Data Mining I

Summer semester 2019

Lecture 3: Frequent Itemsets and Association Rules Mining

Lectures: Prof. Dr. Eirini Ntoutsi

TAs: Tai Le Quy, Vasileios Iosifidis, Maximilian Idahl, Shaheer Asghar, Wazed Ali
Recap from previous lecture

- Data preprocessing
- Decomposing a dataset: instances and features/attributes, the class attribute
- Basic feature types
  - binary
  - categorical (nominal, ordinal)
  - numeric (interval-scale, ratio-scale)
- Basic data descriptors
  - Uni-variate
    - Measures of central tendency (mean, median, mode) and of spread (variance, standard deviation); Visual tools (histograms, boxplots)
  - Bi-variate
    - Correlation coefficient (for numerical variables)
    - Chi-square (for categorical variables)

What we didn’t cover from last lecture slides
- The material on proximity (similarity, distance) measures and normalization
- We will discuss it later in the lecture, in the clustering part
Consider the nominal feature $\text{haircolor}=$\{black, brown, red, blond\}. Consider a DM algorithm that works only with numerical values.

How can we convert the categorical values to numerical values?

Option 1: Integer encoding
- Convert the haircolor values into \{1,2,3,4\} values

Option 2: Convert into binary variables, known as one-hot encoding
- 4 new binary variables: is_it_black, is_it_brown, is_it_red, is_it_blond

Which approach is better and why?
Correlation is not causation

- In the last lecture we talked about correlation between two variables
- Correlation does not imply causation
  - Correlation shows how strongly the pair of variables are linearly related and change together
  - Causation is when one variable affects the other, i.e., any change in the value of one variable will cause a change in the value of the other variable
- Consider underlying factors before conclusion
  - In our example, the weather

Source: http://www.blog44.ca/kaiz/2018/06/19/correlation-is-not-causation/
Outline

- Introduction
- Basic concepts
- Frequent Itemsets Mining (FIM) – Apriori
- Association Rules Mining
  - Apriori improvements
  - Closed Frequent Itemsets (CFI) & Maximal Frequent Itemsets (MFI)
- Homework/tutorial
- Things you should know from this lecture
Introduction

- Frequent patterns are patterns that appear frequently in a dataset.
  - Patterns: items, substructures, subsequences ...
- Typical example: Market basket analysis

![Market basket analysis example]

- We want to know: What products were often purchased together?
  - e.g., gin and tonic?
  - e.g.: beer and diapers?
- Applications:
  - Improving Store layout, Sales campaigns, Cross-marketing, Advertising

<table>
<thead>
<tr>
<th>Tid</th>
<th>Transaction items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Butter, Bread, Milk, Sugar</td>
</tr>
<tr>
<td>2</td>
<td>Butter, Flour, Milk, Sugar</td>
</tr>
<tr>
<td>3</td>
<td>Butter, Eggs, Milk, Salt</td>
</tr>
<tr>
<td>4</td>
<td>Eggs</td>
</tr>
<tr>
<td>5</td>
<td>Butter, Flour, Milk, Salt, Sugar</td>
</tr>
</tbody>
</table>

The parable of the beer and diapers: [http://www.theregister.co.uk/2006/08/15/beer_diapers/](http://www.theregister.co.uk/2006/08/15/beer_diapers/)
Applications beyond marked basket data

- Market basket analysis
  - Items are the products, transactions are the products bought by a customer during a supermarket visit
  - Example: \{“Diapers”\} \(\rightarrow\) \{“Beer”\} (0.5%, 60%)
- Similarly in an online shop, e.g. Amazon
  - Example: \{“Computer”\} \(\rightarrow\) \{“MS office”\} (50%, 80%)
- University library
  - Items are the books, transactions are the books borrowed by a student during the semester
  - Example: \{“Kumar book”\} \(\rightarrow\) \{“Weka book”\} (60%, 70%)
- University
  - Items are the courses, transactions are the courses that are chosen by a student
  - Example: \{“Algorithms”, “DB”\} \(\rightarrow\) \{“DM”\} (10%, 75%)
- ... and many other applications.

Also, frequent pattern mining is fundamental in other DM tasks, e.g., high dimensional clustering.
Outline

- Introduction
- Basic concepts
  - Frequent Itemsets Mining (FIM) – Apriori
  - Association Rules Mining
    - Apriori improvements
    - Closed Frequent Itemsets (CFI) & Maximal Frequent Itemsets (MFI)
- Homework/tutorial
- Things you should know from this lecture
Basic concepts: Items, itemsets and transactions 1/2

- **Items \( I \):** the set of items \( I = \{i_1, ..., i_m\} \)
  - e.g., products in a supermarket, books in a bookstore

- **Itemset \( X \):** A subset of items \( X \subseteq I \)
  - e.g., \{Butter, Bread, Milk, Sugar\}, \{Butter, Bread\}

- **Itemset size:** the number of items in the itemset

- **\( k \)-Itemset:** an itemset of size \( k \)
  - e.g., \{Butter, Bread, Milk, Sugar\} is a 4-Itemset, \{Butter, Bread\} is a 2-Itemset

- **Transaction \( T \):** \( T = (tid, X_T) \)
  - e.g., products bought during a customer visit to the supermarket

- **Database DB:** A set of transactions \( T \)
  - e.g., customers purchases in a supermarket during the last week

- **Convention:** Items in transactions or itemsets are lexicographically ordered
  - Itemset \( X = (x_1, x_2, ..., x_k) \), such as \( x_1 \leq x_2 \leq ... \leq x_k \)

### Table: Transaction Items

<table>
<thead>
<tr>
<th>Tid</th>
<th>Transaction items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Butter, Bread, Milk, Sugar</td>
</tr>
<tr>
<td>2</td>
<td>Butter, Flour, Milk, Sugar</td>
</tr>
<tr>
<td>3</td>
<td>Butter, Eggs, Milk, Salt</td>
</tr>
<tr>
<td>4</td>
<td>Eggs</td>
</tr>
<tr>
<td>5</td>
<td>Butter, Flour, Milk, Salt, Sugar</td>
</tr>
</tbody>
</table>
Basic concepts: Items, itemsets and transactions 2/2

Let X be an itemset.

