - Suggests tie-in “tricks”, e.g., run sale on diapers + raise the price of beer
  - Need the rule to occur frequently, or no $$’$$’s
- **Amazon’s people who bought X also bought Y**
Applications – (2)

- **Baskets** = sentences; **Items** = documents containing those sentences
  - Items that appear together too often could represent plagiarism
  - Notice items do not have to be “in” baskets

- **Baskets** = patients; **Items** = drugs & side-effects
  - Has been used to detect combinations of drugs that result in particular side-effects
  - **But requires extension**: Absence of an item needs to be observed as well as presence
More generally

- A general many-to-many mapping (association) between two kinds of things
  - But we ask about connections among “items”, not “baskets”

- For example:
  - Finding communities in graphs (e.g., Twitter)
Frequent Itemsets

- **Simplest question:** Find sets of items that appear together “frequently” in baskets
- **Support** for itemset $I$: Number of baskets containing all items in $I$
  - (Often expressed as a fraction of the total number of baskets)
- **Given a support threshold $s$**, then sets of items that appear in at least $s$ baskets are called *frequent itemsets*

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bread, Coke, Milk</td>
</tr>
<tr>
<td>2</td>
<td>Beer, Bread</td>
</tr>
<tr>
<td>3</td>
<td>Beer, Coke, Diaper, Milk</td>
</tr>
<tr>
<td>4</td>
<td>Beer, Bread, Diaper, Milk</td>
</tr>
<tr>
<td>5</td>
<td>Coke, Diaper, Milk</td>
</tr>
</tbody>
</table>

Support of \{Beer, Bread\} = 2
Example: Frequent Itemsets

- **Items = \{milk, coke, pepsi, beer, juice\}**
- **Support threshold = 3 baskets**

\[
\begin{align*}
B_1 &= \{m, c, b\} & B_2 &= \{m, p, j\} \\
B_3 &= \{m, b\} & B_4 &= \{c, j\} \\
B_5 &= \{m, c, b\} & B_6 &= \{m, c, b, j\} \\
B_7 &= \{c, b, j\} & B_8 &= \{b, c\}
\end{align*}
\]

- **Frequent itemsets:**

\[
\begin{align*}
\{m\}:5, & \{c\}:6, \{b\}:6, \{j\}:4, \{m,b\}:4, \\
\{m,c\}: 3, & \{c,b\}:5, \{c,j\}:3, \{m,c,b\}:3
\end{align*}
\]
Association Rules

- If-then rules about the contents of baskets
- \[ \{i_1, i_2, \ldots, i_k\} \rightarrow j \] means: “if a basket contains all of \( i_1, \ldots, i_k \) then it is likely to contain \( j \)”
- In practice there are many rules, want to find significant/interesting ones!
- **Confidence** of this association rule is the probability of \( j \) given \( I = \{i_1, \ldots, i_k\} \)

\[
\text{conf}(I \rightarrow j) = \frac{\text{support}(I \cup j)}{\text{support}(I)}
\]
Interesting Association Rules

- Not all high-confidence rules are interesting
  - The rule $X \rightarrow \text{milk}$ may have high confidence because milk is just purchased very often (independent of $X$)

- Interest of an association rule $I \rightarrow j$: difference between its confidence and the fraction of baskets that contain $j$

  \[
  \text{Interest}(I \rightarrow j) = \text{conf}(I \rightarrow j) - \Pr[j]
  \]

- Interesting rules are those with high positive or negative interest values (usually above 0.5)
Example: Confidence and Interest

\[ B_1 = \{m, c, b\} \quad B_2 = \{m, p, j\} \]
\[ B_3 = \{m, b\} \quad B_4 = \{c, j\} \]
\[ B_5 = \{m, c, b\} \quad B_6 = \{m, c, b, j\} \]
\[ B_7 = \{c, b, j\} \quad B_8 = \{b, c\} \]

- Association rule: \{m\} \rightarrow b
  - **Confidence** = \frac{4}{5}
  - **Interest** = \frac{4}{5} - \frac{6}{8} = \frac{1}{20}
    - Item \(b\) appears in \frac{6}{8} of the baskets
    - Rule is not very interesting!
Many measures of interest

