Machine Learning Notes

by Mattia Orlandi

4. Association Rules

4.1. Introduction to Market Basket Analysis

- Given a set of commercial transactions, find rules to predict the occurrences of an item based on the occurrences of other items in the transaction.
- Implication means **co-occurrence**, **not causality**.

Definitions:

- **Itemset**: collection of one or more items.
- **k-itemset**: itemset containing k items.
- **Support count σ**: frequency of occurrence of an itemset.
- **Support**: fraction of transactions containing an itemset.
- **Frequent itemset**: itemset whose support is greater than or equal to a minsup threshold.
- **Association Rule**: expression of the form $A \Rightarrow C$, where A (antecedent) and C (consequent) are itemsets.
- **Rule Evaluation Metrics**:
  - **Support (sup)**: fraction of the N transactions containing both A and $C : \text{sup} = \frac{\sigma(A \Rightarrow C)}{N}$.
  - **Confidence (conf)**: measures how often all the items in C appear in transactions containing $A : \text{conf} = \frac{\sigma(A \Rightarrow C)}{\sigma(A)}$.

Support and confidence

- Rules with low support can be generated by random associations;
- Rules with low confidence are not reliable;
- Rules with low support but high confidence can represent an uncommon but interesting phenomenon.

Association Rules Mining:

- Given a set of transactions N, the goal is to find all rules having:
  - $\text{sup} \geq \text{minsup}$ threshold
  - $\text{conf} \geq \text{minconf}$ threshold

- Brute-force approach:
  1. List all possible association rules.
  2. Compute support and confidence for each rule.
  3. Prune rules failing the thresholds.
  - $\Rightarrow$ **Computationally prohibitive** (3^N possible association rules).
- Two-step approach:
  1. **Frequent Itemset Generation**: generate all itemsets whose support is greater that minsup.
  2. **Rule Generation**: generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset.
  - $\Rightarrow$ still computationally expensive.

4.2. Frequent Itemset Generation

- Given D items, there are $M = 2^D$ candidate itemsets $\Rightarrow$ complexity with a brute-force approach: $\mathcal{O}(NWM)$, N : transactions and W : items per transaction.
- To reduce the number of candidates M, pruning techniques are used.
- To reduce the number of comparisons NM, efficient data structures are used (moreover, there's no need to match every candidate against every transaction).

Apriori principle

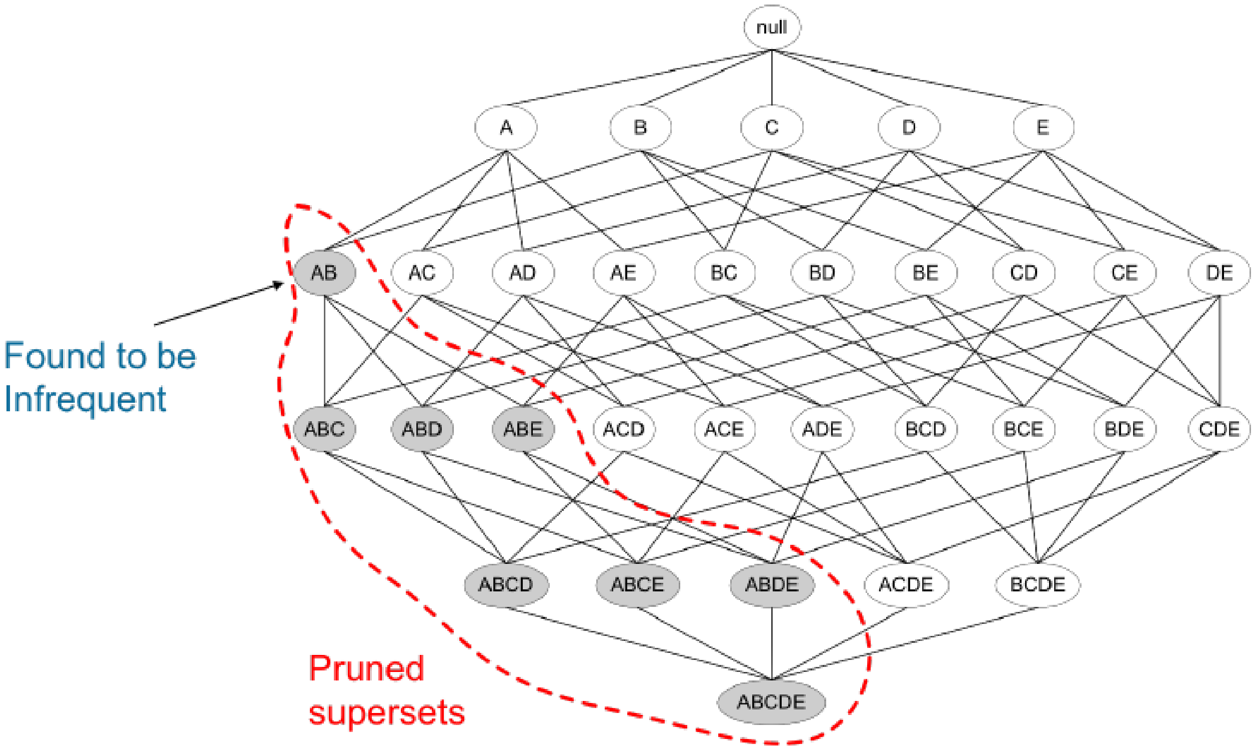
- If an itemset is frequent, then all of its subsets must also be frequent:

$$\forall X, Y : (X \subseteq Y) \Rightarrow sup(X) \geq sup(Y)$$

The support of an itemset never exceeds the support of its subsets (**anti-monotone** property).

Apriori algorithm

- Definitions:
  - C_k : candidate itemsets of size k;
  - L_k : frequent itemset of size k;
  - $subset_k(c)$: set of the subsets of c with k elements.
- Candidate generation:
  1. **Join Step**:
     1. Let L_k be represented as a table with k columns where each row is a frequent itemset.
     2. Let the items in each row of L_k be in lexicographic order.
     3. C_{k+1} is generated by a self join of L_k.
  2. **Prune Step**: each $(k+1)$-itemset including a k-itemset that is not in L_k is deleted from C_{k+1}.
- Computation at level k uses *prior knowledge* acquired for previous levels to reduce search space.



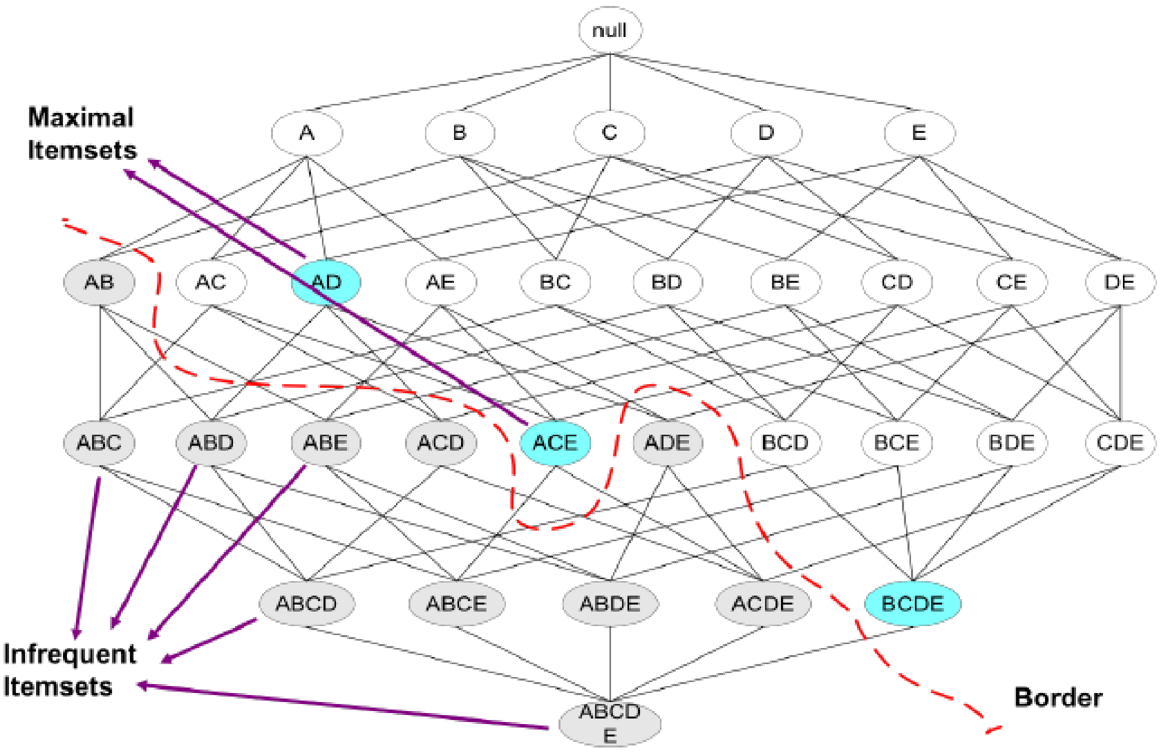
- Factors affecting complexity:
  - Choice of minimum support threshold:
    - lowering support threshold results in greater number of frequent itemsets;
    - this may reduce pruning and increase the maximum length of frequent itemsets;
    - number of complete reads of dataset given by the maximum length of frequent itemsets plus one.
  - Dimensionality of the dataset:
    - more space is needed to store support count of each item;
    - both computation and I/O costs may increase.
  - Size of database:
    - since *Apriori* makes multiple passes, run time may increase with number of transactions.
  - Average transaction width:
    - transaction width increases with denser datasets;
    - this may increase max length of frequent itemsets and traversals of data structures.