- **Itemset cover**: the set of transactions containing X:
  \[
  \text{cover}(X) = \{\text{tid} \mid (\text{tid}, X_T) \in DB, X \subseteq X_T\}
  \]

- **(absolute) Support**/support count of X: # transactions containing X
  \[
  \text{supportCount}(X) = |\text{cover}(X)|
  \]

- **(relative) Support** of X: fraction of transactions containing X (or the probability that a transaction contains X)
  \[
  \text{support}(X) = P(X) = \frac{\text{supportCount}(X)}{|DB|}
  \]

- **Frequent itemset**: An itemset X is frequent in DB if its support is no less than a \(\text{minSupport}\) threshold \(s\):
  \[
  \text{support}(X) \geq s
  \]

- **\(L_k\)**: the set of frequent \(k\)-itemsets
  - \(L\) comes from “Large” ("large itemsets"), another term for “frequent itemsets”

---

<table>
<thead>
<tr>
<th>Tid</th>
<th>Transaction items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Butter, Bread, Milk, Sugar</td>
</tr>
<tr>
<td>2</td>
<td>Butter, Flour, Milk, Sugar</td>
</tr>
<tr>
<td>3</td>
<td>Butter, Eggs, Milk, Salt</td>
</tr>
<tr>
<td>4</td>
<td>Eggs</td>
</tr>
<tr>
<td>5</td>
<td>Butter, Flour, Milk, Salt, Sugar</td>
</tr>
</tbody>
</table>
Example: Itemsets

- \( I = \{\text{Butter, Bread, Eggs, Flour, Milk, Salt, Sugar}\} \)

<table>
<thead>
<tr>
<th>Tid</th>
<th>Transaction items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Butter, Bread, Milk, Sugar</td>
</tr>
<tr>
<td>2</td>
<td>Butter, Flour, Milk, Sugar</td>
</tr>
<tr>
<td>3</td>
<td>Butter, Eggs, Milk, Salt</td>
</tr>
<tr>
<td>4</td>
<td>Eggs</td>
</tr>
<tr>
<td>5</td>
<td>Butter, Flour, Milk, Salt, Sugar</td>
</tr>
</tbody>
</table>

**Itemset cover**: the set of transactions containing \( X \):
\[
\text{cover}(X) = \{\text{tid} \mid (\text{tid}, X_T) \in DB, X \subseteq X_T\}
\]

(absolute) **Support**: support count of \( X \): # transactions containing \( X \)
\[
\text{supportCount}(X) = |\text{cover}(X)|
\]

- What is the cover and support of \( \{\text{Butter}\} \)?
- What is the cover and support of \( \{\text{Butter, Bread}\} \)?
- What is the cover and support of \( \{\text{Butter, Flour}\} \)?
- What is the cover and support of \( \{\text{Butter, Milk, Sugar}\} \)?
Example: Itemsets

- \( I = \{\text{Butter, Bread, Eggs, Flour, Milk, Salt, Sugar}\} \)

<table>
<thead>
<tr>
<th>Tid</th>
<th>Transaction items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Butter, Bread, Milk, Sugar</td>
</tr>
<tr>
<td>2</td>
<td>Butter, Flour, Milk, Sugar</td>
</tr>
<tr>
<td>3</td>
<td>Butter, Eggs, Milk, Salt</td>
</tr>
<tr>
<td>4</td>
<td>Eggs</td>
</tr>
<tr>
<td>5</td>
<td>Butter, Flour, Milk, Salt, Sugar</td>
</tr>
</tbody>
</table>

- \( \text{support}(\text{Butter}) = \frac{4}{5}=80\% \)
  - \( \text{cover}(\text{Butter}) = \{1,2,3,4\} \)
- \( \text{support}(\text{Butter, Bread}) = \frac{1}{5}=20\% \)
  - \( \text{cover}(\text{Butter, Bread}) = \ldots \)
- \( \text{support}(\text{Butter, Flour}) = \frac{2}{5}=40\% \)
  - \( \text{cover}(\text{Butter, Flour}) = \ldots \)
- \( \text{support}(\text{Butter, Milk, Sugar}) = \frac{3}{5}=60\% \)
  - \( \text{Cover}(\text{Butter, Milk, Sugar}) = \ldots \)

**Itemset cover:** the set of transactions containing \( X \):
\[
\text{cover}(X) = \{\text{tid} \mid (\text{tid}, X) \in DB, X \subseteq X_T\}
\]

(absolute) Support/ support count of \( X \): # transactions containing \( X \)
\[
\text{supportCount}(X) = |\text{cover}(X)|
\]
The Frequent Itemsets Mining (FIM) problem

- **Given:**
  - A set of items $I$
  - A transactions database $DB$ over $I$
  - A $minSupport$ threshold $s$

- **Goal:** Find all frequent itemsets in $DB$, i.e.:
  \[
  \{X \subseteq I \mid support(X) \geq s\}
  \]

Data Mining I @SS19, Lecture 3: Frequent Itemsets and Association Rules Mining
Basic concepts: association rules, support, confidence

Let $X$, $Y$ be two itemsets: $X, Y \subseteq I$ and $X \cap Y = \emptyset$.

- **Association rules**: rules of the form
  
  \[
  X \rightarrow Y
  \]

  head or LHS (left-hand side) or antecedent of the rule

  \[
  \rightarrow
  \]

  body or RHS (right-hand side) or consequent of the rule

- **Support $s$** of a rule: the percentage of transactions containing $X \cup Y$ in the DB or the probability $P(X \cap Y)$

  \[
  \text{support}(X \rightarrow Y) = \text{support}(X \cup Y) = P(X \cap Y)
  \]

- **Confidence $c$** of a rule: the percentage of transactions containing $X \cup Y$ in the set of transactions containing $X$. Or, in other words the conditional probability that a transaction containing $X$ also contains $Y$

  \[
  \text{confidence}(X \rightarrow Y) = \frac{\text{support}(X \cup Y)}{\text{support}(X)} = \frac{P(X \cap Y)}{P(X)} = P(Y | X)
  \]

- Support and confidence are measures of rules’ interestingness.

- Rules are usually written as follows: $X \rightarrow Y$ (support, confidence)

Data Mining I @SS19, Lecture 3: Frequent Itemsets and Association Rules Mining
Example: association rules

- \( I = \{\text{Butter, Bread, Eggs, Flour, Milk, Salt, Sugar}\} \)

<table>
<thead>
<tr>
<th>Tid</th>
<th>Transaction items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Butter, Bread, Milk, Sugar</td>
</tr>
<tr>
<td>2</td>
<td>Butter, Flour, Milk, Sugar</td>
</tr>
<tr>
<td>3</td>
<td>Butter, Eggs, Milk, Sugar</td>
</tr>
<tr>
<td>4</td>
<td>Eggs</td>
</tr>
<tr>
<td>5</td>
<td>Butter, Flour, Milk, Salt, Sugar</td>
</tr>
</tbody>
</table>

What is the support and confidence of \( \{\text{Butter}\} \rightarrow \{\text{Bread}\} \)?

