# UNIT-5 Mining Association Rules in Large Databases

<table>
<thead>
<tr>
<th>Lecture</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lecture-27</td>
<td>Association rule mining</td>
</tr>
<tr>
<td>Lecture-28</td>
<td>Mining single-dimensional Boolean association rules from transactional databases</td>
</tr>
<tr>
<td>Lecture-29</td>
<td>Mining multilevel association rules from transactional databases</td>
</tr>
<tr>
<td>Lecture-30</td>
<td>Mining multidimensional association rules from transactional databases and data warehouse</td>
</tr>
<tr>
<td>Lecture-31</td>
<td>From association mining to correlation analysis</td>
</tr>
<tr>
<td>Lecture-32</td>
<td>Constraint-based association mining</td>
</tr>
</tbody>
</table>
Lecture-27

Association rule mining
What Is Association Mining?

• Association rule mining
  
  – Finding frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.

• Applications
  
  – Basket data analysis, cross-marketing, catalog design, loss-leader analysis, clustering, classification, etc.
Association Mining

• Rule form

  prediction (Boolean variables) => prediction (Boolean variables) [support, confidence]

  – Computer => antivirus_software [support = 2%, confidence = 60%]
  – buys (x, “computer”) => buys (x, “antivirus_software”) [0.5%, 60%]
Association Rule: Basic Concepts

• Given a database of transactions each transaction is a list of items (purchased by a customer in a visit)
• Find all rules that correlate the presence of one set of items with that of another set of items
• Find frequent patterns
• Example for frequent itemset mining is market basket analysis.
Association rule performance measures

• Confidence
• Support
• Minimum support threshold
• Minimum confidence threshold
Rule Measures: Support and Confidence

- Find all the rules $X \& Y \Rightarrow Z$ with minimum confidence and support
  - support, $s$, probability that a transaction contains $\{X \& Y \& Z\}$
  - confidence, $c$, conditional probability that a transaction having $\{X \& Y\}$ also contains $Z$

Let minimum support 50%, and minimum confidence 50%, we have
- $A \Rightarrow C$ (50%, 66.6%)
- $C \Rightarrow A$ (50%, 100%)
Martket Basket Analysis

• Shopping baskets
• Each item has a Boolean variable representing the presence or absence of that item.
• Each basket can be represented by a Boolean vector of values assigned to these variables.
• Identify patterns from Boolean vector
• Patterns can be represented by association rules.
Association Rule Mining: A Road Map

• Boolean vs. quantitative associations
  - Based on the types of values handled
    – \( \text{bought}(x, \text{"SQLServer"}) \land \text{bought}(x, \text{"DMBook"}) \Rightarrow \text{bought}(x, \text{"DBMiner"}) \) [0.2%, 60%]
    – \( \text{age}(x, \text{"30..39"}) \land \text{income}(x, \text{"42..48K"}) \Rightarrow \text{bought}(x, \text{"PC"}) \) [1%, 75%]
• Single dimension vs. multiple dimensional associations
• Single level vs. multiple-level analysis
Lecture-28
Mining single-dimensional Boolean association rules from transactional databases
Apriori Algorithm

• Single dimensional, single-level, Boolean frequent item sets
• Finding frequent item sets using candidate generation
• Generating association rules from frequent item sets
Mining Association Rules—An Example

<table>
<thead>
<tr>
<th>Transaction ID</th>
<th>Items Bought</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>A, B, C</td>
</tr>
<tr>
<td>1000</td>
<td>A, C</td>
</tr>
<tr>
<td>4000</td>
<td>A, D</td>
</tr>
<tr>
<td>5000</td>
<td>B, E, F</td>
</tr>
</tbody>
</table>

Min. support 50%
Min. confidence 50%

<table>
<thead>
<tr>
<th>Frequent Itemset</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>75%</td>
</tr>
<tr>
<td>{B}</td>
<td>50%</td>
</tr>
<tr>
<td>{C}</td>
<td>50%</td>
</tr>
<tr>
<td>{A, C}</td>
<td>50%</td>
</tr>
</tbody>
</table>

