Discovering Rare Itemsets

Philippe Fournier-Viger
http://www.philippe-Fournier-viger.com

Source code and datasets available in the SPMF library
Introduction

- **Pattern mining:** using algorithms to discover interesting patterns in data.
- One of the most important pattern mining task is **frequent itemset mining**.
- It consists of finding sets of values (items) that appear frequently in the data (**frequent itemsets**).
- Today, I will talk about the opposite problem of discovering **rare itemsets**.
FREQUENT ITEMSET MINING (BRIEF REVIEW)
Frequent itemset mining (频繁项集挖掘)

A transaction database:

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Items appearing in the transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>{pasta, lemon, bread, orange}</td>
</tr>
<tr>
<td>T2</td>
<td>{pasta, lemon}</td>
</tr>
<tr>
<td>T3</td>
<td>{pasta, orange, cake}</td>
</tr>
<tr>
<td>T4</td>
<td>{pasta, lemon, orange cake}</td>
</tr>
</tbody>
</table>

For \( \text{minsup} = 2 \), the frequent itemsets are:
\{lemon\}, \{pasta\}, \{orange\}, \{cake\}, \{lemon, pasta\}, \{lemon, orange\}, \{pasta, orange\}, \{pasta, cake\}, \{orange, cake\}, \{lemon, pasta, orange\}
$\text{minsup} = 2$
Property 2: Let there be an itemset $Y$. If there exists an itemset $X \subset Y$ such that $X$ is infrequent, then $Y$ is infrequent.

Example:
- Consider $\{\text{bread, lemon}\}$.
- If we know that $\{\text{bread}\}$ is infrequent, then we can infer that $\{\text{bread, lemon}\}$ is also infrequent.

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Items appearing in the transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>{pasta, lemon, bread, orange}</td>
</tr>
<tr>
<td>T2</td>
<td>{pasta, lemon}</td>
</tr>
<tr>
<td>T3</td>
<td>{pasta, orange, cake}</td>
</tr>
<tr>
<td>T4</td>
<td>{pasta, lemon, orange, cake}</td>
</tr>
</tbody>
</table>
This property is useful to reduce the search space.

Example:

\[ \text{minsup} = 2 \]

If « bread » is infrequent
This property is useful to reduce the search space.

Example:

\[ \text{minsup} = 2 \]

If « bread » is infrequent, all its supersets are infrequent.
Limitations of frequent itemset mining

- There is an underlying hypothesis that something frequent must be important.
- But in practice, many frequent itemsets are unimportant.
- **Example**: many persons purchase bread and milk but it is not something surprising or profitable.
- Too many frequent patterns may make it hard to find rarer patterns that are interesting.
RARE PATTERN MINING
Finding rare patterns

- We first need to define what is a **rare pattern**.
- There are different definitions.
- I will give an overview.
We could define rare itemsets as **infrequent itemsets**.

**Definition 1 (infrequent itemset):** An itemset \( X \) is infrequent if \( \text{sup}(X) < \text{minsup} \).

\[ \text{minsup} = 2 \]

![Diagram showing frequent and infrequent itemsets with nodes labeled as l (lemon), p (pasta), b (bread), o (orange), c (cake).](image_url)
We could define rare itemsets as **infrequent itemsets**

**Problem:** Too many itemsets. Some infrequent itemsets do not even exist (support = 0)

minsups = 2

l = lemon  
p = pasta  
b = bread  
0 = orange  
c = cake
Another definition: minimal rare itemsets

Proposed for the AprioriRare algorithm:
Laszlo Szathmary, Amedeo Napoli, Petko Valtchev: Towards Rare Itemset Mining. ICTAI (1) 2007: 305-312

**Definition 1 (minimal rare itemset):** An itemset $X$ is a minimal rare itemset if $\text{sup}(X) < \text{minsup}$ and all its proper subsets are frequent itemsets (i.e. for any subset $Y \subset X$, $\text{sup}(Y) \geq \text{minsup}$).
Minimal rare itemsets

