**Data Mining**

**Association**

NEXT PAGE

NOTES TO FOLLOW

* Association rule induction: Originally designed for **market basket analysis.**
* Aims at finding patterns in the shopping behaviour of customers of supermarkets, mail-order companies, on-line shops etc.
* More specifically:

#### Find sets of products that are frequently bought together.

* Example of an association rule:

*If a customer buys bread and wine,*

*then she/he will probably also buy cheese.*

### Possible applications of found association rules:

* + Improve arrangement of products in shelves, on a catalogue's pages.
  + Support of cross-selling (suggestion of other products),

product bundling.

* + Fraud detection, technical dependence analysis.
  + Finding business rules and detection of data quality problems.

– ……

# Association Rules

### Assessing the quality of association rules:

* + Support of an item set:
  + Fraction of transactions (shopping baskets/carts) that contain the item set.
  + Support of an association rule x→ y :

Either: Support of x ∪ y

(more common: rule is correct)

Or: Support of X

(more plausible: rule is applicable)

– Confidence of an association rule x → y:

Support of � ∪ � divided by support of X (estimate of P(Y|X)).

# Association Rules

### Two step implementation of the search for

association rules:

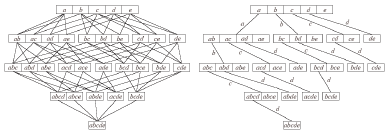
* + Find the frequent item sets (also called large item sets), i.e., the item sets that have at least a user-defined

#### minimum support.

* + Form rules using the frequent item sets found and select those that have at least a user-defined **minimum confidence.**

# Finding frequent item sets

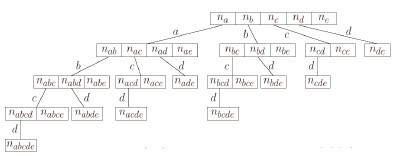
### Subset lattice and a prefix tree for five items:



* It is not possible to determine the support of all possible item sets,

because their number grows exponentially with the number of items.

* Efficient methods to search the subset lattice are needed.



A (full) item set tree for the five items a, b, c, d, and e.

* Based on a global order of the items.
* The item sets counted in a node consist of
  + all items labelling the edges to the node (common prefix) and
  + one item following the last edge label.

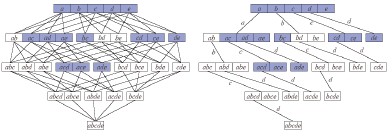
### In applications item set trees tend to get very large, so pruning is needed.

* Structural Pruning:
  + Make sure that there is only one counter for each possible item set.
  + Explains the unbalanced structure of the full item set tree.
* Size Based Pruning:
  + Prune the tree if a certain depth (a certain size of the item sets) is

reached.

* + Idea: Rules with too many items are difficult to interpret.
* Support Based Pruning:
  + No superset of an infrequent item set can be frequent.
  + No counters for item sets having an infrequent subset are needed.

Boundary between frequent (blue) and infrequent (white) item sets:



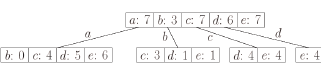
* Apriori: Breadth-first search (item sets of same size).
* Eclat: Depth-first search (item sets with same prefix).

|  |  |
| --- | --- |
| **Transaction ID** | **Items** |
| 1 | a, d, e |
| 2 | b, c, d |
| 3 | a, c, e |
| 4 | a, c, d, e |
| 5 | a, e |

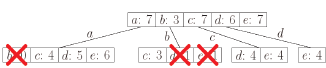
|  |  |
| --- | --- |
| **Transaction ID** | **Items** |
| 6 | a, c, d |
| 7 | b, c |
| 8 | a, c, d, e |
| 9 | c, b, e |
| 10 | a, d, e |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **item** | **a** | **b** | **c** | **d** | **e** |
| frequency | 7 | 3 | 7 | 6 | 7 |

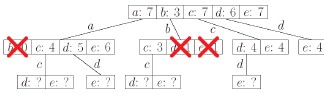
* Example transaction database with 5 items and 10 transactions.
* Minimum support: 30%, i.e., at least 3 transactions must contain the item set.
* All one item sets are frequent → full second level is needed.



* Determining the support of item sets: For each item set traverse the database and count the transactions that contain it (highly inefficient).
* Better: Traverse the tree for each transaction and find the item sets it contains (efficient: can be implemented as a simple double recursive procedure).



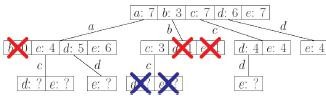
* Minimum support: 30%, i.e., at least 3 transactions must contain the item set.
* Infrequent item sets: {a, b}, {b, d}, {b, c}
* The subtrees starting at these item sets can be pruned.



