Frequent Item Sets and Association Rules

Prof. Dan A. Simovici

UMB
1. Rymon Trees
2. Frequent Item Sets
3. Association Rules
Search enumeration trees were introduced by R. Rymon in order to provide a unified search-based framework for several problems in artificial intelligence; they are also useful for data mining algorithms. Let $S$ be a set and let $d : S \rightarrow \mathbb{N}$ be an injective function. The number $d(x)$ is the index of $x \in S$.

If $P \subseteq S$, the view of $P$ is the subset

$$\text{view}(d, P) = \left\{ s \in S \mid d(s) > \max_{p \in P} d(p) \right\}.$$

Definition

If $C$ is a collection of sets we say that $C$ is **hereditary** if $A \in C$ and $B \subseteq A$ implies $B \in C$.

The collection $\mathcal{P}(S)$ of subsets of a set $S$ is hereditary.
Definition

Let $C$ be a hereditary collection of subsets of a set $S$. The graph $G = (C, E)$ is a Rymon tree for $C$ and the indexing function $d$ if

1. the root of $G$ is the empty set, and
2. the children of a node $P$ are the sets of the form $P \cup \{s\}$, where $s \in \text{view}(d, P)$.

If $S = \{s_1, \ldots, s_n\}$ and $d(s_i) = i$ for $1 \leq i \leq n$, we will omit the indexing function from the definition of the Rymon tree for $\mathcal{P}(S)$. 
Example

Let \( S = \{i_1, i_2, i_3, i_4\} \) and let \( C = \mathcal{P}(S) \), which is clearly a hereditary collection of sets. Define the injective mapping \( d \) by \( d(i_k) = k \) for \( 1 \leq k \leq 4 \). The Rymon tree for \( C \) and \( d \) is shown next.
Theorem

Let \( G \) be a Rymon tree for a hereditary collection \( C \) of subsets of a set \( S \) and an indexing function \( d \). Every set \( P \) of \( C \) occurs exactly once in the tree.

Proof.

The argument is by induction on \( p = |P| \). If \( p = 0 \), then \( P \) is the root of the tree and the theorem obviously holds.

Suppose that the theorem holds for sets having fewer than \( p \) elements, and let \( P \in C \) be such that \( |P| = p \). Since \( C \) is hereditary, every set of the form \( P - \{x\} \) with \( x \in P \) belongs to \( C \) and, by the inductive hypothesis, occurs exactly once in the tree.

Let \( z \) be the element of \( P \) that has the largest value of the index function \( d \). Then \( \text{view}(P - \{z\}) \) contains \( z \) and \( P \) is a child of the vertex \( P - \{z\} \). Since the parent of \( P \) is unique, it follows that \( P \) occurs exactly once in the tree.
If a set $U$ is located at the left of a set $V$ in the tree $G_I$, we shall write $U \sqsubset V$. Thus, we have

$$\emptyset \sqsubset \{i_1\} \sqsubset \{i_1, i_2\} \sqsubset \{i_1, i_2, i_3, i_4\}$$

$$\sqsubset \{i_1, i_2, i_4\} \sqsubset \{i_1, i_3\} \sqsubset \{i_1, i_3, i_4\}$$

$$\sqsubset \{i_1, i_4\} \sqsubset \{i_2\} \sqsubset \{i_2, i_3\}$$

$$\sqsubset \{i_2, i_3, i_4\} \sqsubset \{i_2, i_4\} \sqsubset \{i_3\}$$

$$\sqsubset \{i_3, i_4\} \sqsubset \{i_4\}.$$

Note that in the Rymon tree of a collection of the form $\mathcal{P}(S)$, the collection of sets of $S_r$ that consists of sets located at distance $r$ from the root denotes all $\binom{n}{r}$ subsets of size $r$ of $S$. 
Suppose that \( I \) is a finite set; we refer to the elements of \( I \) as *items*.

**Definition**

A *transaction data set on \( I \)* is a function \( T : \{1, \ldots, n\} \rightarrow \mathcal{P}(I) \).

The set \( T(k) \) is the \( k^{th} \) transaction of \( T \). The numbers \( 1, \ldots, n \) are the *transaction identifiers* (tids).

An example of a transaction set is the set of items present in the shopping cart of a consumer that completed a purchase in a store.
Example

The table below describes a transaction data set on the set of over-the-counter medicines in a drugstore.