Compact representation of frequent itemsets

- Even after filtering on support, the number of frequent itemsets can be *very large*.
- Useful to identify a small representative of frequent itemsets.

Maximal frequent itemsets (MFI): smallest set of itemsets from which the frequent itemsets can be derived

- MFIs do not have any frequent immediate supersets;
- MFIs are near the border dividing frequent by non-frequent itemsets;
- provides a compact representation of all frequent itemsets;
- efficient algorithms explicitly find them without enumerating their subsets.

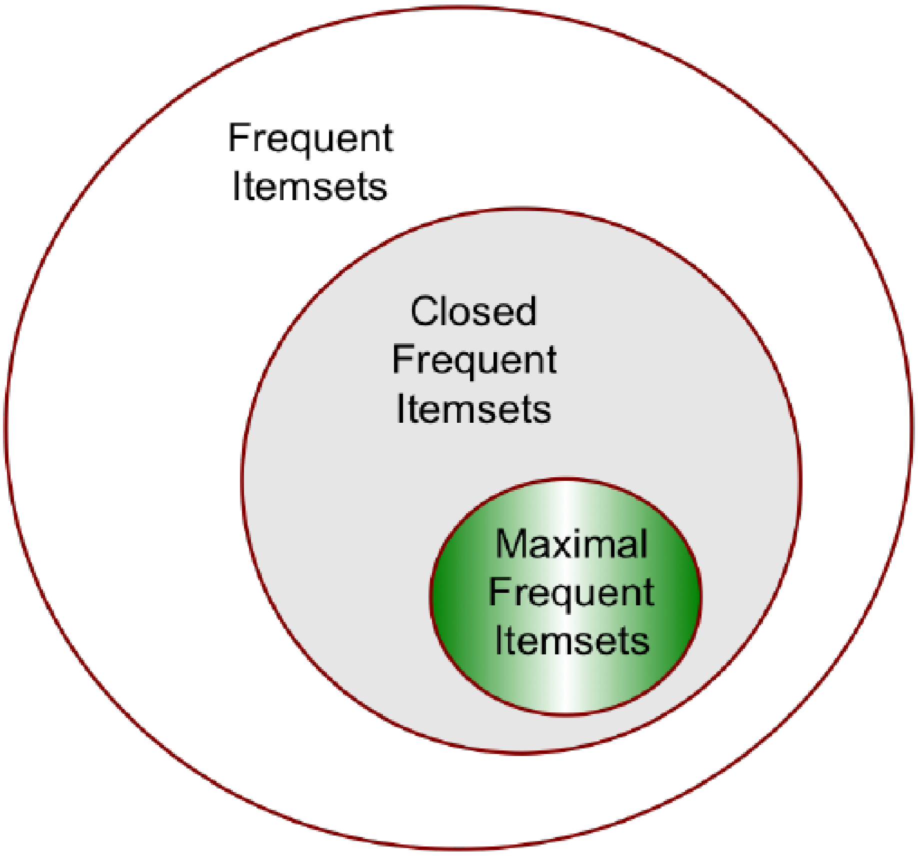


Closed itemsets: minimal representation of itemsets without losing support information

- some itemsets are redundant since they have the same support as their supersets;
- an itemset is closed if none of its immediate supersets has the same support.