What is the support and confidence of \( \{\text{Butter, Milk}\} \rightarrow \{\text{Sugar}\} \)?

support(\(X\rightarrow Y\)) = support(\(X \cup Y\)) = P(X \cap Y)

\[\text{confidence}(X \rightarrow Y) = P(Y|X) = \frac{P(X \cap Y)}{P(X)} = \frac{\text{support}(X \cup Y)}{\text{support}(X)}\]
Example: association rules

I = {Butter, Bread, Eggs, Flour, Milk, Salt, Sugar}

<table>
<thead>
<tr>
<th>Tid</th>
<th>Transaction items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Butter, Bread, Milk, Sugar</td>
</tr>
<tr>
<td>2</td>
<td>Butter, Flour, Milk, Sugar</td>
</tr>
<tr>
<td>3</td>
<td>Butter, Eggs, Milk, Salt</td>
</tr>
<tr>
<td>4</td>
<td>Eggs</td>
</tr>
<tr>
<td>5</td>
<td>Butter, Flour, Milk, Salt, Sugar</td>
</tr>
</tbody>
</table>

- \{Butter\} → \{Bread\} (20%, 25%)
  - support(Butter ∪ Bread) = 1/5 = 20%
  - support(Butter) = 4/5 = 80%
  - confidence = 20%/80% = 1/4 = 25%

- \{Butter, Milk\} → \{Sugar\} (60%, 75%)
  - support(Butter, Milk ∪ Sugar) = 3/5 = 60%
  - support(Butter, Milk) = 4/5 = 80%
  - confidence = 60%/80% = 3/4 = 75%

support(\(X\)→\(Y\)) = support(\(X \cup Y\)) = \(P(X \cap Y)\)

certainty(\(X\)→\(Y\)) = \(P(Y|X) = \frac{P(X \cap Y)}{P(X)} = \frac{support(X \cup Y)}{support(X)}\)
The Association Rules Mining (ARM) problem

- Given:
  - A set of items \( I \)
  - A transactions database \( DB \) over \( I \)
  - A \textit{minSupport} threshold \( s \) and a \textit{minConfidence} threshold \( c \)

- Goal: Find all association rules \( X \rightarrow Y \) in \( DB \) w.r.t. minimum support \( s \) and minimum confidence \( c \), i.e.:
  \[
  \{X \rightarrow Y | support(X \cup Y) \geq s, \text{confidence}(X \rightarrow Y) \geq c\}
  \]

- These rules are called \textit{strong}. 

<table>
<thead>
<tr>
<th>transactionID</th>
<th>items</th>
<th>support ( s \geq 25% )</th>
<th>confidence ( c \geq 50% )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>A,B,C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td>A,C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4000</td>
<td>A,D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5000</td>
<td>B,E,F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Association rules:
- \( A \Rightarrow C \) (support = 50\%, confidence= 66.6\%)
- \( C \Rightarrow A \) (support = 50\%, confidence= 100\%)
- ...
Solving the problems

- **Problem 1 (FIM):** Find all frequent itemsets in $DB$, i.e.: $\{X \subseteq I \mid \text{support}(X) \geq s\}$

- **Problem 2 (ARM):** Find all association rules $X \rightarrow Y$ in $DB$, w.r.t. min support $s$ and min confidence $c$, i.e.,: $\{X \rightarrow Y \mid \text{support}(X \cup Y) \geq s, \text{confidence}(X \rightarrow Y) \geq c, X,Y \subseteq I \text{ and } X \cap Y = \emptyset\}$

- Problem 1 is part of Problem 2.

- 2-step method to extract the association rules:
  - Step 1: Determine the frequent itemsets w.r.t. min support $s$
  - Step 2: Generate the association rules w.r.t. min confidence $c$

- Step 1 (FIM) is the most costly, so the overall performance of an association rules mining algorithm is determined by this step.
A naïve solution to the FIM problem

- Naïve solution: count the frequencies for all $k$-itemsets, $k=1 \ldots |I|$
- The number of itemsets can be really huge.
- Let us consider a small set of items: $I = \{A,B,C,D\}$

# 1-itemsets: \[
\binom{4}{1} = \frac{4!}{(4-1)!*1!} = \frac{4!}{3!} = 4
\]

# 2-itemsets: \[
\binom{4}{2} = \frac{4!}{(4-2)!*2!} = \frac{4!}{2!*2!} = 6
\]

# 3-itemsets: \[
\binom{4}{3} = \frac{4!}{(4-3)!*3!} = \frac{4!}{3!} = 4
\]

# 4-itemsets: \[
\binom{4}{4} = \frac{4!}{(4-4)!*4!} = 1
\]
The exponential itemsets search space

- In the general case, for all $k$-itemsets, $k=1 \ldots |I|$, all combinations of $1\ldots|I|$ size itemsets:

$$\binom{|I|}{1} + \binom{|I|}{2} + \ldots + \binom{|I|}{|I|} = 2^{|I|} - 1$$

- The itemset search space is exponential
  - Search space: all itemsets that can be formed with items in $I$
  - This forms a lattice (itemsets lattice)

- So, generating all possible combinations and computing their support is inefficient!
  - Solution: Apriori and its variants!
Outline

- Introduction
- Basic concepts
- Frequent Itemsets Mining (FIM) – Apriori
- Association Rules Mining
  - Apriori improvements
  - Closed frequent itemsets (CFI) & Maximal frequent itemsets (MFI)
- Homework/tutorial
- Things you should know from this lecture
Apriori algorithm [Agrawal & Srikant @VLDB’94]

- **Idea**: First determine frequent 1-itemsets, then frequent 2-itemsets and so on

  - **Method overview**:
    - Initially, scan $DB$ once to get frequent 1-itemset $L_1$
    - Generate length $(k+1)$ candidate itemsets $C_{k+1}$ from length $k$ frequent itemsets $L_k$
    - Evaluate whether the candidates $C_{k+1}$ are really frequent, query the DB → frequent $(k+1)$-itemsets $L_{k+1}$
    - Terminate when no frequent or candidate set can be generated

To reduce complexity, the set of candidates should be as small as possible!!!

level-wise or breadth-first search exploration of the itemset search space
Let $X$ be an itemset. According to the Apriori property or Downward closure property or Monotonicity property:

- If $X$ is frequent, all its subsets $Y \subseteq X$ are also frequent.
  - e.g., if $\{\text{beer, diaper, nuts}\}$ is frequent, so is $\{\text{beer, diaper}\}$
  - i.e., every transaction having $\{\text{beer, diaper, nuts}\}$ also contains $\{\text{beer, diaper}\}$

- Inversing: When $X$ is not frequent, all its supersets are not frequent and thus they should not be generated/ tested!
  - e.g., if $\{\text{beer, diaper}\}$ is not frequent, $\{\text{beer, diaper, nuts}\}$ would not be frequent also

- The inverse property helps us to reduce the candidate itemsets set by pruning non-promising itemsets
Illustration of the Apriori property

If \( X = \{c, d, e\} \) is frequent then all its subsets are frequent.
Illustration of the Apriori property

If $X = \{c, d, e\}$ is not frequent then all its supersets are not frequent.
Let us consider the following transaction database