<table>
<thead>
<tr>
<th>Trans.</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T(1) )</td>
<td>{Aspirin, Vitamin C}</td>
</tr>
<tr>
<td>( T(2) )</td>
<td>{Aspirin, Sudafed}</td>
</tr>
<tr>
<td>( T(3) )</td>
<td>{Tylenol}</td>
</tr>
<tr>
<td>( T(4) )</td>
<td>{Aspirin, Vitamin C, Sudafed}</td>
</tr>
<tr>
<td>( T(5) )</td>
<td>{Tylenol, Cepacol}</td>
</tr>
<tr>
<td>( T(6) )</td>
<td>{Aspirin, Cepacol}</td>
</tr>
<tr>
<td>( T(7) )</td>
<td>{Aspirin, Vitamin C}</td>
</tr>
</tbody>
</table>
Example

The same data set can be presented as a 0/1 table:

<table>
<thead>
<tr>
<th></th>
<th>Aspirin</th>
<th>Vitamin C</th>
<th>Sudafed</th>
<th>Tylenol</th>
<th>Cepacol</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T(1)$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$T(2)$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$T(3)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$T(4)$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$T(5)$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$T(6)$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$T(7)$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The entry in the row $T(k)$ and the column $i_j$ is set to 1 if $i_j \in T(k)$; otherwise, it is set to 0.
Given a transaction data set $T$ on the set $I$, we would like to determine those subsets of $I$ that occur often enough as values of $T$.

**Definition**

Let $T : \{1, \ldots, n\} \rightarrow \mathcal{P}(I)$ be a transaction data set on a set of items $I$. The *support count* of a subset $K$ of the set of items $I$ in $T$ is the number $\text{suppcount}_T(K)$ given by

$$\text{suppcount}_T(K) = |\{k \mid 1 \leq k \leq n \text{ and } K \subseteq T(k)\}|.$$

The *support* of an item set $K$ is the number

$$\text{supp}_T(K) = \frac{\text{suppcount}_T(K)}{n}.$$
Example

For the transaction data set $T$ considered before, we have

$$\text{suppcount}_T(\{\text{Aspirin, VitaminC}\}) = 3$$

because $\{\text{Aspirin, VitaminC}\}$ is a subset of three of the sets $T(k)$. Therefore, $\text{supp}_T(\{\text{Aspirin, VitaminC}\}) = \frac{3}{7}$. 
Example

Let $I = \{i_1, i_2, i_3, i_4\}$ be a collection of items. Consider the transaction data set $T$ given by

\[
\begin{align*}
T(1) &= \{i_1, i_2\}, \\
T(2) &= \{i_1, i_3\}, \\
T(3) &= \{i_1, i_2, i_4\}, \\
T(4) &= \{i_1, i_3, i_4\}, \\
T(5) &= \{i_1, i_2\}, \\
T(6) &= \{i_3, i_4\}.
\end{align*}
\]

Thus, the support count of the item set $\{i_1, i_2\}$ is 3; similarly, the support count of the item set $\{i_1, i_3\}$ is 2. Therefore, $\text{supp}_T(\{i_1, i_2\}) = \frac{1}{2}$ and $\text{supp}_T(\{i_1, i_3\}) = \frac{1}{3}$. 
The following rather straightforward statement is fundamental for the study of frequent item sets.

**Theorem**

Let $T : \{1, \ldots, n\} \rightarrow \mathcal{P}(I)$ be a transaction data set on a set of items $I$. If $K$ and $K'$ are two item sets, then $K' \subseteq K$ implies $\text{supp}_T(K') \geq \text{supp}_T(K)$.

**Proof:** Note that every transaction that contains $K$ also contains $K'$. The statement follows immediately.
If we seek those item sets that enjoy a minimum support level relative to a transaction data set $T$, then it is natural to start the process with the smallest nonempty item sets.

**Definition**

An item set $K$ is $\mu$-frequent relative to the transaction data set $T$ if $\text{supp}_T(K) \geq \mu$.

We denote by $\mathcal{F}^\mu_T$ the collection of all $\mu$-frequent item sets relative to the transaction data set $T$ and by $\mathcal{F}^\mu_T, r$ the collection of $\mu$-frequent item sets that contain $r$ items for $r \geq 1$. 
Note that

$$\mathcal{F}_T^\mu = \bigcup_{r \geq 1} \mathcal{F}_T^\mu, r.$$  

If $\mu$ and $T$ are clear from the context, then we may omit either or both adornments from this notation.

