Rule Mining and the Apriori Algorithm

MIT 15.097 Course Notes
Cynthia Rudin

The Apriori algorithm - often called the “first thing data miners try,” but somehow doesn’t appear in most data mining textbooks or courses!

Start with market basket data:

Some important definitions:

- **Itemset**: a subset of items, e.g., (bananas, cherries, elderberries), indexed by \{2, 3, 5\}.

- **Support** of an itemset: number of transactions containing it,

  \[
  \text{Supp}(\text{bananas, cherries, elderberries}) = \sum_{i=1}^{m} M_{i,2} \cdot M_{i,3} \cdot M_{i,5}.
  \]

- **Confidence** of rule \(a \rightarrow b\): the fraction of times itemset \(b\) is purchased when itemset \(a\) is purchased.

  \[
  \text{Conf}(a \rightarrow b) = \frac{\text{Supp}(a \cup b)}{\text{Supp}(a)} = \frac{\#\text{times } a \text{ and } b \text{ are purchased}}{\#\text{times } a \text{ is purchased}} = \hat{P}(b|a).
  \]
We want to find all strong rules. These are rules \( a \rightarrow b \) such that:

\[
\text{Supp}(a \cup b) \geq \theta, \quad \text{and} \quad \text{Conf}(a \rightarrow b) \geq \minconf.
\]

Here \( \theta \) is called the minimum support threshold.

The support has a monotonicity property called downward closure:

\[
\text{If Supp}(a \cup b) \geq \theta \text{ then } \text{Supp}(a) \geq \theta \text{ and } \text{Supp}(b) \geq \theta.
\]

That is, if \( a \cup b \) is a frequent item set, then so are \( a \) and \( b \).

\[
\text{Supp}(a \cup b) = \# \text{times } a \text{ and } b \text{ are purchased} \\
\leq \# \text{times } a \text{ is purchased} = \text{Supp}(a).
\]

Apriori finds all frequent itemsets \( (a \text{ such that Supp}(a) \geq \theta) \). We can use Apriori’s result to get all strong rules \( a \rightarrow b \) as follows:

- For each frequent itemset \( \ell \):
  - Find all nonempty subsets of \( \ell \)
  - For each subset \( a \), output \( a \rightarrow \{\ell \setminus a\} \) whenever
    \[
    \frac{\text{Supp}(\ell)}{\text{Supp}(a)} \geq \minconf.
    \]

Now for Apriori. Use the downward closure property: generate all \( k \)-itemsets (itemsets of size \( k \)) from \( (k - 1) \)-itemsets. It’s a breadth-first-search.
Example:

θ = 10

<table>
<thead>
<tr>
<th></th>
<th>apples</th>
<th>bananas</th>
<th>cherries</th>
<th>elderberries</th>
<th>grapes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-itemsets:</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
</tr>
<tr>
<td>supp:</td>
<td>25</td>
<td>20</td>
<td>30</td>
<td>45</td>
<td>29</td>
</tr>
<tr>
<td>2-itemsets:</td>
<td>{a,b}</td>
<td>{a,c}</td>
<td>{a,d}</td>
<td>{a,e}</td>
<td>...</td>
</tr>
<tr>
<td>supp:</td>
<td>7</td>
<td>25</td>
<td>15</td>
<td>23</td>
<td>3</td>
</tr>
<tr>
<td>3-itemsets:</td>
<td>{a,c,d}</td>
<td>{a,c,e}</td>
<td>{b,d,g}</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>supp:</td>
<td>15</td>
<td>22</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4-itemsets:</td>
<td>{a,c,d,e}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>supp:</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Apriori Algorithm:

Input: Matrix M

$L_1=\{\text{frequent 1-itemsets; } i \text{ such that } \text{Supp}(i) \geq \theta\}$.

For $k = 2$, while $L_{k-1} \neq \emptyset$ (while there are large $k - 1$-itemsets), $k ++$

- $C_k = \text{apriori\_gen}(L_{k-1})$ generate candidate itemsets of size $k$
- $L_k = \{c : c \in C_k, \text{Supp}(c) \geq \theta\}$ frequent itemsets of size $k$ (loop over transactions, scan the database)

end

Output: $\bigcup_k L_k$. 

---

3
The subroutine `apriori_gen` joins $L_{k-1}$ to $L_{k-1}$.

**apriori_gen Subroutine:**

**Input:** $L_{k-1}$

Find all pairs of itemsets in $L_{k-1}$ where the first $k - 2$ items are identical.

Union them (lexicographically) to get $C_k^{too big}$,

$$\text{e.g., } \{a, b, c, d, e, f\}, \{a, b, c, d, e, g\} \rightarrow \{a, b, c, d, e, f, g\}$$

**Prune:** $C_k = \{c \in C_k^{too big}, \text{all } (k - 1)\text{-subsets } c_s \text{ of } c \text{ obey } c_s \in L_{k-1}\}$.

**Output:** $C_k$.

Example of Prune step: consider $\{a, b, c, d, e, f, g\}$ which is in $C_k^{too big}$, and I want to know whether it’s in $C_k$. Look at $\{a, b, c, d, e, f, g\}, \{a, b, c, d, e, f, g\}, \{a, b, c, d, e, f, g\}$, $\{a, b, c, d, e, f, g\}$, etc. If any are not in $L_6$, then prune $\{a, b, c, d, e, f, g\}$ from $L_7$.

---

**The Apriori Algorithm**

<table>
<thead>
<tr>
<th>Database D</th>
<th>C1</th>
<th>L1</th>
</tr>
</thead>
<tbody>
<tr>
<td>TID</td>
<td>Items</td>
<td>Item Set</td>
</tr>
<tr>
<td>100</td>
<td>1 3 4</td>
<td>{1}</td>
</tr>
<tr>
<td>200</td>
<td>2 3 5</td>
<td>{2}</td>
</tr>
<tr>
<td>300</td>
<td>1 2 3 5</td>
<td>{3}</td>
</tr>
<tr>
<td>400</td>
<td>2 5</td>
<td>{4}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{5}</td>
</tr>
</tbody>
</table>

Scan D

**C2**

<table>
<thead>
<tr>
<th>Item Set</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>

Scan D

**C3**

<table>
<thead>
<tr>
<th>Item Set</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>{2 3 5}</td>
<td>2</td>
</tr>
</tbody>
</table>

| Note: {1 2, 3} {1 2, 5} and {1 3, 5} not in C3. |
Apriori scans the database at most how many times?

• Huge number of candidate sets. ☺
• Spawned huge number of apriori-like papers.

What do you do with the rules after they’re generated?

• Information overload (give up)
• Order rules by “interestingness”
  
  – Confidence
    \[ \hat{P}(b|a) = \frac{\text{Supp}(a \cup b)}{\text{Supp}(a)} \]
  
  – “Lift” / “Interest”
    \[ \frac{\hat{P}(b|a)}{\hat{P}(b)} = \frac{\text{Supp}(b)}{1 - \frac{\text{Supp}(a \cup b)}{\text{Supp}(a)}} \]
    
  :$

  – Hundreds!

Research questions:

• mining more than just itemsets (e.g, sequences, trees, graphs)
• incorporating taxonomy in items
• boolean logic and “logical analysis of data”
• Cynthia’s questions: Can we use rules within ML to get good predictive models?