Transaction Database
1, {Chips, Pizza}
2, {Beer, Chips}
3, {Chips, Pizza, Wine}
4, {Wine}

and a minSupport threshold \( \text{minSupp} = 2 \)
Search space pruning via the Apriori property: an example

Transaction Database
1, {Chips, Pizza}
2, {Beer, Chips}
3, {Chips, Pizza, Wine}
4, {Wine}

\[ \text{minSupp} = 2 \]

**Note:**
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

Transaction Database
1, \{Chips, Pizza\}
2, \{Beer, Chips\}
3, \{Chips, Pizza, Wine\}
4, \{Wine\}

\(\text{minSupp} = 2\)

Note:
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

Transaction Database
1, \{Chips, Pizza\}
2, \{Beer, Chips\}
3, \{Chips, Pizza, Wine\}
4, \{Wine\}

\(\text{minSupp} = 2\)

Note:
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

Transaction Database
1, {Chips, Pizza}
2, {Beer, Chips}
3, {Chips, Pizza, Wine}
4, {Wine}

\( \text{minSupp} = 2 \)

\[
\begin{align*}
\{\text{Beer}\}:1 & \\
\{\text{Chips}\}:3 & \\
\{\text{Pizza}\}:? & \\
\{\text{Wine}\}:? & \\
\{\text{Beer, Chips}\}:? & \\
\{\text{Beer, Pizza}\}:? & \\
\{\text{Beer, Wine}\}:? & \\
\{\text{Chips, Pizza}\}:? & \\
\{\text{Chips, Wine}\}:? & \\
\{\text{Pizza, Wine}\}:? & \\
\{\text{Beer, Chips, Pizza}\}:? & \\
\{\text{Beer, Chips, Wine}\}:? & \\
\{\text{Beer, Pizza, Wine}\}:? & \\
\{\text{Chips, Pizza, Wine}\}:? & \\
\{\text{Beer, Chips, Pizza, Wine}\}:? & \\
\end{align*}
\]

Note:
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

Transaction Database
1, {Chips, Pizza}
2, {Beer, Chips}
3, {Chips, Pizza, Wine}
4, {Wine}

\( \text{minSupp} = 2 \)

Note:
We don’t generate the lattice in advance … this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

Transaction Database
1, {Chips, Pizza}
2, {Beer, Chips}
3, {Chips, Pizza, Wine}
4, {Wine}

\[ \text{minSupp} = 2 \]

**Note:**
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

Transaction Database
1, \{Chips, Pizza\}
2, \{Beer, Chips\}
3, \{Chips, Pizza, Wine\}
4, \{Wine\}

\textit{Note:}
We don’t generate the lattice in advance
... this is to showcase the pruning effect

\(\text{minSupp} = 2\)
Search space pruning via the Apriori property: an example

Transaction Database
1. {Chips, Pizza}
2. {Beer, Chips}
3. {Chips, Pizza, Wine}
4. {Wine}

\[ \text{minSupp} = 2 \]

\[
\begin{align*}
\{\text{Beer}\}:1 & \quad \{\text{Chips}\}:3 & \quad \{\text{Pizza}\}:2 & \quad \{\text{Wine}\}:2 \\
\{\text{Chips, Pizza}\}:? & \quad \{\text{Chips, Wine}\}:? & \quad \{\text{Pizza, Wine}\}:? & \quad \{\text{Chips, Pizza, Wine}\}:? \\
\end{align*}
\]

Note:
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

**Transaction Database**
1. {Chips, Pizza}
2. {Beer, Chips}
3. {Chips, Pizza, Wine}
4. {Wine}

\[ \text{minSupp} = 2 \]

**Note:**
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

Transaction Database
1. {Chips, Pizza}
2. {Beer, Chips}
3. {Chips, Pizza, Wine}
4. {Wine}

\( \text{minSupp} = 2 \)

Note:
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

**Transaction Database**
1. \{Chips, Pizza\}
2. \{Beer, Chips\}
3. \{Chips, Pizza, Wine\}
4. \{Wine\}

\[\text{minSupp} = 2\]

**Note:**
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Search space pruning via the Apriori property: an example

Transaction Database
1, {Chips, Pizza}
2, {Beer, Chips}
3, {Chips, Pizza, Wine}
4, {Wine}

\[
\text{minSupp} = 2
\]
Search space pruning via the Apriori property: an example

Transaction Database
1, {Chips, Pizza}
2, {Beer, Chips}
3, {Chips, Pizza, Wine}
4, {Wine}

\(\text{minSupp} = 2\)

\[
\begin{align*}
{} & 4 \\
{} & {\{\text{Chips,Pizza}\}}:2 \quad {\{\text{Chips,Wine}\}}:1 \quad {\{\text{Pizza,Wine}\}}:1 \\
{\{\text{Beer}\}}:1 & \\
{\{\text{Chips}\}}:3 & \\
{\{\text{Pizza}\}}:2 & \\
{\{\text{Wine}\}}:2 & \\
\end{align*}
\]

**Note:**
We don’t generate the lattice in advance
... this is to showcase the pruning effect
Border itemsets

- Border itemsets $X$: all subsets $Y \subseteq X$ are frequent, all supersets $Z \supset X$ are not frequent

The idea of many algorithms is to not explore beyond the border
Border itemsets

- **Border itemsets** $X$: all subsets $Y \subseteq X$ are frequent, all supersets $Z \supseteq X$ are not frequent

The border depends on the `minSupport` threshold

```plaintext
minSupport \text{ s } = 1
```
How these concepts apply to the Apriori algorithm?

- **Apriori**: level-wise or breadth-first search exploration of the itemsets search space

  ![Diagram of Apriori algorithm]

  ![Diagram of Apriori algorithm]

- **Method overview**:
  - Initially, scan DB once to get frequent 1-itemset $L_1$
  - Generate length $(k+1)$ candidate itemsets $C_{k+1}$ from length $k$ frequent itemsets $L_k$
  - Evaluate whether the candidates $C_{k+1}$ are really frequent, query the DB $\rightarrow$ frequent $(k+1)$-itemsets $L_{k+1}$
  - Terminate when no frequent or candidate set can be generated
A 2-step process: from $L_{k-1}$ to $L_k$

Notation: $L_k$: frequent itemsets of size $k$; $C_k$: candidate itemsets of size $k$

- **Join step**: generate candidates $C_k$
  - $L_k$ is generated by self-joining $L_{k-1}$: $L_{k-1} \times L_{k-1}$
  - Two ($k$-1)-itemsets $p$, $q$ are joined, if they agree in their prefix (i.e., first ($k$-$2$) items)

- **Prune step**: prune $C_k$ and return $L_k$
  - $C_k$ is superset of $L_k$
  - Naïve idea: count the support for all candidate itemsets in $C_k$ ....but this requires DB access
  - **Prune by Apriori** property: a candidate $k$-itemset that has some non-frequent ($k$-1)-itemset cannot be frequent
    - Prune all those $k$-itemsets, that have some ($k$-1)-subset that is not frequent (i.e. does not belong to $L_{k-1}$)
    - Due to the level-wise approach of Apriori, we only need to check ($k$-1)-subsets
  - For the remaining itemsets in $C_k$, **prune by support count (DB)**