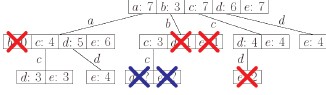
* Generate candidate item sets with 3 items (parents must be frequent).
* Before counting, check whether the candidates contain

an infrequent item set.

* An item set with k items has k subsets of size k - 1.
* The parent is only one of these subsets.



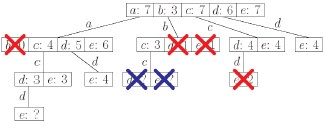
* The item sets {b, c, d} and {b, c, e} can be pruned, because
* {b, c, d} contains the infrequent item set {b, d} and
* {b, c, e} contains the infrequent item set {b, c}.
* Only the remaining four item sets of size 3 are evaluated.



* Minimum support: 30%, i.e., at least 3 transactions

must contain the item set.

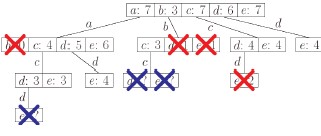
* Infrequent item set: {c, d, e}.



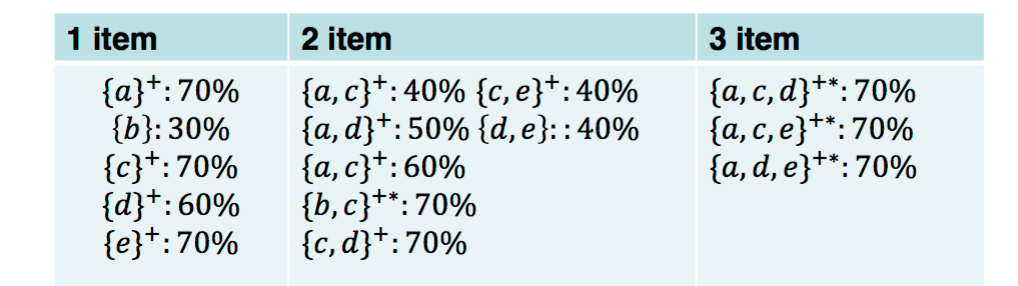
* Generate candidate item sets with 4 items (parents

must be frequent).

* Before counting, check whether the candidates contain an infrequent item set.

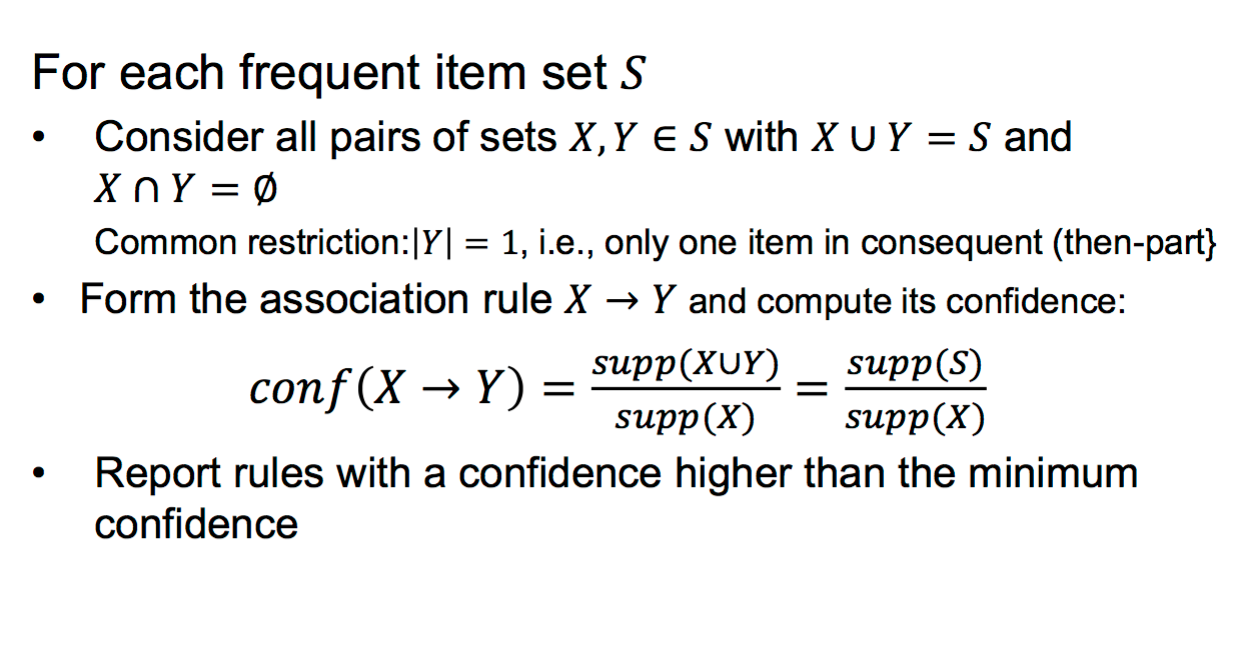


* The item set {a, c, d, e} can be pruned, because it contains the infrequent item set {c, d, e}
* Consequence: No candidate item sets with four items.
* Fourth access to the transaction database is not necessary.