\( \text{minsup} = 2 \)

\[
\begin{array}{cccccc}
\emptyset & l & p & b & o & c \\
lp & lb & lo & lc & pb & po & pc & bo & bc & oc \\
lpb & lpo & lpc & lbo & lbc & loc & pbo & pbc & poc & boc \\
lpbo & lpbc & lpoc & lboc & pboc & lpboc
\end{array}
\]

l = lemon
p = pasta
b = bread
0 = orange
c = cake

frequent itemsets

Minimal rare itemsets
Minimal rare itemsets

\[ \text{minsup} = 2 \]

\( l = \text{lemon} \)
\( p = \text{pasta} \)
\( b = \text{bread} \)
\( o = \text{orange} \)
\( c = \text{cake} \)

\[ \emptyset \]

Support 1

frequent itemsets

Minimal rare itemsets
Minimal rare itemsets

\[ \text{minsup} = 2 \]

\[ l = \text{lemon} \]
\[ p = \text{pasta} \]
\[ b = \text{bread} \]
\[ o = \text{orange} \]
\[ c = \text{cake} \]

Support 1

Interesting, because not too many itemsets...
How can we find the minimal rare itemsets?

- It is not easy!
- Generally, frequent itemset mining algorithms start from single items and combine them to find larger itemsets.
- As itemsets become larger, the support can decrease.
- Thus, to search for rare itemsets, we must «pass through» the frequent itemsets to reach the rare itemsets. How?
The AprioriRare algorithm (2007)

- It is based on Apriori
- As Apriori, the itemsets are generated by levels:
  - Itemsets of size 1 (one item)
  - Itemsets of size 2 (two items)
  - Itemsets of size 3 (three items)
  - ...

- Two differences:
  - If AprioriRare finds an itemset of size k that is infrequent, AprioriRare checks if its subsets are frequent. If yes, it is a minimal rare itemset.
  - AprioriRare do not use the infrequent itemsets to generate larger itemsets.
The AprioriRare algorithm

- I will now explain how the AprioriRare algorithm works

**Input:**
- $\text{minsup}$
- a transactional database

**Output:**
- all the minimal rare itemsets

Consider $\text{minsup} = 2$. 
The AprioriRare algorithm

**Step 1**: Scan the database to calculate the support of all itemsets of size 1.

**e.g.**

- \{pasta\} support = 4
- \{lemon\} support = 3
- \{bread\} support = 1
- \{orange\} support = 3
- \{cake\} support = 2
The AprioriRare algorithm

Step 2: Check all infrequent itemsets. If all subsets are frequent, they are minimal rare itemsets.

e.g.

{pasta} support = 4
{lemon} support = 3
{bread} support = 1
{orange} support = 3
{cake} support = 2
The AprioriRare algorithm

**Step 2**: Check all infrequent itemsets. If all subsets are frequent, they are minimal rare itemsets.

e.g.

- \{pasta\} support = 4
- \{lemon\} support = 3
- \{bread\} support = 1
- \{orange\} support = 3
- \{cake\} support = 2
The AprioriRare algorithm

Step 2: Keep frequent itemsets.

e.g.

\{pasta\} \quad \text{support} = 4
\{lemon\} \quad \text{support} = 3
\{orange\} \quad \text{support} = 3
\{cake\} \quad \text{support} = 2
The AprioriRare algorithm

**Step 3:** Generate candidates of size 2 by combining pairs of frequent itemsets of size 1.

- Frequent items:
  - \{pasta\}
  - \{lemon\}
  - \{orange\}
  - \{cake\}

- Candidates of size 2:
  - \{pasta, lemon\}
  - \{pasta, orange\}
  - \{pasta, cake\}
  - \{lemon, orange\}
  - \{lemon, cake\}
  - \{orange, cake\}
The AprioriRare algorithm

**Step 5**: Scan the database to calculate the support of remaining candidate itemsets of size 2.

Candidates of size 2

- \{pasta, lemon\} support: 3
- \{pasta, orange\} support: 3
- \{pasta, cake\} support: 2
- \{lemon, orange\} support: 2
- \{lemon, cake\} support: 1
- \{orange, cake\} support: 2
The AprioriRare algorithm