Example:
Input: $L_3\{abc, abd, acd, ace, bcd\}$
Output: $L_4 = ?$
  - Join step: ?
  - Prune steps
    - Apriori-based?
    - DB-based?
Example

- Let $L_3 = \{abc, abd, acd, ace, bcd\}$
- Join step: $C_4 = L_3 \ast L_3$
  - $C_4 = \{abc\ast abd = abcd; acd\ast ace = acde\}$
- Prune step (Apriori-based):
  - $acde$ is pruned since $cde$ is not frequent
- Prune step (DB-based):
  - check $abcd$ support in the DB and prune accordingly
Apriori algorithm (pseudo-code)

C_k: Candidate itemset of size k
L_k : frequent itemset of size k

L_1 = {frequent items};
for (k = 1; L_k != ∅; k++) do begin
C_{k+1} = candidates generated from L_k;
   for each transaction t in database do
      increment the count of all candidates in C_{k+1} that are contained in t
   L_{k+1} = candidates in C_{k+1} with min_support
end
return \bigcup_k L_k;

Candidate generation (self-join, apriori property)
DB scan
subset function
Prune by support count (ask DB)

Subset function:
- The subset function must for each transaction T in DB check all candidates in the candidate set C_k whether they are part of the transaction T
- Organize candidates C_k in a hash tree
A full example of Apriori

- Given the following transaction database and a minSupport threshold \( s = 4 \), find all frequent itemsets
- Report on \( C_k, L_k \) sets as well as on how \( L_k \) was derived from \( C_k \) (Apriori pruning or support count)

### Database TDB

<table>
<thead>
<tr>
<th>Tid</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A, B, C, D, E</td>
</tr>
<tr>
<td>2</td>
<td>C, D, E, F, G</td>
</tr>
<tr>
<td>3</td>
<td>A, B, C, D</td>
</tr>
<tr>
<td>4</td>
<td>B, C, D, E</td>
</tr>
<tr>
<td>5</td>
<td>A, D, E, F</td>
</tr>
<tr>
<td>6</td>
<td>A, B, C, E</td>
</tr>
<tr>
<td>7</td>
<td>B, C, E, F</td>
</tr>
<tr>
<td>8</td>
<td>A, B, G</td>
</tr>
<tr>
<td>9</td>
<td>A, B, C, E, F</td>
</tr>
<tr>
<td>10</td>
<td>A, C, D, E</td>
</tr>
</tbody>
</table>

\textit{minSupport} \( s = 4 \)
A full example of Apriori

Database TDB

<table>
<thead>
<tr>
<th>Tid</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A, B, C, D, E</td>
</tr>
<tr>
<td>2</td>
<td>C, D, E, F, G</td>
</tr>
<tr>
<td>3</td>
<td>A, B, C, D</td>
</tr>
<tr>
<td>4</td>
<td>B, C, D, E</td>
</tr>
<tr>
<td>5</td>
<td>A, D, E, F</td>
</tr>
<tr>
<td>6</td>
<td>A, B, C, E</td>
</tr>
<tr>
<td>7</td>
<td>B, C, E, F</td>
</tr>
<tr>
<td>8</td>
<td>A, B, G</td>
</tr>
<tr>
<td>9</td>
<td>A, B, C, E, F</td>
</tr>
<tr>
<td>10</td>
<td>A, C, D, E</td>
</tr>
</tbody>
</table>

\( \text{minSupport} = 4 \)

- Green rows: prune by Apriori property
- Red rows: prune by minSupport threshold

1\textsuperscript{st} scan

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>7</td>
</tr>
<tr>
<td>{B}</td>
<td>7</td>
</tr>
<tr>
<td>{C}</td>
<td>8</td>
</tr>
<tr>
<td>{D}</td>
<td>6</td>
</tr>
<tr>
<td>{E}</td>
<td>8</td>
</tr>
<tr>
<td>{F}</td>
<td>4</td>
</tr>
<tr>
<td>{G}</td>
<td>2</td>
</tr>
</tbody>
</table>

2\textsuperscript{nd} scan

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>5</td>
</tr>
<tr>
<td>{A, C}</td>
<td>5</td>
</tr>
<tr>
<td>{A, D}</td>
<td>4</td>
</tr>
<tr>
<td>{A, E}</td>
<td>5</td>
</tr>
<tr>
<td>{A, F}</td>
<td>2</td>
</tr>
<tr>
<td>{B, C}</td>
<td>6</td>
</tr>
<tr>
<td>{B, D}</td>
<td>3</td>
</tr>
<tr>
<td>{B, E}</td>
<td>5</td>
</tr>
<tr>
<td>{B, F}</td>
<td>2</td>
</tr>
<tr>
<td>{C, D}</td>
<td>5</td>
</tr>
<tr>
<td>{C, E}</td>
<td>7</td>
</tr>
<tr>
<td>{C, F}</td>
<td>3</td>
</tr>
<tr>
<td>{D, E}</td>
<td>5</td>
</tr>
<tr>
<td>{D, F}</td>
<td>2</td>
</tr>
<tr>
<td>{E, F}</td>
<td>4</td>
</tr>
</tbody>
</table>

C\textsubscript{1} \rightarrow \boxed{L_1} \rightarrow C\textsubscript{2}
A full example of Apriori

Database TDB

<table>
<thead>
<tr>
<th>Tid</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A, B, C, D, E</td>
</tr>
<tr>
<td>2</td>
<td>C, D, E, F, G</td>
</tr>
<tr>
<td>3</td>
<td>A, B, C, D</td>
</tr>
<tr>
<td>4</td>
<td>B, C, D, E</td>
</tr>
<tr>
<td>5</td>
<td>A, D, E, F</td>
</tr>
<tr>
<td>6</td>
<td>A, B, C, E</td>
</tr>
<tr>
<td>7</td>
<td>B, C, E, F</td>
</tr>
<tr>
<td>8</td>
<td>A, B, G</td>
</tr>
<tr>
<td>9</td>
<td>A, B, C, E, F</td>
</tr>
<tr>
<td>10</td>
<td>A, C, D, E</td>
</tr>
</tbody>
</table>

minSupport = 4

- Green rows: prune by Apriori property
- Red rows: prune by minSupport threshold