Types of frequent item sets

* **Free Item Set**: Any frequent item set (support is higher than the minimal support).
* **Closed Item Set** (marked with +): A frequent item set is called *closed* if no superset has the same support.
* **Maximal Item Set** (marked with \*): A frequent item set is called *maximal* if no superset is frequent.



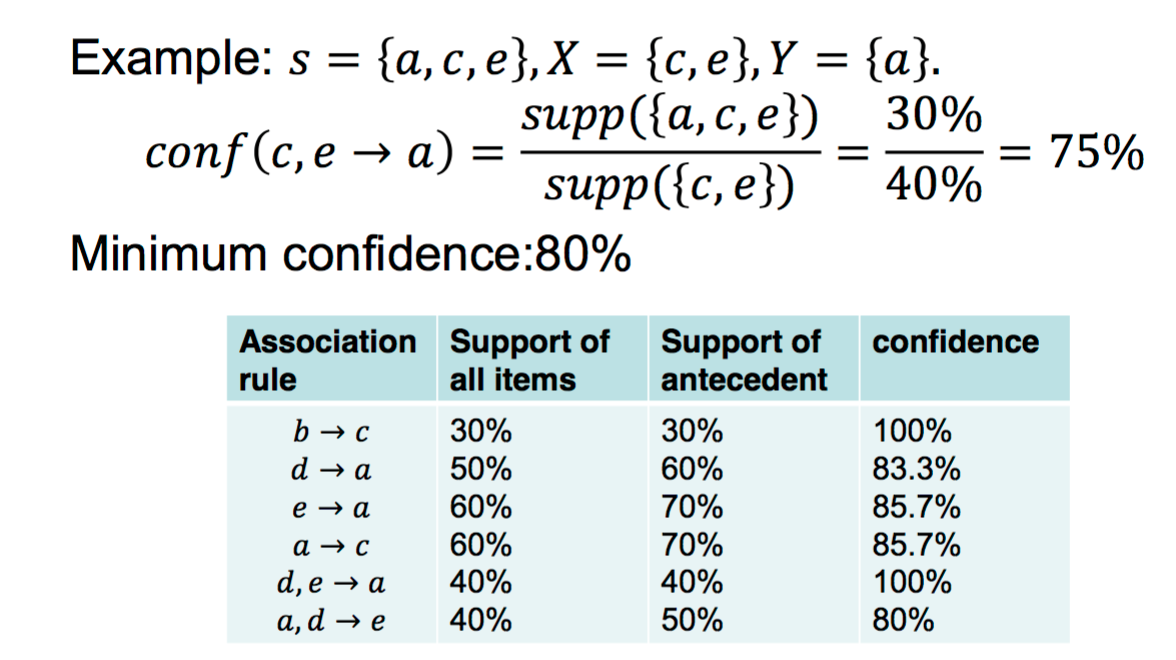
### Further rule filtering can rely on:

* Require a minimum difference between rule

confidence and consequent support.

* Compute information gain or 𝑥2 for antecedent (if-

part) and consequent.

Generating association rules

* Association Rule Induction is a Two Step Process
* Find the frequent item sets (minimum support).
* Form the relevant association rules (minimum confidence).
* Finding the Frequent Item Sets
* Top-down search in the subset lattice / item set tree.
* Apriori: Breadth first search;
* Other algorithms: Eclat, FP-growth, H-Mine, LCM, Mafia, Relim etc.
* Search Tree Pruning:

No superset of an infrequent item set can be frequent.

* Generating the Association Rules
* Form all possible association rules from the frequent item sets.
* Filter "interesting" association rules
* Sometimes, an additional structure is imposed on the

"item sets".

* The "item sets" are sequences of events.
  + For instance: Customer contact (buying, complaint, questionnaire)
  + Association rules have the form: If a and then b happens, then

probably c happens next.

* Items sets are molecules: Find frequent substructures.

The additional structure leads to different tree structure, but the principal algorithm remains the same.

## Other applications

### Finding business rules and detection of data

quality problems.

* Association rules with confidence close to 100% could

be business rules.

* Exceptions might be caused by data quality problems.

### Construction of partial classifiers.

* Search for association rules with a given conclusion part.
* If …, then the customer probably buys the product.

# References

### M. Berthold, C. Borgelt, F. Hoppner and F. Klawonn Guide to Intelligent Data Analysis, Spriner-Verlag,

2010

* + Chapter 7, Section 7.6

### I. Witten, E. Frank and M. Hall (2011)

*Data Mining: Practical Machine Learning Tools and Techniques*,

### Morgan Kaufmann Publishers.

* <http://www.kdnuggets.com/>