<table>
<thead>
<tr>
<th>Measure (Symbol)</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goodman-Kruskal (λ)</td>
<td>( \frac{\sum_j \max_k f_{jk} - \max_k f_{+k}}{N - \max_k f_{+k}} )</td>
</tr>
<tr>
<td>Mutual Information (M)</td>
<td>( \frac{\sum_i \sum_j \frac{f_{ij}}{N} \log \frac{N f_{ij}}{f_{i+} f_{+j}}}{- \sum_i \frac{f_{i+}}{N} \log \frac{f_{i+}}{N}} )</td>
</tr>
<tr>
<td>J-Measure (J)</td>
<td>( \frac{f_{11}}{N} \log \frac{N f_{11}}{f_{1+} f_{+1}} + \frac{f_{10}}{N} \log \frac{N f_{10}}{f_{1+} f_{+0}} )</td>
</tr>
<tr>
<td>Gini index (G)</td>
<td>( \frac{f_{1+}}{N} \times \left( \frac{f_{11}}{f_{1+}} \right)^2 + \left( \frac{f_{10}}{f_{1+}} \right)^2 ) - ( \frac{f_{+1}}{N} )^2 )</td>
</tr>
<tr>
<td></td>
<td>( + \frac{f_{0+}}{N} \times \left[ \left( \frac{f_{01}}{f_{0+}} \right)^2 + \left( \frac{f_{00}}{f_{0+}} \right)^2 \right] - \left( \frac{f_{+0}}{N} \right)^2 )</td>
</tr>
<tr>
<td>Laplace (L)</td>
<td>( \frac{f_{11} + 1}{f_{1+} + 2} )</td>
</tr>
<tr>
<td>Conviction (V)</td>
<td>( \frac{f_{1+} f_{+0}}{(N f_{10})} )</td>
</tr>
<tr>
<td>Certainty factor (F)</td>
<td>( \frac{f_{11} - f_{+1}}{N f_{1+} - f_{+1}} ) - ( \frac{f_{+1}}{N} ) )</td>
</tr>
<tr>
<td>Added Value (AV)</td>
<td>( \frac{f_{11}}{f_{1+}} - \frac{f_{+1}}{N} )</td>
</tr>
</tbody>
</table>

**source**: Tan, Steinbach & Kumar, “Introduction to Data Mining”,
Finding Association Rules

Problem: Find all association rules with support ≥ s and confidence ≥ c

- Note: Support of an association rule is the support of the set of items on the left side

Hard part: Finding the frequent itemsets!

- If \{i_1, i_2, \ldots, i_k\} \rightarrow j has high support and confidence, then both \{i_1, i_2, \ldots, i_k\} and \{i_1, i_2, \ldots, i_k, j\} will be “frequent”

\[
\text{conf}(I \rightarrow j) = \frac{\text{support}(I \cup j)}{\text{support}(I)}
\]
Mining Association Rules

- **Step 1**: Find all frequent itemsets \( I \)
  - (we will explain this next)

- **Step 2**: Rule generation
  - For every subset \( A \) of \( I \), generate a rule \( A \rightarrow I \setminus A \)
    - Since \( I \) is frequent, \( A \) is also frequent
  - **Variant 1**: Single pass to compute the rule confidence
    - \( \text{confidence}(A,B \rightarrow C,D) = \frac{\text{support}(A,B,C,D)}{\text{support}(A,B)} \)
  - **Variant 2**:
    - **Observation**: If \( A, B, C \rightarrow D \) is below confidence, so is \( A, B \rightarrow C, D \)
    - Can generate “bigger” rules from smaller ones!
  - **Output the rules above the confidence threshold**
Example: Mining Association Rules

\[ B_1 = \{m, c, b\}, \quad B_2 = \{m, p, j\} \]
\[ B_3 = \{m, b\}, \quad B_4 = \{c, j\} \]
\[ B_5 = \{m, c, b\}, \quad B_6 = \{m, c, b, j\} \]
\[ B_7 = \{c, b, j\}, \quad B_8 = \{b, c\} \]

- Support threshold \( s = 3 \), confidence \( c = 0.75 \)

1) Frequent itemsets:
   - \( \{b,m\}:4 \quad \{c,m\}:3 \quad \{b,c\}:5 \quad \{c,j\}:3 \quad \{m,c,b\}: 3 \)

2) Generate rules:
   - \( b \rightarrow m: e=4/6 \quad b \rightarrow c: c=5/6 \quad b,c \rightarrow m: e=3/5 \)
   - \( m \rightarrow b: c=4/5 \quad \ldots \quad b,m \rightarrow c: c=3/4 \)
   - \( b \rightarrow c,m: e=3/6 \)
Finding Frequent Item Sets

A lattice structure can be used to enumerate the list of all possible itemsets. Figure 6.1 shows an itemset lattice for $I = \{a, b, c, d, e\}$. In general, a data set that contains $k$ items can potentially generate up to $2^k - 1$ frequent itemsets, excluding the null set. Because $k$ can be very large in many practical applications, the search space of itemsets that need to be explored is exponentially large.