Step 6: For each infrequent itemset, check if all the subsets are frequent

Candidates of size 2

\{\text{pasta, lemon}\} \quad \text{support: 3}
\{\text{pasta, orange}\} \quad \text{support: 3}
\{\text{pasta, cake}\} \quad \text{support: 2}
\{\text{lemon, orange}\} \quad \text{support: 2}
\{\text{lemon, cake}\} \quad \text{support: 1}
\{\text{orange, cake}\} \quad \text{support: 2}
**The AprioriRare algorithm**

**Step 6:** For each infrequent itemset, check if all the subsets are frequent

Candidates of size 2

- \{pasta, lemon\} support: 3
- \{pasta, orange\} support: 3
- \{pasta, cake\} support: 2
- \{lemon, orange\} support: 2
- \{lemon, cake\} support: 1
- \{orange, cake\} support: 2

**Minimal Rare Itemset**
Step 6: Keep frequent itemsets of size 2

Frequent itemsets of size 2

{pasta, lemon} support: 3
{pasta, orange} support: 3
{pasta, cake} support: 2
{lemon, orange} support: 2
{orange, cake} support: 2
The AprioriRare algorithm

**Step 7:** Generate candidates of size 3 by combining frequent pairs of itemsets of size 2.

**Frequent itemsets of size 2**

- \{pasta, lemon\}
- \{pasta, orange\}
- \{pasta, cake\}
- \{lemon, orange\}
- \{orange, cake\}

**Candidates of size 3**

- \{pasta, lemon, orange\}
- \{pasta, lemon, cake\}
- \{pasta, orange, cake\}
- \{lemon, orange, cake\}
The AprioriRare algorithm

**Step 7:** Scan the database to find the support of candidates

**Frequent itemsets of size 2**

- {pasta, lemon}
- {pasta, orange}
- {pasta, cake}
- {lemon, orange}
- {orange, cake}

**Candidates of size 3**

- {pasta, lemon, orange}: 2
- {pasta, lemon, cake}: 1
- {pasta, orange, cake}: 2
- {lemon, orange, cake}: 1
Step 8: For each infrequent itemset, check if all the subsets are frequent

Frequent itemsets of size 2

{pasta, lemon}
{pasta, orange}
{pasta, cake}
{lemon, orange}
{orange, cake}

Candidates of size 3

{pasta, lemon, orange}: 2
{pasta, lemon, cake}: 1
{pasta, orange, cake}: 2
{lemon, orange, cake}: 1
The Apriori Rare algorithm

**Step 8**: For each infrequent itemset, check if all the subsets are frequent

**Frequent itemsets of size 2**

\{pasta, lemon\}  
\{pasta, orange\}  
\{pasta, cake\}  
\{lemon, orange\}  
\{orange, cake\}

**Candidates of size 3**

\{pasta, lemon, orange\}: 2  
\{pasta, lemon, cake\}: 1  
\{pasta, orange, cake\}: 2  
\{lemon, orange, cake\}: 1
The AprioriRare algorithm

Step 8: For each infrequent itemset, check if all the subsets are frequent

Frequent itemsets of size 2

{pasta, lemon}
{pasta, orange}
{pasta, cake}
{lemon, orange}
{orange, cake}

Candidates of size 3

{pasta, lemon, orange}
{pasta, orange, cake}
The AprioriRare algorithm

Step 10: Keep the frequent itemsets (all)

frequent itemsets of size 3

\{pasta, lemon, orange\} support: 2
\{pasta, orange, cake\} support: 2
The AprioriRare algorithm

Step 11: generate candidates of size 4 by combining pairs of frequent itemsets of size 3.

Frequent itemsets of size 3

\{pasta, lemon, orange\}

\{pasta, orange, cake\}

Candidates of size 4

\{pasta, lemon, orange, cake\}
The AprioriRare algorithm

Step 12: Check to see if the subsets of each infrequent itemset are frequent. They are not.

Frequent itemsets of size 3

\{pasta, lemon, orange\}

\{pasta, orange, cake\}

Candidates of size 4

\{pasta, lemon, orange, cake\}
The AprioriRare algorithm

Step 12: Since there is no more frequent itemsets, we cannot generate candidates of size 5 and the algorithm stops.