For rule A \( \Rightarrow \) C:

support = support({A \( \supseteq \) C}) = 50%

confidence = support({A \( \supseteq \) C})/support({A}) = 66.6%

The Apriori principle:

Any subset of a frequent itemset must be frequent

Lecture-28
Mining single-dimensional Boolean association rules from transactional databases
Mining Frequent Itemsets: the Key Step

• Find the *frequent itemsets*: the sets of items that have minimum support
  – A subset of a frequent itemset must also be a frequent itemset
    • i.e., if \( \{AB\} \) is a frequent itemset, both \( \{A\} \) and \( \{B\} \) should be a frequent itemset
  – Iteratively find frequent itemsets with cardinality from 1 to \( k \) (\( k \)-itemset)

• Use the frequent itemsets to generate association rules.
The Apriori Algorithm

• Join Step
  – $C_k$ is generated by joining $L_{k-1}$ with itself

• Prune Step
  – Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset
The Apriori Algorithm

- **Pseudo-code:**
  
  $C_k$: Candidate itemset of size $k$
  $L_k$: frequent itemset of size $k$

  $L_1 = \{\text{frequent items}\}$;

  for ($k = 1; L_k \neq \emptyset; k++$) do begin
    $C_{k+1} = \text{candidates generated from } L_k$;
    for each transaction $t$ in database do
      for each transaction $t$ in database do
        increment the count of all candidates in $C_{k+1}$ that are contained in $t$
    end
    $L_{k+1} = \text{candidates in } C_{k+1} \text{ with min_support}$
  end
  return $\bigcup_k L_k$;
The Apriori Algorithm — Example

Database D

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1 3 4</td>
</tr>
<tr>
<td>200</td>
<td>2 3 5</td>
</tr>
<tr>
<td>300</td>
<td>1 2 3 5</td>
</tr>
<tr>
<td>400</td>
<td>2 5</td>
</tr>
</tbody>
</table>

**C1**

<table>
<thead>
<tr>
<th>itemset</th>
<th>sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>2</td>
</tr>
<tr>
<td>{2}</td>
<td>3</td>
</tr>
<tr>
<td>{3}</td>
<td>3</td>
</tr>
<tr>
<td>{4}</td>
<td>1</td>
</tr>
<tr>
<td>{5}</td>
<td>3</td>
</tr>
</tbody>
</table>

Scan D

**L1**

<table>
<thead>
<tr>
<th>itemset</th>
<th>sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>2</td>
</tr>
<tr>
<td>{2}</td>
<td>3</td>
</tr>
<tr>
<td>{3}</td>
<td>3</td>
</tr>
<tr>
<td>{4}</td>
<td></td>
</tr>
<tr>
<td>{5}</td>
<td>3</td>
</tr>
</tbody>
</table>

**C2**

<table>
<thead>
<tr>
<th>itemset</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>itemset</th>
<th>sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>{2 3 5}</td>
<td></td>
</tr>
</tbody>
</table>

Scan D

**L3**

<table>
<thead>
<tr>
<th>itemset</th>
<th>sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>{2 3 5}</td>
<td>2</td>
</tr>
</tbody>
</table>

Lecture-28

Mining single-dimensional Boolean association rules from transactional databases
How to Generate Candidates?

• Suppose the items in $L_{k-1}$ are listed in an order

• Step 1: self-joining $L_{k-1}$

  insert into $C_k$

  select $p.item_1$, $p.item_2$, ..., $p.item_{k-1}$, $q.item_{k-1}$

  from $L_{k-1} p$, $L_{k-1} q$

  where $p.item_1$=$q.item_1$, ..., $p.item_{k-2}$=$q.item_{k-2}$, $p.item_{k-1}$ < $q.item_{k-1}$

• Step 2: pruning

  forall itemsets $c$ in $C_k$ do

    forall (k-1)-subsets $s$ of $c$ do

      if (s is not in $L_{k-1}$) then delete $c$ from $C_k$
How to Count Supports of Candidates?