A brute-force approach for finding frequent itemsets is to determine the support count for every candidate itemset in the lattice structure. To do this, we need to compare each candidate against every transaction, an operation that is shown in Figure 6.2. If the candidate is contained in a transaction, its support count will be incremented. For example, the support for $\{\text{Bread}, \text{Milk}\}$ is incremented three times because the itemset is contained in transactions 1, 4, and 5. Such an approach can be very expensive because it requires $O(NMw)$ comparisons, where $N$ is the number of transactions, $M = 2^k - 1$ is the number of candidate itemsets, and $w$ is the maximum transaction width.

Given $k$ products, how many possible item sets are there?
6.2 Frequent Itemset Generation

A lattice structure can be used to enumerate the list of all possible itemsets. Figure 6.1 shows an itemset lattice for \( I = \{a, b, c, d, e\} \). In general, a data set that contains \( k \) items can potentially generate up to \( 2^k - 1 \) frequent itemsets, excluding the null set. Because \( k \) can be very large in many practical applications, the search space of itemsets that need to be explored is exponentially large.

A brute-force approach for finding frequent itemsets is to determine the support count for every candidate itemset in the lattice structure. To do this, we need to compare each candidate against every transaction, an operation that is shown in Figure 6.2. If the candidate is contained in a transaction, its support count will be incremented. For example, the support for \( \{\text{Bread}, \text{Milk}\} \) is incremented three times because the itemset is contained in transactions 1, 4, and 5. Such an approach can be very expensive because it requires \( O(NMw) \) comparisons, where \( N \) is the number of transactions, \( M = 2^k - 1 \) is the number of candidate itemsets, and \( w \) is the maximum transaction width.

**Answer:** \( 2^k - 1 \rightarrow \text{Cannot enumerate all possible sets} \)

*Finding Frequent Item Sets*
Observation: A-priori Principle

Subsets of a frequent item set are also frequent
Corollary: Pruning of Candidates

If we know that a subset is not frequent, then we can ignore all its supersets.
A-priori Algorithm

Algorithm 6.1 Frequent itemset generation of the Apriori algorithm.

1: \( k = 1 \).
2: \( F_k = \{ i \mid i \in I \land \sigma(\{i\}) \geq N \times \text{minsup} \} \). \{Find all frequent 1-itemsets\}
3: repeat
4: \( k = k + 1 \).
5: \( C_k = \text{apriori-gen}(F_{k-1}) \). \{Generate candidate itemsets\}
6: for each transaction \( t \in T \) do
7: \( C_t = \text{subset}(C_k, t) \). \{Identify all candidates that belong to \( t \}\)
8: for each candidate itemset \( c \in C_t \) do
9: \( \sigma(c) = \sigma(c) + 1 \). \{Increment support count\}
10: end for
11: end for
12: \( F_k = \{ c \mid c \in C_k \land \sigma(c) \geq N \times \text{minsup} \} \). \{Extract the frequent \( k \)-itemsets\}
13: until \( F_k = \emptyset \)
14: Result = \( \bigcup F_k \).
Generating Candidates $C_k$

1. **Self-joining**: Find pairs of sets in $L_{k-1}$ that differ by **one** element

2. **Pruning**: Remove all candidates with infrequent subsets
Example: Generating Candidates $C_k$

- Frequent itemsets of size 2:
  - $\{m,b\}:4$, $\{m,c\}:3$, $\{c,b\}:5$, $\{c,j\}:3$

- Self-joining:
  - $\{m,b,c\}$, $\{b,c,j\}$

- Pruning:
  - $\{b,c,j\}$ since $\{b,j\}$ not frequent
Compacting the Output

To reduce the number of rules we can post-process them and only output:

- **Maximal frequent itemsets:**
  No immediate superset is frequent
  - Gives more pruning