Let $I = \{i_1, \ldots, i_n\}$ be an item set that contains $n$ elements.

Denote by $G_I = (\mathcal{P}(I), E)$ the Rymon tree of $\mathcal{P}(I)$. Recall that the root of the tree is $\emptyset$. A vertex $K = \{i_{p_1}, \ldots, i_{p_k}\}$ with $i_{p_1} < i_{p_2} < \cdots < i_{p_k}$ has $n - i_{p_k}$ children $K \cup \{j\}$, where $i_{p_k} < j \leq n$. 
Let $S_r$ be the collection of item sets that have $r$ elements. The next theorem suggests a technique for generating $S_{r+1}$ starting from $S_r$.

**Theorem**

Let $G$ be the Rymon tree of $\mathcal{P}(I)$, where $I = \{i_1, \ldots, i_n\}$. If $W \in S_{r+1}$, where $r \geq 2$, then there exists a unique pair of distinct sets $U, V \in S_r$ that has a common immediate ancestor $T \in S_{r-1}$ in $G$ such that $U \cap V \in S_{r-1}$ and $W = U \cup V$. 
Frequent Item Sets

Proof:
Let $u$ and $v$ be the two elements of $W$ that have the largest and the second-largest subscripts, respectively. Consider the sets $U = W - \{u\}$ and $V = W - \{v\}$. Both sets belong to $S_r$. Moreover, $Z = U \cap V$ belongs to $S_{r-1}$ because it consists of the first $r - 1$ elements of $W$. Note that both $U$ and $V$ are descendants of $Z$ and that $U \cup V = W$. 


The pair \((U, V)\) is unique. Indeed, suppose that \(W\) can be obtained in the same manner from another pair of distinct sets \(U', V' \in S_r\) such that \(U'\) and \(V'\) are immediate descendants of a set \(Z' \in S_{r-1}\). The definition of the Rymon tree \(G_I\) implies that \(U' = Z' \cup \{i_m\}\) and \(V' = Z' \cup \{i_q\}\), where the letters in \(Z'\) are indexed by a number smaller than \(\min\{m, q\}\). Then, \(Z'\) consists of the first \(r - 1\) symbols of \(W\), so \(Z' = Z\). If \(m < q\), then \(m\) is the second-highest index of a symbol in \(W\) and \(q\) is the highest index of a symbol in \(W\), so \(U' = U\) and \(V' = V\).
Example

Consider the Rymon tree of the collection $\mathcal{P}(\{i_1, i_2, i_3, i_4\})$. 

The set $\{i_1, i_3, i_4\}$ is the union of the sets $\{i_1, i_2\}$ and $\{i_1, i_4\}$ that have the common ancestor $\{i_1\}$. 
Next we discuss an algorithm that allows us to compute the collection $F^\mu_T$ of all $\mu$-frequent item sets for a transaction data set $T$. The algorithm is known as the *Apriori algorithm*. We begin with the procedure `apriori_gen`, which starts with the collection $F^\mu_{T,k}$ of frequent item sets for the transaction data set $T$ that contain $k$ elements and generates a collection $C_{k+1}$ of sets of items that contains $F^\mu_{T,k+1}$, the collection of the frequent item sets that have $k+1$ elements. The justification for this procedure is based on the next statement.
Theorem

Let $T$ be a transaction data set on a set of items $I$ and let $k \in \mathbb{N}$ such that $k > 1$.

If $W$ is a $\mu$-frequent item set and $|W| = k + 1$, then there exists a $\mu$-frequent item set $Z$ and two items $i_m$ and $i_q$ such that $|Z| = k - 1$, $Z \subseteq W$, $W = Z \cup \{i_m, i_q\}$, and both $Z \cup \{i_m\}$ and $Z \cup \{i_q\}$ are $\mu$-frequent item sets.

Proof: If $W$ is an item set such that $|W| = k + 1$, then we already know that $W$ is the union of two subsets $U$ and $V$ of $I$ such that $|U| = |V| = k$ and that $Z = U \cap V$ has $k - 1$ elements. Since $W$ is a $\mu$-frequent item set and $Z, U, V$ are subsets of $W$, it follows that each of these sets is also a $\mu$-frequent item set.
The reciprocal statement is not true, as the next example shows.