• Why counting supports of candidates a problem?
  – The total number of candidates can be very huge
  – One transaction may contain many candidates

• Method
  – Candidate itemsets are stored in a hash-tree
  – Leaf node of hash-tree contains a list of itemsets and counts
  – Interior node contains a hash table
  – Subset function: finds all the candidates contained in a transaction
Example of Generating Candidates

- \( L_3 = \{abc, abd, acd, ace, bcd\} \)
- Self-joining: \( L_3 \times L_3 \)
  - \( abcd \) from \( abc \) and \( abd \)
  - \( acde \) from \( acd \) and \( ace \)
- Pruning:
  - \( acde \) is removed because \( ade \) is not in \( L_3 \)
- \( C_4 = \{abcd\} \)
Methods to Improve Apriori’s Efficiency

• Hash-based itemset counting
  – A $k$-itemset whose corresponding hashing bucket count is below the threshold cannot be frequent

• Transaction reduction
  – A transaction that does not contain any frequent $k$-itemset is useless in subsequent scans

• Partitioning
  – Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB

Lecture-28
Mining single-dimensional Boolean association rules from transactional databases
Methods to Improve Apriori’s Efficiency

• **Sampling**
  – mining on a subset of given data, lower support threshold
  + a method to determine the completeness

• **Dynamic itemset counting**
  – add new candidate itemsets only when all of their subsets are estimated to be frequent
Mining Frequent Patterns Without Candidate Generation

• Compress a large database into a compact, Frequent-Pattern tree (FP-tree) structure
  – highly condensed, but complete for frequent pattern mining
  – avoid costly database scans

• Develop an efficient, FP-tree-based frequent pattern mining method
  – A divide-and-conquer methodology: decompose mining tasks into smaller ones
  – Avoid candidate generation: sub-database test only
Lecture-29

Mining multilevel association rules from transactional databases
Mining various kinds of association rules

- Mining Multilevel association rules
  - Concepts at different levels
- Mining Multidimensional association rules
  - More than one dimensional
- Mining Quantitative association rules
  - Numeric attributes
Multiple-Level Association Rules

- Items often form hierarchy.
- Items at the lower level are expected to have lower support.
- Rules regarding itemsets at appropriate levels could be quite useful.
- Transaction database can be encoded based on dimensions and levels
- We can explore shared multi-level mining

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>{111, 121, 211, 221}</td>
</tr>
<tr>
<td>T2</td>
<td>{111, 211, 222, 323}</td>
</tr>
<tr>
<td>T3</td>
<td>{112, 122, 221, 411}</td>
</tr>
<tr>
<td>T4</td>
<td>{111, 121}</td>
</tr>
<tr>
<td>T5</td>
<td>{111, 122, 211, 221, 413}</td>
</tr>
</tbody>
</table>
Multi-level Association

• Uniform Support- the same minimum support for all levels
  – + One minimum support threshold. No need to examine itemsets containing any item whose ancestors do not have minimum support.
  – – Lower level items do not occur as frequently. If support threshold
    • too high ⇒ miss low level associations
    • too low ⇒ generate too many high level associations
Multi-level Association

• Reduced Support- reduced minimum support at lower levels
  – There are 4 search strategies:
    • Level-by-level independent
    • Level-cross filtering by k-itemset
    • Level-cross filtering by single item
    • Controlled level-cross filtering by single item
Uniform Support

Multi-level mining with uniform support

Level 1
min_sup = 5%

Level 2
min_sup = 5%

Milk
[support = 10%]

2% Milk
[support = 6%]

Skim Milk
[support = 4%]
Reduced Support

Multi-level mining with reduced support

Level 1
min_sup = 5%

Level 2
min_sup = 3%

Milk
[support = 10%]

2% Milk
[support = 6%]

Skim Milk
[support = 4%]
Multi-level Association: Redundancy Filtering

• Some rules may be redundant due to “ancestor” relationships between items.