- {B, D} is non frequent -> pruned by Apriori property

L2

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>5</td>
</tr>
<tr>
<td>{A, C}</td>
<td>5</td>
</tr>
<tr>
<td>{A, D}</td>
<td>4</td>
</tr>
<tr>
<td>{A, E}</td>
<td>5</td>
</tr>
<tr>
<td>{B, C}</td>
<td>6</td>
</tr>
<tr>
<td>{B, E}</td>
<td>5</td>
</tr>
<tr>
<td>{C, D}</td>
<td>5</td>
</tr>
<tr>
<td>{C, E}</td>
<td>7</td>
</tr>
<tr>
<td>{D, E}</td>
<td>5</td>
</tr>
<tr>
<td>{E, F}</td>
<td>4</td>
</tr>
</tbody>
</table>

C3

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B, C}</td>
<td>4</td>
</tr>
<tr>
<td>{A, B, D}</td>
<td></td>
</tr>
<tr>
<td>{A, B, E}</td>
<td></td>
</tr>
<tr>
<td>{A, C, D}</td>
<td></td>
</tr>
<tr>
<td>{A, C, E}</td>
<td></td>
</tr>
<tr>
<td>{A, D, E}</td>
<td></td>
</tr>
<tr>
<td>{B, C, E}</td>
<td></td>
</tr>
<tr>
<td>{C, D, E}</td>
<td></td>
</tr>
</tbody>
</table>

L3

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B, C}</td>
<td>4</td>
</tr>
<tr>
<td>{A, C, E}</td>
<td>4</td>
</tr>
<tr>
<td>{B, C, E}</td>
<td>5</td>
</tr>
<tr>
<td>{C, D, E}</td>
<td>4</td>
</tr>
</tbody>
</table>

Data Mining I @SS19, Lecture 3: Frequent Itemsets and Association Rules Mining
Apriori overview

- "Naïve" algorithm: count the frequencies for all $k$-itemsets
  - Inefficient!!! There are $O\left(\binom{|I|}{k}\right)$ such subsets
  - Total cost: $O\left(2^{|I|}\right)$
- Apriori algorithm
  - Level-wise traversal of the lattice (breadth-first search)
  - Generate-and-test strategy
    - Candidate generation step $C_k \leftarrow L_{k-1} \times L_{k-1}$
    - Pruning step $\rightarrow L_k$
      - Apriori property
      - Support-based pruning (check the DB)

How does Apriori reduce the (exponential) search space?
How many database scans are required?
Which parameters affect the complexity of the algorithm?
Apriori overview

- **Advantages:**
  - Apriori property
  - Easy implementation

- **Disadvantages:**
  - It requires up to $|I|$ database scans
  - It assumes that the DB is in memory

- **Complexity depends on**
  - $\textit{minSupport}$ threshold
  - Number of items $|I|$ (dimensionality)
  - Number of transactions $|DB|$
Outline

- Introduction
- Basic concepts
- Frequent Itemsets Mining (FIM) – Apriori
- Association Rules Mining
  - Apriori improvements
  - Closed Frequent Itemsets (CFI) & Maximal Frequent Itemsets (MFI)
- Homework/tutorial
- Things you should know from this lecture
Association Rules Mining

- (Recall the) 2-step method to extract the association rules:
  - Determine the frequent itemsets w.r.t. min support $s$
  - Generate the association rules w.r.t. min confidence $c$.

- Regarding step 2, the following method is followed:
  - For every frequent itemset $X$
  - for every subset $Y$ of $X$: $Y \neq \emptyset$, $Y \neq X$, the rule $Y \rightarrow (X - Y)$ is formed
  - Remove rules that violate min confidence $c$
  - No extra database access is required
    - Store the frequent itemsets and their supports in a hash table

Let $X=\{1,2,3\}$ be frequent
There are 6 candidate rules that can be generated from $X$:
- $\{1,2\} \rightarrow 3$
- $\{1,3\} \rightarrow 2$
- $\{2,3\} \rightarrow 1$
- $\{1\} \rightarrow \{2,3\}$
- $\{2\} \rightarrow \{1,3\}$
- $\{3\} \rightarrow \{1,2\}$

We can decide if there are strong using the support counts (already computed during the FIM step)

**Recall** $\text{confidence}(X \rightarrow Y) = \frac{\text{support}(X \cup Y)}{\text{support}(X)} = \frac{P(X \cap Y)}{P(X)} = P(Y|X)$
Pseudocode

Input:
- $D$ // Database of transactions
- $I$ // Items
- $L$ // Large itemsets
- $s$ // Support
- $\alpha$ // Confidence

Output:
- $R$ // Association Rules satisfying $s$ and $\alpha$

ARGen Algorithm:

$R = \emptyset$;

for each $l \in L$ do
  for each $x \subset l$ such that $x \neq \emptyset$ and $x \neq l$ do
    if $\frac{support(l)}{support(x)} \geq \alpha$ then
      $R = R \cup \{x \Rightarrow (l - x)\}$;
### Example

#### Transaction database

<table>
<thead>
<tr>
<th>tid</th>
<th>$X_t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{Bier, Chips, Wine}</td>
</tr>
<tr>
<td>2</td>
<td>{Bier, Chips}</td>
</tr>
<tr>
<td>3</td>
<td>{Pizza, Wine}</td>
</tr>
<tr>
<td>4</td>
<td>{Chips, Pizza}</td>
</tr>
</tbody>
</table>

#### $I = \{\text{Bier, Chips, Pizza, Wine}\}$

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Cover</th>
<th>Sup.</th>
<th>Freq.</th>
</tr>
</thead>
<tbody>
<tr>
<td>{}</td>
<td>{1,2,3,4}</td>
<td>4</td>
<td>100 %</td>
</tr>
<tr>
<td>{Bier}</td>
<td>{1,2}</td>
<td>2</td>
<td>50 %</td>
</tr>
<tr>
<td>{Chips}</td>
<td>{1,2,4}</td>
<td>3</td>
<td>75 %</td>
</tr>
<tr>
<td>{Pizza}</td>
<td>{3,4}</td>
<td>2</td>
<td>50 %</td>
</tr>
<tr>
<td>{Wine}</td>
<td>{1,3}</td>
<td>2</td>
<td>50 %</td>
</tr>
<tr>
<td>{Bier, Chips}</td>
<td>{1,2}</td>
<td>2</td>
<td>50 %</td>
</tr>
<tr>
<td>{Bier, Wine}</td>
<td>{1}</td>
<td>1</td>
<td>25 %</td>
</tr>
<tr>
<td>{Chips, Pizza}</td>
<td>{4}</td>
<td>1</td>
<td>25 %</td>
</tr>
<tr>
<td>{Chips, Wine}</td>
<td>{1}</td>
<td>1</td>
<td>25 %</td>
</tr>
<tr>
<td>{Pizza, Wine}</td>
<td>{3}</td>
<td>1</td>
<td>25 %</td>
</tr>
<tr>
<td>{Bier, Chips, Wine}</td>
<td>{1}</td>
<td>1</td>
<td>25 %</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rule</th>
<th>Sup.</th>
<th>Freq.</th>
<th>Conf.</th>
</tr>
</thead>
<tbody>
<tr>
<td>{Bier} ⇒ {Chips}</td>
<td>2</td>
<td>50 %</td>
<td>100 %</td>
</tr>
<tr>
<td>{Bier} ⇒ {Wine}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
<tr>
<td>{Chips} ⇒ {Bier}</td>
<td>2</td>
<td>50 %</td>
<td>66 %</td>
</tr>
<tr>
<td>{Pizza} ⇒ {Chips}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
<tr>
<td>{Pizza} ⇒ {Wine}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
<tr>
<td>{Wine} ⇒ {Bier}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
<tr>
<td>{Wine} ⇒ {Chips}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
<tr>
<td>{Wine} ⇒ {Pizza}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
<tr>
<td>{Bier, Chips} ⇒ {Wine}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
<tr>
<td>{Bier, Wine} ⇒ {Chips}</td>
<td>1</td>
<td>25 %</td>
<td>100 %</td>
</tr>
<tr>
<td>{Chips, Wine} ⇒ {Bier}</td>
<td>1</td>
<td>25 %</td>
<td>100 %</td>
</tr>
<tr>
<td>{Bier} ⇒ {Chips, Wine}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
<tr>
<td>{Wine} ⇒ {Bier, Chips}</td>
<td>1</td>
<td>25 %</td>
<td>50 %</td>
</tr>
</tbody>
</table>
Interesting and misleading association rules