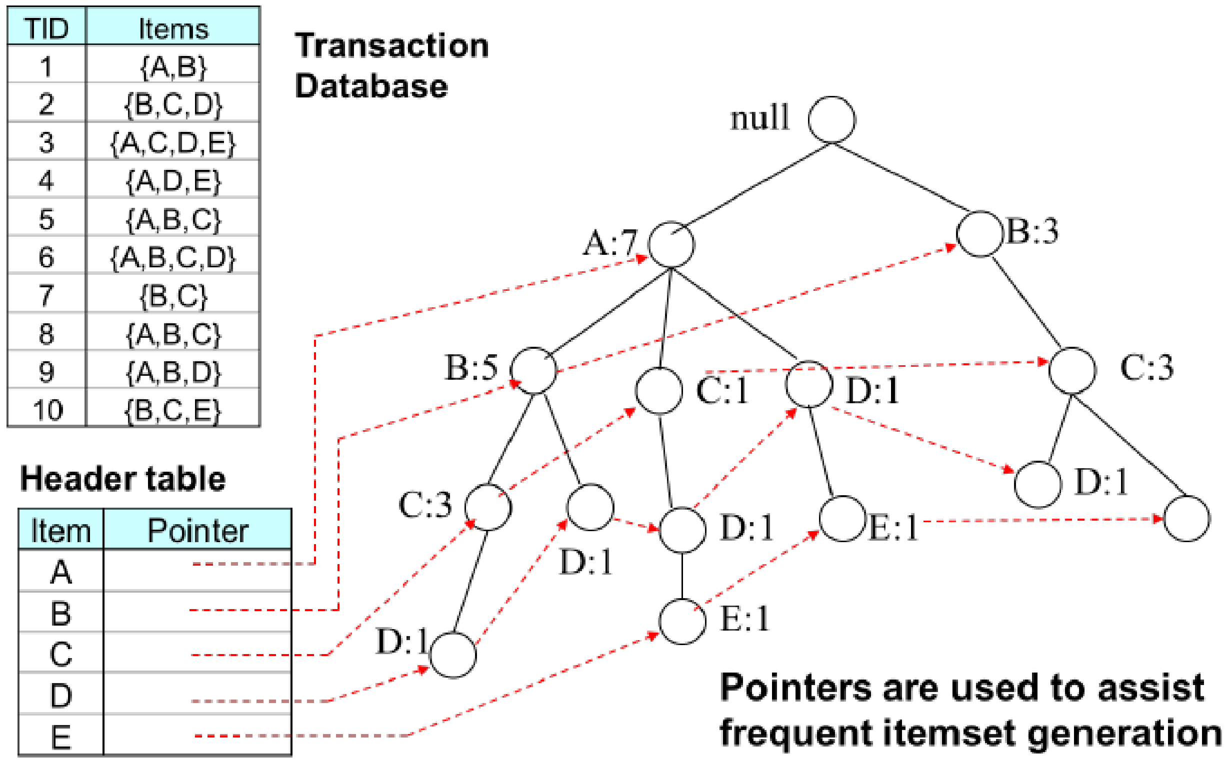
Closed frequent itemsets: general case of MFI

- $X \to Y$ is redundant if there exists $X' \to Y'$ s.t. support and confidence are the same, $X \subseteq X'$ and $Y \subseteq Y'$ (but equality cannot be in both);
- algorithms for efficient computation of closed frequent itemsets.



FP-growth algorithm:

- transforms the problem of finding long frequent patterns into looking for shorter ones and concatenating the suffix;
- uses a compressed representation of the database using **FP-tree**;
- once the FP-tree is built, is uses a recursive divide-and-conquer approach to mine frequent itemsets;
- construction process:
  - the first transaction generates a sequence of branches (one node per item), and each node has support 1;
  - if the next transaction starts with a node already in the tree, the sequence is added to that node;
  - otherwise, a new sequence is added to the root, and a *lateral* link connects nodes with the same label;



Conditional Pattern Base: a *sub-database* consisting of the set of frequent items co-occurring with the suffix pattern.

Obs.: FP-growth is more efficient than Apriori w.r.t. support threshold.

4.3. Rule Generation

The confidence of a rule can be computed from the supports:

$$conf(A \Rightarrow C) = \frac{sup(A \Rightarrow C)}{sup(A)}$$

for confidence based on pruning of rules it is sufficient to know the support of frequent itemsets.

Given a frequent itemset L:

- find all non-empty subsets $f \in L$ s.t. confidence of rule $f \Rightarrow (L - f)$ is not less than minimum confidence;
- if $|L| = k$ there are $2^k - 2$ candidate rules ($L \Rightarrow \emptyset$ and $\emptyset \Rightarrow L$ can be ignored).

Confidence of rules generated from the same itemset is anti-monotone w.r.t. the number of items on the right-hand-side of the rule (i.e. it decreases when an item is moved from the left hand to the right hand).

Rule generation in Apriori:

- candidate rule is generated by merging two rules sharing same prefix in rule consequent;
- the rule is pruned if the confidence of one of its subsets is too low.

Setting the minimum support threshold

Sometimes a single $\textbf{minsup}$ threshold may not be effective:

- if it's too low, it is computationally expensive and the number of itemsets is too large;
- if it's too high, itemsets involving interesting rare items would be lost.