• Example
  – milk ⇒ wheat bread [support = 8%, confidence = 70%]
  – 2% milk ⇒ wheat bread [support = 2%, confidence = 72%]

• We say the first rule is an ancestor of the second rule.

• A rule is redundant if its support is close to the “expected” value, based on the rule’s ancestor.
Lecture-30

Mining multidimensional association rules from transactional databases and data warehouse
Multi-Dimensional Association

• Single-dimensional rules
  \[ \text{buys}(X, \text{“milk”}) \implies \text{buys}(X, \text{“bread”}) \]

• Multi-dimensional rules
  – Inter-dimension association rules -no repeated predicates
    \[ \text{age}(X, \text{“19-25”}) \land \text{occupation}(X, \text{“student”}) \implies \text{buys}(X, \text{“coke”}) \]
  – hybrid-dimension association rules -repeated predicates
    \[ \text{age}(X, \text{“19-25”}) \land \text{buys}(X, \text{“popcorn”}) \implies \text{buys}(X, \text{“coke”}) \]
Multi-Dimensional Association

• Categorical Attributes
  – finite number of possible values, no ordering among values

• Quantitative Attributes
  – numeric, implicit ordering among values
Techniques for Mining MD Associations

• Search for frequent $k$-predicate set:
  – Example: \{age, occupation, buys\} is a 3-predicate set.
  – Techniques can be categorized by how age are treated.
1. Using static discretization of quantitative attributes
   – Quantitative attributes are statically discretized by using predefined concept hierarchies.
2. Quantitative association rules
   – Quantitative attributes are dynamically discretized into “bins” based on the distribution of the data.
3. Distance-based association rules
   – This is a dynamic discretization process that considers the distance between data points.
Static Discretization of Quantitative Attributes

- Discretized prior to mining using concept hierarchy.
- Numeric values are replaced by ranges.
- In relational database, finding all frequent k-predicate sets will require k or k+1 table scans.
- Data cube is well suited for mining.
- The cells of an n-dimensional cuboid correspond to the predicate sets.
- Mining from data cubes can be much faster.

Lecture-30 - Mining multidimensional association rules from transactional databases and data warehouse
Quantitative Association Rules

- Numeric attributes are *dynamically* discretized
  - Such that the confidence or compactness of the rules mined is maximized.

- 2-D quantitative association rules: \( A_{\text{quan1}} \land A_{\text{quan2}} \Rightarrow A_{\text{cat}} \)

- Cluster “adjacent” association rules to form general rules using a 2-D grid.

- Example:

\[
\text{age}(X, "30-34") \land \text{income}(X, "24K - 48K") \\
\Rightarrow \text{buys}(X, "high resolution TV")
\]
Lecture-31
From association mining to correlation analysis
Interestingness Measurements

• Objective measures
  – Two popular measurements
    support
    confidence

• Subjective measures
  A rule (pattern) is interesting if
  *it is unexpected (surprising to the user); and/or
  *actionable (the user can do something with it)
Criticism to Support and Confidence

• Example
  – Among 5000 students
    • 3000 play basketball
    • 3750 eat cereal
    • 2000 both play basketball and eat cereal
  – play basketball ⇒ eat cereal [40%, 66.7%] is misleading because the overall percentage of students eating cereal is 75% which is higher than 66.7%.
  – play basketball ⇒ not eat cereal [20%, 33.3%] is far more accurate, although with lower support and confidence