**Example**

Let $T$ be the transaction data set introduced above. Note that both $\{i_1, i_2\}$ and $\{i_1, i_3\}$ are $\frac{1}{3}$-frequent item sets; however,

$$supp_T(\{i_1, i_2, i_3\}) = 0,$$

so $\{i_1, i_2, i_3\}$ fails to be a $\frac{1}{3}$-frequent item set.
The procedure apriori-gen mentioned above is given next. It starts with the collection of item sets $\mathcal{F}_{T,k}$ and produces a collection of item sets $\mathcal{C}_{T,k+1}$ that includes the collection of item sets $\mathcal{F}_{T,k+1}$ of frequent item sets having $k + 1$ elements.

**Data:** a minimum support $\mu$, the collection $\mathcal{F}_{T,k}^{\mu}$ of frequent item sets having $k$ elements;

**Result:** the set of candidate frequent item sets $\mathcal{C}_{T,k+1}^{\mu}$;

set $j = 1$;

$C_{T,j+1}^{\mu} = \emptyset$;

**For** $(L, M \in \mathcal{F}_{T,k}^{\mu}$ such that $L \neq M$ and $L \cap M \in \mathcal{F}_{T,k-1}^{\mu}$)

add $L \cup M$ to $C_{T,k+1}^{\mu}$;

remove all sets $K$ in $C_{T,k+1}^{\mu}$ where there is a subset of $K$ containing $k$ elements that does not belong to $\mathcal{F}_{T,k}^{\mu}$;
Note that in apriori_gen no access to the transaction data set is needed.

- The Apriori algorithm operates on “levels.”
- Each level $k$ consists of a collection $C_{T,k}^{\mu}$ of candidate item sets of $\mu$-frequent item sets.
- To build the initial collection of candidate item sets $C_{T,1}^{\mu}$, every single item set is considered for membership in $C_{T,1}^{\mu}$.
The initial set of frequent item sets consists of those singletons that pass the minimal support test.

The algorithm alternates between a candidate generation phase (accomplished by using \texttt{apriori\_gen}) and an evaluation phase that involves a data set scan and is therefore the most expensive component of the algorithm.
The Apriori Algorithm

Data: transaction data set $T$ and a minimum support $\mu$;
Result: the collection $\mathcal{F}_T^\mu$ of $\mu$-frequent item sets;

$C_{T,1}^\mu = \{\{i\} \mid i \in I\}$;
set $i = 1$;
while ($C_{T,i}^\mu \neq \emptyset$) {
    $\mathcal{F}_{T,i}^\mu = \{L \in C_{T,i}^\mu \mid \text{supp}_T(L) \geq \mu\}$;
    $C_{T,i+1}^\mu = \text{apriori.gen}(\mathcal{F}_{T,i}^\mu)$;
    $i++$;
}
return $\mathcal{F}_T^\mu = \bigcup_{j<i} \mathcal{F}_{T,j}^\mu$
Definition

An association rule on an item set \( I \) is a pair of nonempty disjoint item sets \((X, Y)\).
Note that if \( |I| = n \), then there exist \( 3^n - 2^{n+1} + 1 \) association rules on \( I \).

Indeed, suppose that the set \( X \) contains \( k \) elements; there are \( \binom{n}{k} \) ways of choosing \( X \). Once \( X \) is chosen, \( Y \) can be chosen among the remaining \( 2^{n-k} - 1 \) nonempty subsets of \( I - X \). In other words, the number of association rules is

\[
\sum_{k=1}^{n} \binom{n}{k} (2^{n-k} - 1) = \sum_{k=1}^{n} \binom{n}{k} 2^{n-k} - \sum_{k=1}^{n} \binom{n}{k}.
\]

By taking \( x = 2 \) in the equality

\[
(1 + x)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k},
\]

we obtain

\[
\sum_{k=1}^{n} \binom{n}{k} 2^{n-k} = 3^n - 2^n.
\]

Since \( \sum_{k=1}^{n} \binom{n}{k} = 2^n - 1 \) the desired equality follows immediately.
The number of association rules can be quite considerable even for small values of \( n \). For example, for \( n = 10 \), we have \( 3^{10} - 2^{11} + 1 = 57002 \) association rules.