Example:

- Database on the behavior of students in a school with 5,000 students
- We extract the following itemsets:
  - 60% of the students play Soccer,
  - 75% of the students eat chocolate bars
  - 40% of the students play Soccer and eat chocolate bars
- Association rules: \{“Play Soccer”\} → \{“Eat chocolate bars”\}, confidence = 40%/60% = 67%
  - The rule has a high confidence, however:
    - \{“Eat chocolate bars”\}, support = 75%, regardless of whether they play soccer.
  - Thus, knowing that one is playing soccer decreases his/her probability of eating chocolate (from 75% → 67%)!
  - Therefore, the rule \{“Play Soccer”\} → \{“Eat chocolate bars”\} is misleading despite its high confidence
Evaluating Association Rules 2/2

Task: Filter out misleading rules

Let \( \{A\} \rightarrow \{B\} \)

- Measure of “interestingness” of a rule: interest
  \[
  \frac{P(\{A \cap B\})}{P(A)} - P(B)
  \]
  - the higher the value the more interesting the rule is

- Measure of dependent/correlated events: lift
  \[
  \text{lift} = \frac{P(\{A \cap B\})}{P(A)P(B)} = \frac{\sup \text{ port}(\{A \cup B\})}{\sup \text{ port}(A)\sup \text{ port}(B)}
  \]
  - the ratio of the observed support to that expected if \( X \) and \( Y \) were independent.
Measuring Quality of Association Rules: overview of different measures

For a rule $A \rightarrow B$

- **Support**
  
  $P(A \cap B)$
  
  - e.g. support($milk$, $bread$, $butter$)=20%, i.e. 20% of the transactions contain these

- **Confidence**
  
  $\frac{P(A \cap B)}{P(A)}$
  
  - e.g. confidence($milk$, $bread \rightarrow butter$)=50%, i.e. 50% of the times a customer buys milk and bread, butter is bought as well.

- **Lift**
  
  $\frac{P(A \cap B)}{P(A)P(B)}$
  
  - e.g. lift($milk$, $bread \rightarrow butter$)=20%/(40%*40%)=1.25. the observed support is 20%, the expected (if they were independent) is 16%. 
Outline

- Introduction
- Basic concepts
- Frequent Itemsets Mining (FIM) – Apriori
- Association Rules Mining
  - Apriori improvements
  - Closed Frequent Itemsets (CFI) & Maximal Frequent Itemsets (MFI)
- Homework/tutorial
- Things you should know from this lecture
Apriori improvements

- Major computational challenges in Apriori:
  - Multiple scans of the DB: For each step $k$ (i.e., $k$-itemsets), a database scan is required
  - Huge number of candidates
  - Tedious workload of support counting for candidates
    - Too many candidates; One transaction may contain many candidates.

- Improving Apriori directions:
  - Reduce passes of transaction database scans
  - Shrink number of candidates
  - Facilitate support counting of candidates
  - Different algorithms: FPGrowth, Partition, Sampling, ECLAT
Horizontal vs vertical transaction representation

- **Horizontal layout:** \((TID, \text{item set})\)
- **Vertical layout:** \((\text{item, TID set})\)

### Horizontal Data Layout

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A, B, E</td>
</tr>
<tr>
<td>2</td>
<td>B, C, D</td>
</tr>
<tr>
<td>3</td>
<td>C, E</td>
</tr>
<tr>
<td>4</td>
<td>A, C, D</td>
</tr>
<tr>
<td>5</td>
<td>A, B, C, D</td>
</tr>
<tr>
<td>6</td>
<td>A, E</td>
</tr>
<tr>
<td>7</td>
<td>A, B</td>
</tr>
<tr>
<td>8</td>
<td>A, B, C</td>
</tr>
<tr>
<td>9</td>
<td>A, C, D</td>
</tr>
<tr>
<td>10</td>
<td>B</td>
</tr>
</tbody>
</table>

### Vertical Data Layout

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TID-list**
Eclat (Zaki, TKDE’00)

- Vertical data layout
- For each itemset $X$, a list of the transaction *ids* that contain $X$ is maintained:
  - $X.tidlist = \{t_1, t_4, t_5, t_7, t_8, t_9\}$
  - $Y.tidlist = \{t_1, t_2, t_5, t_7, t_8, t_{10}\}$
- To find the support of $\{XY\}$, we use their lists intersection:
  - $X.tidlist \cap Y.tidlist = \{t_1, t_5, t_7, t_8\}$
  - $support(XY) = |X.tidlist \cap Y.tidlist| = 4$
Example ECLAT

Data Mining I @SS19, Lecture 3: Frequent Itemsets and Association Rules Mining
Eclat (Zaki, TKDE’00)

- Advantage:
  - No need to access the DB (use instead lists intersection)
  - Very fast support counting (using the lists intersection)
    - As we proceed, the size of the lists decreases, so intersection computation is faster

- Disadvantage:
  - Intermediate tid-lists may become too large for memory
FPGrowth (Han, Pei & Yin, SIGMOD’00)

- Overcomes bottlenecks of the Apriori approach, namely
  - Breadth-first (i.e., level-wise) search
  - Candidate generation and test (which often generates a huge number of candidates)

- The FPGrowth (frequent pattern growth) approach
  - Compresses the database using FP-tree, an extension of prefix-tree
  - It retains the transactions information
  - Never breaks a long pattern of any transaction
  - Depth-first search (DFS)
  - Avoids explicit candidate generation
    - Frequent itemsets are generated directly from the FP-tree.