<table>
<thead>
<tr>
<th></th>
<th>basketball</th>
<th>not basketball</th>
<th>sum(row)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cereal</td>
<td>2000</td>
<td>1750</td>
<td>3750</td>
</tr>
<tr>
<td>not cereal</td>
<td>1000</td>
<td>250</td>
<td>1250</td>
</tr>
<tr>
<td>sum(col.)</td>
<td>3000</td>
<td>2000</td>
<td>5000</td>
</tr>
</tbody>
</table>
Criticism to Support and Confidence

• Example
  – X and Y: positively correlated,
  – X and Z, negatively related
  – support and confidence of
    X => Z dominates
• We need a measure of dependent or correlated events

\[ corr_{A,B} = \frac{P(A \cup B)}{P(A)P(B)} \]

• \( P(B \mid A)/P(B) \) is also called the lift of rule \( A \Rightarrow B \)

<table>
<thead>
<tr>
<th>Rule</th>
<th>Support</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>X =&gt; Y</td>
<td>25%</td>
<td>50%</td>
</tr>
<tr>
<td>X =&gt; Z</td>
<td>37.50%</td>
<td>75%</td>
</tr>
</tbody>
</table>
Other Interestingness Measures: Interest

- Interest (correlation, lift) \( \frac{P(A \land B)}{P(A)P(B)} \)
  - taking both \( P(A) \) and \( P(B) \) in consideration
  - \( P(A \land B)=P(B)*P(A) \), if \( A \) and \( B \) are independent events
  - \( A \) and \( B \) negatively correlated, if the value is less than 1; otherwise \( A \) and \( B \) positively correlated

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Support</th>
<th>Interest</th>
</tr>
</thead>
<tbody>
<tr>
<td>X,Y</td>
<td>25%</td>
<td>2</td>
</tr>
<tr>
<td>X,Z</td>
<td>37.50%</td>
<td>0.9</td>
</tr>
<tr>
<td>Y,Z</td>
<td>12.50%</td>
<td>0.57</td>
</tr>
</tbody>
</table>

Lecture-31 - From association mining to correlation analysis
Lecture-32
Constraint-based association mining
Constraint-Based Mining

• Interactive, exploratory mining
• kinds of constraints
  – Knowledge type constraint: classification, association, etc.
  – Data constraint: SQL-like queries
  – Dimension/level constraints
  – Rule constraint
  – Interestingness constraints
Rule Constraints in Association Mining

• Two kind of rule constraints:
  – Rule form constraints: meta-rule guided mining.
    • \( P(x, y) \land Q(x, w) \rightarrow \text{takes}(x, \text{“database systems”}) \).
  – Rule (content) constraint: constraint-based query optimization (Ng, et al., SIGMOD’98).
    • \( \text{sum(LHS)} < 100 \land \text{min(LHS)} > 20 \land \text{count(LHS)} > 3 \land \text{sum(RHS)} > 1000 \)

• 1-variable vs. 2-variable constraints
  – 1-var: A constraint confining only one side (L/R) of the rule, e.g., as shown above.
  – 2-var: A constraint confining both sides (L and R).
    • \( \text{sum(LHS)} < \text{min(RHS)} \land \text{max(RHS)} < 5 \ast \text{sum(LHS)} \)
Constrain-Based Association Query

• Database: (1) trans (TID, Itemset), (2) itemInfo (Item, Type, Price)
• A constrained asso. query (CAQ) is in the form of \{(S_1, S_2)/C\},
  where C is a set of constraints on S_1, S_2 including frequency constraint
• A classification of (single-variable) constraints:
  – Class constraint: S \subseteq A.  \textit{e.g.} S \subseteq Item
  – Domain constraint:
    • S\theta v, \theta \in \{=, \neq, <, \leq, >, \geq\}. \textit{e.g.} S.Price < 100
    • v\theta S, \theta \in \in or \notin. \textit{e.g.} snacks \notin S.Type
    • V\theta S, or S\theta V, \theta \in \{\subseteq, \subset, \varnothing, =, \neq\}
      – \textit{e.g.} \{snacks, sodas\} \subseteq S.Type
  – Aggregation constraint: agg(S) \theta v, where agg is in \{min, max, sum, count, avg\}, and \theta \in \{=, \neq, <, \leq, >, \geq\}.
    • \textit{e.g.} count(S_1.Type) = 1, \text{avg}(S_2.Price) < 100
Constrained Association Query Optimization Problem