An association rule \((X, Y)\) is denoted by \( X \Rightarrow Y \). The confidence of \( X \Rightarrow Y \) is the number

\[
\text{conf}_T(X \Rightarrow Y) = \frac{\text{supp}_T(XY)}{\text{supp}_T(X)}.
\]

**Definition**

An association rule holds in a transaction data set \( T \) with support \( \mu \) and confidence \( c \) if \( \text{supp}_T(XY) \geq \mu \) and \( \text{conf}_T(X \Rightarrow Y) \geq c \).
Once a $\mu$-frequent item set $Z$ is identified, we need to examine the support levels of the subsets $X$ of $Z$ to ensure that an association rule of the form $X \Rightarrow Z - X$ has a sufficient level of confidence,

$$\text{conf}_T(X \Rightarrow Z - X) = \frac{\mu}{\text{supp}_T(X)}.$$ 

Observe that $\text{supp}_T(X) \geq \mu$ because $X$ is a subset of $Z$. To obtain a high level of confidence for $X \Rightarrow Z - X$, the support of $X$ must be as small as possible. Clearly, if $X \Rightarrow Z - X$ does not meet the level of confidence, then it is pointless to look for rules of the form $X' \Rightarrow Z - X'$ among the subsets $X'$ of $X$. 
Note that $i_2i_3i_4 \Rightarrow i_5$, $i_2i_4i_5 \Rightarrow i_3$, and $i_3i_4i_5 \Rightarrow i_2$ have 100% confidence. We refer to such rules as exact association rules.
The rule \( i_2i_3i_5 \Rightarrow i_4 \) has confidence \( \frac{2}{3} \). It is clear that the confidence of rules of the form \( U \Rightarrow V \) with \( U \subseteq i_2i_3i_5 \) and \( UV = L \) will be lower than \( \frac{2}{3} \) since \( \text{supp}_T(U) \) is at least 3. Indeed, the possible rules of this form are:

<table>
<thead>
<tr>
<th>Rule</th>
<th>( \text{suppcount}_T(X) )</th>
<th>( \text{conf}_T(X \Rightarrow Y) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i_2i_3 \Rightarrow i_4i_5 )</td>
<td>5</td>
<td>( \frac{2}{3} )</td>
</tr>
<tr>
<td>( i_2i_5 \Rightarrow i_3i_4 )</td>
<td>3</td>
<td>( \frac{2}{3} )</td>
</tr>
<tr>
<td>( i_3i_5 \Rightarrow i_2i_4 )</td>
<td>3</td>
<td>( \frac{2}{3} )</td>
</tr>
<tr>
<td>( i_2 \Rightarrow i_3i_4i_5 )</td>
<td>6</td>
<td>( \frac{5}{3} )</td>
</tr>
<tr>
<td>( i_3 \Rightarrow i_2i_4i_5 )</td>
<td>5</td>
<td>( \frac{5}{3} )</td>
</tr>
<tr>
<td>( i_5 \Rightarrow i_2i_3i_4 )</td>
<td>5</td>
<td>( \frac{5}{3} )</td>
</tr>
</tbody>
</table>

Obviously, if we seek association rules having a confidence larger than \( \frac{2}{3} \), no such rule \( U \Rightarrow V \) can be found such that \( U \) is a subset of \( i_2i_3i_5 \).
Suppose, for example, that we seek association rules $U \Rightarrow V$ that have a minimal confidence of 80%. We need to examine subsets $U$ of the other sets, $i_2i_3i_4$, $i_2i_4i_5$, or $i_3i_4i_5$, which are not subsets of $i_2i_3i_5$ (since the subsets of $i_2i_3i_5$ cannot yield levels of confidence higher than $\frac{2}{3}$). There are five such sets:

<table>
<thead>
<tr>
<th>Rule</th>
<th>$suppcount_T(X)$</th>
<th>$conf_T(X \Rightarrow Y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i_2i_4 \Rightarrow i_3i_5$</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$i_3i_4 \Rightarrow i_2i_5$</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$i_4i_5 \Rightarrow i_2i_3$</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$i_3i_4 \Rightarrow i_2i_5$</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$i_4 \Rightarrow i_2i_3i_5$</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Indeed, all these sets yield exact rules, that is, rules having 100% confidence.