Multiple Minimum Support

- $\text{MS}(i)$: minimum support for item i.
- $\text{MS}(i, j) = \min(\text{MS}(i), \text{MS}(j)) \Rightarrow$ no longer anti-monotone.
- Apriori needs to be modified s.t.:
    - L_1 : set of frequent items;
    - F_1 : set of items whose support is $\text{MS}(1)$, where $\text{MS}(1) = \min_i(\text{MS}(i))$;
    - C_2 : candidate itemset of size 2 generated from F_1 instead of L_1;
    - pruning is performed only if a subset is infrequent and it contains the first item of the itemsets which were merged.

Pattern evaluation

- Association rule algorithms tend to produce too many rules, many uninteresting and redundant (i.e. have same support and confidence).
- Interestingness measures could be used to prune/rank derived patterns.
- Given a rule $A \Rightarrow C$, the information needed to compute rule interestingness can be obtained by a contingency table.

Obs.: confidence can be misleading.

Statistical-based measures:

- **Lift**:

$$\text{lift}(A \Rightarrow C) = \frac{\text{conf}(A \Rightarrow C)}{\text{sup}(C)} = \frac{P(A, C)}{P(A)P(C)}$$

    - 1 for independence;
    - insensitive to rule direction;
    - ratio of true cases w.r.t. independence.

- **Leverage**:

$$\text{leve}(A \Rightarrow C) = \text{sup}(A \cup C) - \text{sup}(A)\text{sup}(C) = P(A, C) - P(A)P(C)$$

    - 0 for independence;
    - insensitive to rule direction;

- number of additional cases w.r.t. independence.

- **Conviction**:

$$\mathrm{conv}(A \Rightarrow C) = \frac{1 - \mathrm{sup}(C)}{1 - \mathrm{conf}(A \Rightarrow C)} = \frac{P(A)(1 - P(C))}{P(A) - P(A, C)}$$

  - infinite if rule is always true;
  - sensitive to rule direction;
  - ratio of expected frequency that A occurs without C if A and C were independent, divided by the observed frequency of incorrect predictions;
  - also called **novelty**.

- Higher support $\Rightarrow$ rules applies to more records.

- Higher confidence $\Rightarrow$ chance that rule is true for some record is higher.

- Higher lift $\Rightarrow$ chance that rule is a coincidence is lower.

- Higher conviction $\Rightarrow$ rule is violated less often than it would be if antecedent and consequent were independent.

Particular cases:

- High confidence rule can have small lift if both sides are very frequent.
- Low confidence rule can have high lift if both sides are very infrequent.

Properties of a good measure M:

- $M(A, B) = 0$ (or 1) if A and B are statistically independent.
- $M(A, B)$ increases monotonically with $P(A, B)$ when $P(A)$ and $P(B)$ remain unchanged.
- $M(A, B)$ decreases monotonically with $P(A)$ (or $P(B)$) when $P(A, B)$ and $P(B)$ (or $P(A)$) remain unchanged.

4.4. Multidimensional Association Rules

Mono-dimensional (intra-attribute):

- event: **transaction**;
- event description: items A, B, C are together in a transaction.

Multi-dimensional (inter-attribute):

- event: **tuple**;
- event description: attributes A has value a, B has value b and C has value c in a tuple.

Equivalence mono/multi

Multi–dimensional		*Mono–dimensional*
Schema: (TID,nationality,age,income)		
1, Italian, 50, low	→	1, {nat/Ita, age/50, inc/low}
2, French, 45, high		2, {nat/Fre, age/45, inc/high}
Schema: (TID,a?,b?,c?,d?)		
1, yes, yes, no, no	←	1, {a, b}
2, yes, no, yes, no		2, {a, c}

- In case of quantitative attributes, there are too many distinct values for a multi/mono transformation.
- Most software packages for association rules discovery do not deal with quantitative attributes ⇒ **discretization**, possibly *equi-frequency* or with *mono-dimensional clustering* for optimal covering of original value domains.

4.5. Multilevel Association Rules

- In a real scenario, frequently it is necessary to find a tradeoff between general and detailed reasoning ⇒ choice of the right level of abstraction.
- A common background knowledge is the organization of items in a hierarchy of concepts.

Support and Confidence in MAR

Support:

- from specialized to general:
  - support of rules generally increases;
  - new rules can become interesting;
- from general to specialized:
  - support of rules generally decreases;
  - support of rules can go under the threshold.

Confidence:

- a level change can influence confidence in any direction;
- if specialized rule have approximately the same confidence as the general one, it is redundant.

Algorithm:

- look for frequent itemsets at each level of abstraction, top-down (each level requires a new run of rule discovery algorithm);
- decrease support threshold in lower levels.