• Given a CAQ = \{ (S_1, S_2) / C \}, the algorithm should be:
  – sound: It only finds frequent sets that satisfy the given constraints C
  – complete: All frequent sets satisfy the given constraints C are found

• A naïve solution:
  – Apply Apriori for finding all frequent sets, and then to test them for constraint satisfaction one by one.

• Our approach:
  – Comprehensive analysis of the properties of constraints and try to push them as deeply as possible inside the frequent set computation.
Anti-monotone and Monotone Constraints

• A constraint $C_a$ is anti-monotone iff. for any pattern $S$ not satisfying $C_a$, none of the super-patterns of $S$ can satisfy $C_a$

• A constraint $C_m$ is monotone iff. for any pattern $S$ satisfying $C_m$, every super-pattern of $S$ also satisfies it
Succinct Constraint

• A subset of item $I_s$ is a succinct set, if it can be expressed as $\sigma_p(I)$ for some selection predicate $p$, where $\sigma$ is a selection operator

• $SP \subseteq 2^I$ is a succinct power set, if there is a fixed number of succinct set $I_1, ..., I_k \subseteq I$, s.t. $SP$ can be expressed in terms of the strict power sets of $I_1, ..., I_k$ using union and minus

• A constraint $C_s$ is succinct provided $\text{SAT}_{Cs}(I)$ is a succinct power set
Convertible Constraint

- Suppose all items in patterns are listed in a total order R
- A constraint C is convertible anti-monotone iff a pattern S satisfying the constraint implies that each suffix of S w.r.t. R also satisfies C
- A constraint C is convertible monotone iff a pattern S satisfying the constraint implies that each pattern of which S is a suffix w.r.t. R also satisfies C
Relationships Among Categories of Constraints

- Succinctness
- Anti-monotonicity
- Monotonicity
-Convertible constraints
-Inconvertible constraints
Property of Constraints: Anti-Monotone

- Anti-monotonicity: If a set $S$ violates the constraint, any superset of $S$ violates the constraint.

- Examples:
  - $\text{sum}(S.\text{Price}) \leq v$ is anti-monotone
  - $\text{sum}(S.\text{Price}) \geq v$ is not anti-monotone
  - $\text{sum}(S.\text{Price}) = v$ is partly anti-monotone

- Application:
  - Push “$\text{sum}(S.\text{price}) \leq 1000$” deeply into iterative frequent set computation.
Characterization of Anti-Monotonicity Constraints

<table>
<thead>
<tr>
<th>Condition</th>
<th>Requirement</th>
<th>Validity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \theta v$, $\theta \in {=, \leq, \geq}$</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>$v \in S$</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>$S \geq V$</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>$S \leq V$</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>$S = V$</td>
<td>partly</td>
<td></td>
</tr>
<tr>
<td>$\min(S) \leq v$</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>$\min(S) \geq v$</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>$\min(S) = v$</td>
<td>partly</td>
<td></td>
</tr>
<tr>
<td>$\max(S) \leq v$</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>$\max(S) \geq v$</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>$\max(S) = v$</td>
<td>partly</td>
<td></td>
</tr>
<tr>
<td>$\text{count}(S) \leq v$</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>$\text{count}(S) \geq v$</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>$\text{count}(S) = v$</td>
<td>partly</td>
<td></td>
</tr>
<tr>
<td>$\text{sum}(S) \leq v$</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>$\text{sum}(S) \geq v$</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>$\text{sum}(S) = v$</td>
<td>partly</td>
<td></td>
</tr>
<tr>
<td>$\text{avg}(S) \theta v$, $\theta \in {=, \leq, \geq}$</td>
<td>convertible</td>
<td></td>
</tr>
<tr>
<td>(frequent constraint)</td>
<td>(yes)</td>
<td></td>
</tr>
</tbody>
</table>
Example of Convertible Constraints: \( \text{Avg}(S) \geq V \)