Not relevant for the exams
FP Growth (Han, Pei & Yin, SIGMOD’00)

2 steps

1. Step 1: FP-tree construction
2. Step 2: Frequent Itemsets generation from the FP tree
Step 1: FP tree construction

<table>
<thead>
<tr>
<th>TID</th>
<th>Items bought</th>
<th>(ordered) frequent items</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>{f, a, c, d, g, i, m, p}</td>
<td>{f, c, a, m, p}</td>
</tr>
<tr>
<td>200</td>
<td>{a, b, c, f, l, m, o}</td>
<td>{f, c, a, b, m}</td>
</tr>
<tr>
<td>300</td>
<td>{b, f, h, j, o, w}</td>
<td>{f, b}</td>
</tr>
<tr>
<td>400</td>
<td>{b, c, k, s, p}</td>
<td>{c, b, p}</td>
</tr>
<tr>
<td>500</td>
<td>{a, f, c, e, l, p, m, n}</td>
<td>{f, c, a, m, p}</td>
</tr>
</tbody>
</table>

\[ \text{minSupport} = 3 \]

- Each transaction is mapped into a path in the FP-tree.
- To facilitate tree traversal, each item in the header table points to its occurrences in the tree via a chain of node-links.
- Most common items appear close to the root.

**Header Table**

<table>
<thead>
<tr>
<th>Item frequency</th>
<th>head</th>
</tr>
</thead>
<tbody>
<tr>
<td>f</td>
<td>4</td>
</tr>
<tr>
<td>c</td>
<td>4</td>
</tr>
<tr>
<td>a</td>
<td>3</td>
</tr>
<tr>
<td>b</td>
<td>3</td>
</tr>
<tr>
<td>m</td>
<td>3</td>
</tr>
<tr>
<td>p</td>
<td>3</td>
</tr>
</tbody>
</table>

\[ f\text{-list} = f\text{-c-a-b-m-p} \]
FP-tree construction step by step

<table>
<thead>
<tr>
<th>TID</th>
<th>Items bought</th>
<th>(ordered) frequent items</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>{f, a, c, d, g, i, m, p}</td>
<td>{f, c, a, m, p}</td>
</tr>
<tr>
<td>200</td>
<td>{a, b, c, f, l, m, o}</td>
<td>{f, c, a, b, m}</td>
</tr>
<tr>
<td>300</td>
<td>{b, f, h, j, o, w}</td>
<td>{f, b}</td>
</tr>
<tr>
<td>400</td>
<td>{b, c, k, s, p}</td>
<td>{c, b, p}</td>
</tr>
<tr>
<td>500</td>
<td>{a, f, c, e, l, p, m, n}</td>
<td>{f, c, a, m, p}</td>
</tr>
</tbody>
</table>

Not relevant for the exams
Advantages of the FP-tree structure

- Completeness
  - Preserves complete information for frequent pattern mining
  - Never breaks a long pattern of any transaction

- Compactness
  - Reduces irrelevant info—infrequent items are gone
  - Items in frequency descending order (f-list): the more frequently occurring, the more likely to be shared
  - Never is larger than the original database (not counting node-links and the node-counts fields)
  - Achieves high compression ratio

What is the best-case compression scenario for an FP-tree?

What is the worse-case compression scenario for an FP-tree?
Outline

- Introduction
- Basic concepts
- Frequent Itemsets Mining (FIM) – Apriori
- Association Rules Mining
  - Apriori improvements
  - Closed frequent itemsets (CFI) & Maximal frequent itemsets (MFI)
- Homework/tutorial
- Things you should know from this lecture
Too many frequent itemsets

- minSupport threshold helps pruning non-interesting itemsets. The resulting lattice though still depicts redundancies
  - Structural (i.e., in terms of itemsets' items)
  - Measural (i.e., in terms of itemsets' support)

- It is useful to identify a small representative set of itemsets from which all other itemsets can be derived

- Two compressed representations
  - Closed frequent itemsets (CFI)
  - Maximal frequent itemsets (MFI)

Not relevant for the exams
Closed Frequent Itemsets (CFI)

A frequent itemset $X$ is called closed if there exists no frequent superset $Y \supseteq X$ with:

$$\text{support}(X) = \text{support}(Y)$$

- The set of closed frequent itemsets is denoted by CFI.
- CFIs comprise a lossless representation of the FIs since no information is lost, neither in structure (itemsets), nor in measure (support).

Why $\{2, 3\}$ is not closed?

Why $\{3\}$ is closed?
Maximal Frequent Itemsets (MFI)

A frequent itemset is called maximal if it is not a subset of any other frequent itemset.

- The set of maximal frequent itemsets is denoted by MFI.
- MFIs comprise a lossy representation of the FIs since it is only the lattice structure (i.e., frequent itemsets) that can be determined from MFIs whereas exact frequent itemset supports are lost.

Why \{1,3\} is maximal?

Why \{2,3\} is not maximal? Why \{2,5\} is not maximal?
FIs vs CFIs vs MFIs

Data Mining I @SS19, Lecture 3: Frequent Itemsets and Association Rules Mining
FIs vs CFIs vs MFIs

- Frequent Itemsets
- Closed Frequent Itemsets
- Maximal Frequent Itemsets

Not relevant for the exams
Introduction

Basic concepts

Frequent Itemsets Mining (FIM) – Apriori

Association Rules Mining

Apriori improvements

Closed frequent itemsets (CFI) & Maximal frequent itemsets (MFI)

Homework/tutorial

Things you should know from this lecture
Homework/ tutorial

- Try Apriori and association rules mining in Weka/Python
  - E.g., use Weka installation folder/data/weather.nominal.arff or

- For other datasets to try (including a chess dataset)
  - http://fimi.ua.ac.be/data/

- 2nd tutorial follows next week

- There is no lecture next week (1.5), we meet again the week after

- Readings:
Things you should know from this lecture

- Frequent Itemsets, support, minSupport, itemsets lattice
- Association Rules, support, minSupport, confidence, minConfidence, strong rules
- Frequent Itemsets Mining: computation cost, negative border, downward closure property
- Apriori: join step, prune step, DB scans
- Association rules extraction from frequent itemsets
- Quality measures for association rules
- Pros and cons of Apriori
Acknowledgement

The slides are based on

- KDD I lecture at LMU Munich (Johannes Aßfalg, Christian Böhm, Karsten Borgwardt, Martin Ester, Eshref Januzaj, Karin Kailing, Peer Kröger, Eirini Ntoutsi, Jörg Sander, Matthias Schubert, Arthur Zimek, Andreas Züfle)
- Introduction to Data Mining book slides at http://www-users.cs.umn.edu/~kumar/dmbook/

Thank you to all TAs contributing to their improvement, namely Vasileios Iosifidis, Damianos Melidis, Tai Le Quy, Han Tran.