- **Closed itemsets:**
  No immediate superset has same count (> 0)
  - Stores not only frequent information, but exact counts
Example: **Maximal vs Closed**

\[
\begin{align*}
B_1 &= \{m, c, b\} & B_2 &= \{m, p, j\} \\
B_3 &= \{m, b\} & B_4 &= \{c, j\} \\
B_5 &= \{m, c, b\} & B_6 &= \{m, c, b, j\} \\
B_7 &= \{c, b, j\} & B_8 &= \{b, c\}
\end{align*}
\]

**Frequent itemsets:**

\[
\begin{align*}
\{m\}:5, \{c\}:6, \{b\}:6, \{j\}:4, & \quad \text{Closed} \\
\{m,c\}:3, \{m,b\}:4, \{c,b\}:5, \{c,j\}:3, & \quad \text{Maximal} \\
\{m,c,b\}:3 &
\end{align*}
\]
Example: Maximal vs Closed

- **Frequent Itemsets**
  - **Closed Frequent Itemsets**
    - **Maximal Frequent Itemsets**
Hash Tree for Itemsets

Suppose you have 15 candidate itemsets of length 3:

\{1 \ 4 \ 5\}, \{1 \ 2 \ 4\}, \{4 \ 5 \ 7\}, \{1 \ 2 \ 5\}, \{4 \ 5 \ 8\}, \{1 \ 5 \ 9\}, \{1 \ 3 \ 6\}, \{2 \ 3 \ 4\}, \{5 \ 6 \ 7\}, \{3 \ 4 \ 5\},
\{3 \ 5 \ 6\}, \{3 \ 5 \ 7\}, \{6 \ 8 \ 9\}, \{3 \ 6 \ 7\}, \{3 \ 6 \ 8\}

You need:

• Hash function

• Max leaf size: max number of itemsets stored in a leaf node (if number of candidate itemsets exceeds max leaf size, split the node)
Hash Tree for Itemsets

Hash Function

1,4,7
2,5,8
3,6,9

Candidate Hash Tree

Hash on 1, 4 or 7
Hash Tree for Itemsets

Hash Function

1,4,7
2,5,8
3,6,9

Candidate Hash Tree

Hash on 2, 5 or 8

Yijun Zhao
DATA MINING TECHNIQUES Association Rule Mining
Hash Tree for Itemsets

**Hash Function**

1, 4, 7  
2, 5, 8  
3, 6, 9

**Candidate Hash Tree**

- Hash on 3, 6 or 9

```
1, 4, 7
2, 5, 8
3, 6, 9
```

```
1 4 5
1 2 4
1 2 5
```

```
2 3 4
5 6 7
1 3 6
```

```
3 4 5
3 5 6
3 6 7
3 5 7
3 6 8
```

```
1 3 6
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```

```
1 2 4
1 2 5
4 5 7
4 5 8
```
Given a transaction \( t \), what are the possible subsets of size 3?

### Level 1
- \( 1, 2, 3, 5, 6 \)

### Level 2
- \( 1, 2, 3, 5, 6 \)
- \( 1, 3, 5, 6 \)
- \( 1, 5, 6 \)
- \( 2, 3, 5, 6 \)
- \( 2, 5, 6 \)
- \( 3, 5, 6 \)

### Level 3
- \( 1, 2, 3 \)
- \( 1, 2, 5 \)
- \( 1, 2, 6 \)
- \( 1, 3, 5 \)
- \( 1, 3, 6 \)
- \( 1, 5, 6 \)
- \( 2, 3, 5 \)
- \( 2, 3, 6 \)
- \( 2, 5, 6 \)
- \( 3, 5, 6 \)

**Subsets of 3 items**

**Transaction, \( t \)**

\( 1, 2, 3, 5, 6 \)
Subset Operation

Hash Function

1, 4, 7
2, 5, 8
3, 6, 9

Subset Operation Using Hash Tree

1 2 3 5 6
1 4 5
1 3 6
3 4 5
3 6 7
3 6 8
3 5 6
3 5 7
6 8 9
2 3 4
5 6 7
1 2 4
4 5 7
1 2 5
4 5 8
1 2 3 5 6
1 + 2 3 5 6
3 5 6
2 + 3 5 6
3 + 5 6

Hash Function

1, 4, 7
2, 5, 8
3, 6, 9

Yijun Zhao
DATA MINING TECHNIQUES
Association Rule Mining

34 / 55
Subset Operation

Subset Operation Using Hash Tree

Hash Function

Transaction

Yijun Zhao
DATA MINING TECHNIQUES Association Rule Mining

35 / 55
Subset Operation

Match transaction against 11 out of 15 candidates