Because of barcode technology, stores can collect data (called Market Basket Data) on millions of transactions. Typically, a transaction represents the contents of a single shopping cart. A “1” in the table indicates a particular item that is purchased as part of a particular transaction.

Sets of Items vs. Sets of Transactions
- We are interested in finding sets of related items – those that typically appear together in a shopping cart.
- We use lowercase letters (a, b, c) to represent items and uppercase letters (A, B, C) to represent sets of items (itemsets).
- We are also interested in sets of transactions; we use T(A) to represent the set of all transactions that include every item in itemset A.
- Note that A ⊆ B implies that T(A) ⊇ T(B).
- This is because the more items we have in an itemset the fewer transactions there are that include all the items in the itemset.
- Example: there are lots of people who buy Poptarts, but many fewer who buy both Poptarts and lobster.

Measuring the Support of an Itemset
- An itemset A is only interesting if it occurs in a significant number of transactions.
- In other words, we want |T(A)| to be “large enough.”
- Notation: we use #(A) to represent |T(A)|; in other words, it’s the number of transactions that include all items of itemset A.
- The support of itemset A is defined as #(A) / #(∅).
- #(∅) is just the total number of all transactions.
- We say itemset A is supported if support(A) > s, where s is a constant that the user gets to choose.
- s is typically a small fraction of a percent.
- “A is supported” is another way of saying that the items of A appear together in a significant number of transactions.

Goal: Find all Supported Itemsets
- Outline of algorithm:
  - Choose a set of candidate itemsets.
  - Run through all the transactions and count how many times each candidate itemset appears.
  - Itemsets that appear sufficiently often are reported.
- This algorithm should work as long as our set of candidate itemsets is not too large.
- Strategy 1: Check all possible itemsets.
  - For 1000 items (not unusually large), there are 2^{1000} subsets.
  - 2^{1000} = (2^{10})^{100} = (10^2)^{100} = 10^{200}.
  - We can’t possibly check this many itemsets.

Finding Supported Itemsets II
- Strategy 2: Check only the itemsets that actually occur.
  - A single transaction might include, say, 40 items.
  - We can generate candidate itemsets by looking at subsets of these 40 items; we can do this for each transaction.
  - Number of subsets (for one typical transaction) = 2^{40} = (2^{10})^{4} = (10^2)^{4} = 1 trillion.
  - We can’t check this many itemsets either.

Finding Supported Itemsets III
- Strategy 3: Build candidate itemsets by adding one item at a time.
  - Note that itemset {a, b, c, d} is supported only if itemset {a, b, c} is supported.
    - In other words, once we find an unsupported itemset, adding an additional item will only make it less supported.
  - Algorithm for finding supported itemsets of size k+1:
    - Assume we already know L_k, the set of all supported itemsets of size k.
    - Generate new candidate itemsets of size k+1 by looking for transactions that contain some A ∈ L_k and then adding one more item to A from that transaction.
    - Run through all the transactions and count how many times each candidate itemset appears.
    - Report L_{k+1} = all candidates that appear sufficiently often.
Finding Supported Itemsets IV

- Strategy 4: Build candidate itemsets from subsets
  - If \{a, b, c, d\} is supported then so are \{a, b, c\}, \{a, b, d\}, \{a, c, d\}, and \{b, c, d\}

- Strategy 4 Algorithm: for finding supported itemsets of size \(k+1\)
  - Assume we already know \(L_k\), the set of all supported itemsets of size \(k\); further, \(L_k\) is in lexicographic order
  - Generate new candidate itemsets (of size \(k+1\)) by combining itemsets \(A\) and \(B\) in \(L_k\) where \(A\) and \(B\) are chosen such that
    - \(A\) is before \(B\) in lexicographic order and
    - \(A\) agrees with \(B\) except for the last item
  - [Pruning]: Reject a candidate itemset \(C\) if any size-\(k\) subset of \(C\) is missing from \(L_k\)
  - Report \(L_{k+1}\) = all remaining candidates that appear sufficiently often among all the transactions

Example: Strategy 4

- Let \(L_3\) = \{ \{a, b, c\}, \{a, b, d\}, \{a, c, d\}, \{a, c, e\}, \{b, c, d\}\}
- Generated candidates (before pruning): \{a, b, c, d\}, \{a, c, d, e\}
- Candidates after pruning: \{a, b, c, d\}
- Note that \{a, c, d, e\} was pruned because \{a, d, e\} is missing from \(L_3\)

Association Rules

- An association rule has the form \(A \rightarrow B\) where \(A\) and \(B\) are itemsets
- Each association rule has a confidence factor
- This indicates how often the rule appears to have “worked” in the dataset
- The confidence factor for \(A \rightarrow B\) is defined as
  - \(\text{Confidence} = \frac{|A \cup B|}{|A|}\)
  - In other words: Of all the times that \(A\) appears in transactions, what fraction also includes the items of \(B\)

Example: \(A \rightarrow B\)

- \(A\): \{bread, milk\}
- \(B\): \{eggs\}
- The rule is a way of expressing the idea that people who buy bread and milk are likely to also buy eggs
- Example confidence factor:
  - \(\frac{|\{bread, milk, eggs\}|}{|\{bread, milk\}|}\)

Using Association Rules

- For a rule \(A \rightarrow B\)
  - \(A\) is the antecedent and \(B\) is the consequent
- By finding association rules, we can answer useful questions
  - Find all rules with Coke as a consequent
    - What can be done to boost Coke sales?
  - Find all rules with bagels in the antecedent
    - What products might be affected if bagels are discontinued?
  - Find all rules with sausage in the antecedent and mustard as the consequent
    - What should be placed near sausage to encourage mustard sales?

Confidence vs. Support

- A rule with a high confidence factor is not necessarily useful
  - Example: Suppose there is exactly one transaction that includes both Poptarts and lobster and that transaction also includes pizza
  - The confidence factor for \(\text{Poptarts, lobster} \rightarrow \{\text{pizza}\}\) is \(\frac{1}{1} = 1\)
  - This rule has high confidence, but low support
  - The support for a rule \(A \rightarrow B\) is defined as \(\text{support}(A \rightarrow B) = \frac{|\{A \cup B\}|}{|\emptyset|}\)

Reporting the Useful Association Rules

- Suppose we already know
  - All supported itemsets
  - The value of \(\text{support}(C)\) for each supported itemset \(C\)
- Observe that if \(C\) is a supported itemset and \(C = A \cup B\) then
  - \(A \rightarrow B\) is an association rule that is supported,
  - \(A\) is supported (so we know \(\text{support}(A)\)), and
  - the confidence factor for \(A \rightarrow B\) is given by \(\frac{\text{support}(C)}{\text{support}(A)}\)
- Example: suppose the following itemsets are known to have the given support (measured in fractions of a percent)
  - 0.3 \(\{\text{bread}\}\)
  - 0.25 \(\{\text{eggs}\}\)
  - 0.15 \(\{\text{bread, milk}\}\)
  - 0.10 \(\{\text{bread, eggs}\}\)
  - 0.08 \(\{\text{bread, eggs, milk}\}\)
  - 0.05 \(\{\text{bread, eggs, milk, mustard}\}\)
- Find the association rules involving all of bread, eggs, and milk and determine the confidence factor for each rule