Candidates of size 4

\{pasta, lemon, orange, cake\}
Final result

The minimal rare itemsets:

\{bread\}
\{lemon, cake\}

support = 1
support = 1
Another definition: perfectly rare itemsets

Proposed for the AprioriInverse algorithm:

\textbf{Definition 1 (perfectly rare itemset):} Let there be two thresholds \textit{minsup} and \textit{maxsup}, such that \textit{maxsup} > \textit{minsup}.

An itemset \( Z \) is a \textbf{frequent itemset} if \( \text{sup}(Z) \geq \text{maxsup} \).

An itemset \( X \) is a \textbf{perfectly rare itemset} if \( \text{sup}(X) \geq \text{minsup} \) and \( \text{sup}(X) < \text{maxsup} \) and for any non empty subset \( Y \subset X \), \( \text{sup}(Y) < \text{maxsup} \).
Definition 1 (perfectly rare itemset): Let there be two thresholds \( \text{minsup} \) and \( \text{maxsup} \), such that \( \text{maxsup} > \text{minsup} \). An itemset \( Z \) is a **frequent itemset** if \( \text{sup}(Z) \geq \text{maxsup} \). An itemset \( X \) is a **perfectly rare itemset** if \( \text{sup}(X) \geq \text{minsup} \), \( \text{sup}(X) < \text{maxsup} \) and for any non-empty subset \( Y \subset X \), \( \text{sup}(Y) < \text{maxsup} \).

\( \text{maxsup} = 1.9 \)
\( \text{minsup} = 1 \)
Definition 1 (perfectly rare itemset):
Let there be two thresholds \( \minsup \) and \( \maxsup \), such that \( \maxsup > \minsup \). An itemset \( Z \) is a **frequent itemset** if \( \text{sup}(Z) \geq \maxsup \).
An itemset \( X \) is a **perfectly rare itemset** if \( \text{sup}(X) \geq \minsup \), \( \text{sup}(X) < \maxsup \) and for any non empty subset \( Y \subset X \), \( \text{sup}(Y) < \maxsup \).
Definition 1 (perfectly rare itemset):
Let there be two thresholds \( \text{minsup} \) and \( \text{maxsup} \), such that \( \text{maxsup} > \text{minsup} \). An itemset \( Z \) is a **frequent itemset** if \( \text{sup}(Z) \geq \text{maxsup} \). An itemset \( X \) is a **perfectly rare itemset** if \( \text{sup}(X) \geq \text{minsup}, \text{sup}(X) < \text{maxsup} \) and for any non empty subset \( Y \subset X \), \( \text{sup}(Y) < \text{maxsup} \).

\[
\begin{align*}
\text{maxsup} &= 3.1 \\
\text{minsup} &= 1.1
\end{align*}
\]
Definition 1 (perfectly rare itemset):
Let there be two thresholds minsup and maxsup, such that maxsup > minsup. An itemset $Z$ is a frequent itemset if $\text{sup}(Z) \geq \text{maxsup}$. An itemset $X$ is a perfectly rare itemset if $\text{sup}(X) \geq \text{minsup}$, $\text{sup}(X) < \text{maxsup}$ and for any non empty subset $Y \subset X$, $\text{sup}(Y) < \text{maxsup}$.

$maxsup = 2$
$minsup = 1$
How to find perfectly rare itemsets?

- AprioriInverse (2005)
- Based on Apriori.
- Key difference:
  - Initially, AprioriInverse discards each item $x$ such that $\text{sup}(x) \geq \text{maxsup}$ because we don’t need such item.
  - After that AprioriInverse search for itemsets using the remaining items just like Apriori to find itemsets with a support no less than $\text{minsup}$. 
Conclusion

This video has presented:

- The problem of rare itemset mining
- Three definitions of rare itemsets:
  - Infrequent itemsets
  - Minimal rare itemsets
  - Perfectly rare itemsets
- Two algorithms:
  - AprioriRare
  - AprioriInverse
- To find rare itemsets that are more interesting, we can also combine the concept of rare itemsets with that of correlated itemset (e.g. the CORI algorithm).