- Let \( R \) be the value descending order over the set of items
  - E.g. \( I = \{9, 8, 6, 4, 3, 1\} \)
- \( \text{Avg}(S) \geq v \) is convertible monotone w.r.t. \( R \)
  - If \( S \) is a suffix of \( S_1 \), \( \text{avg}(S_1) \geq \text{avg}(S) \)
    - \( \{8, 4, 3\} \) is a suffix of \( \{9, 8, 4, 3\} \)
    - \( \text{avg}(\{9, 8, 4, 3\}) = 6 \geq \text{avg}(\{8, 4, 3\}) = 5 \)
  - If \( S \) satisfies \( \text{avg}(S) \geq v \), so does \( S_1 \)
    - \( \{8, 4, 3\} \) satisfies constraint \( \text{avg}(S) \geq 4 \), so does \( \{9, 8, 4, 3\} \)
Property of Constraints: Succinctness

• Succinctness:
  – For any set $S_1$ and $S_2$ satisfying $C$, $S_1 \cup S_2$ satisfies $C$
  – Given $A_1$ is the sets of size 1 satisfying $C$, then any set $S$ satisfying $C$ are based on $A_1$, i.e., it contains a subset belongs to $A_1$.

• Example:
  – $\text{sum}(S.\text{Price}) \geq v$ is not succinct
  – $\text{min}(S.\text{Price}) \leq v$ is succinct

• Optimization:
  – If $C$ is succinct, then $C$ is pre-counting prunable. The satisfaction of the constraint alone is not affected by the iterative support counting.
Characterization of Constraints by Succinctness

<table>
<thead>
<tr>
<th>Condition</th>
<th>Succinctness</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \theta v$, $\theta \in {=, \leq, \geq}$</td>
<td>Yes</td>
</tr>
<tr>
<td>$v \in S$</td>
<td>yes</td>
</tr>
<tr>
<td>$S \geq V$</td>
<td>yes</td>
</tr>
<tr>
<td>$S \leq V$</td>
<td>yes</td>
</tr>
<tr>
<td>$S = V$</td>
<td>yes</td>
</tr>
<tr>
<td>$\min(S) \leq v$</td>
<td>yes</td>
</tr>
<tr>
<td>$\min(S) \geq v$</td>
<td>yes</td>
</tr>
<tr>
<td>$\min(S) = v$</td>
<td>yes</td>
</tr>
<tr>
<td>$\max(S) \leq v$</td>
<td>yes</td>
</tr>
<tr>
<td>$\max(S) \geq v$</td>
<td>yes</td>
</tr>
<tr>
<td>$\max(S) = v$</td>
<td>yes</td>
</tr>
<tr>
<td>$\text{count}(S) \leq v$</td>
<td>weakly</td>
</tr>
<tr>
<td>$\text{count}(S) \geq v$</td>
<td>weakly</td>
</tr>
<tr>
<td>$\text{count}(S) = v$</td>
<td>weakly</td>
</tr>
<tr>
<td>$\text{sum}(S) \leq v$</td>
<td>no</td>
</tr>
<tr>
<td>$\text{sum}(S) \geq v$</td>
<td>no</td>
</tr>
<tr>
<td>$\text{sum}(S) = v$</td>
<td>no</td>
</tr>
<tr>
<td>$\text{avg}(S) \theta v$, $\theta \in {=, \leq, \geq}$</td>
<td>no</td>
</tr>
</tbody>
</table>

(for frequent constraint) (no)