Itemset Mining

or co-occurrence mining

given a set of attributes and instance examples/points
find how often the attribute-values co-occur.

\[ \text{FP-tree} \]

\[ \text{Compressed} \]

\[ \text{Invert} \]

\[ \text{Compress} \]

1. ABDE
2. BCE
3. A\ BDE
4. A\ BcE
5. A\ BcDE
6. BCD

\[ \begin{align*}
\text{B} \rightarrow \text{C} \rightarrow \text{A} \rightarrow \text{C} \rightarrow \text{D} \\
6 & \quad 5 & \quad 4 & \quad 4
\end{align*} \]

For better compression
Projected trees

1) discard any infrequent items

2) Project on each item in reverse sorted order
   a) trace back the path from leaves for each item back to the root.
   b) propagate and add the counts
   c) remove item from tree

3) remove the current item from the tree before proceeding with the next item

If a single path remains then do

enumerate all subsets of the items on that path &
    tag on the prefix
lots of items, & they do not share common prefixes.
Problem: Information overload

\[ \rightarrow \text{ lots of frequent patterns} \]

Definition: enumerate ALL frequent patterns

\[ F = \{ X \subseteq I \mid \text{sup}(X) \geq \text{minsup} \} \]

\[ \text{compren/somnambre} \]

\[ \uparrow \]

\[ \text{semantic} \]

1) Maximal frequent itemsets: \( M \)

Set of all frequent itemsets, none of whose supersets are frequent

\[ X = \{A, B, C, D\} \quad \text{frequent} \]

\[ X + \in \]

\[ X + f \quad \ldots \quad \exists \text{ infrequent!} \]
$F_1 = \{A_1, B_6, C_4, D_4, G_5\}$

$|F_1| = 5$

$M = \{A_6, B_4, A_0, C_7, B_9, C_5, C_6, D_3\}$

$|M| = 2$

$A = \{A_6, B_4, C_7, B_9, A_0, C_5, C_6, D_3\}$

$C = \{B_6, A_6, A_0, C_4, B_9, A_0, C_5, B_9, C_6, A_0, C_5\}$

Q: Is $X$ frequent?

$ACD \leftarrow \text{no.}$

$BC \leftarrow \text{yes}$

$supp(BC) \geq 3$

$BD \leftarrow 5$

$? \leftarrow 5$

There exists no superset with the same support

$C$ is set of all closed itemsets

$C = \{X \in F \mid M \subseteq C \subseteq F\}$

$|C| = 8$

$frequent$ an arbitrary frequent set $X$ is

the larger value among those closed sets that contain $X$
1) Mine all the frequent items (minsup)
   \[ F = \{ A, b, e, s, A \_b, A \_e, b \_e, A B E, \ldots \} \]

2) \[
   \frac{A \_b \_e}{A \_b \_e} \]
   \( \Rightarrow \) \# of transactions

   \[ P(B|A) \leftarrow \begin{array}{c}
   A \Rightarrow B \\
   (\text{whenever } A \text{ happens})
   \end{array} \]

   \[ P(A|B) = \frac{\text{sup}(AB)}{\text{sup}(B)} = \frac{4}{6} = 66\% \]

   Evaluate rule using confidence (conditional probability)

   \[ P(X|Y) : \text{Prob of } X \text{ given } Y \]

   \[
   P(B|A) = \frac{P(AB)}{P(A)} = \frac{4/6}{4/6} = 1
   \]

   \[ B \Rightarrow A \]

   has 100% confidence since \( P(B|A) = 1 \)

(aka a strong rule)

because \( B \) always occurs !!!

\[ A \]

\[ C \]

\[ D \]

\[ E \]

\[ F \]

\[ G \]

\[ H \]

\[ I \]

\[ J \]
\[ A = \begin{array}{c}
X \\
\text{tail} \rightarrow \text{head}
\end{array}\]

\[ \frac{C}{D} \rightarrow C \]

\[ \frac{C}{\text{AB} \rightarrow C} \]

\[ \frac{A \rightarrow C}{A \rightarrow \text{AB} \rightarrow C} \]

\[ \frac{A \rightarrow C}{A \rightarrow \text{AB} \rightarrow C} \]

\[ \text{Confidence} = \frac{\text{sup}(A \rightarrow C)}{\text{sup}(A \rightarrow \text{AB} \rightarrow C)} \]

\[ = \frac{1}{4} = 1 \]

\[ \text{minconf} \leftarrow \text{threshold} \]

\[ \forall A \in F \]

generate & test

\[ X \Rightarrow Y, \ Y \subseteq A, \ Y \neq \emptyset, \ X \cap Y = \emptyset, \ X \cup Y = A \]

compute confidence \( Y \) above \text{minconf} then print

\( (X \Rightarrow Y, \ \text{sup}, \ \text{conf}) \)
Sequence:

- Temporal data (time, DNA sequence, ...)
- Positional data (text)

Frequent orderings

\[ \sigma_3 \]

Symbols = \{ A, C, G, T \}

Level-wise algorithm

- All possible gapped permutations
- All sub-sequence elements need not be consecutive

Prefix + all ordered pairs of less items

Frequency/support:
1. Add sequence
2. Add occurrence over all sequence

\[ \text{sup}(\sigma) = \frac{3}{6} \]

\[ \text{freq}(\sigma) = \frac{2}{6} \]

Max-gap constraint = 1
Mike Concanwhie sequence (substrings) $m = 3$

$s_1$: CAGAAGT 
$s_2$: TGAAGT
$s_3$: GAAGT

Window size $w$ 

$O(D \cdot n^2)$

$n$: any length $D$

$a$: seq in $D$

1: CAGAAGT $\leftarrow n$

2: CAG $\leftarrow n - 1$

3: CAGAAGT $\leftarrow n - 2$

$w$: 

$\sum_{w=1}^{n} \sum_{n-w+1}^{n} = O(n^2)$
Build a dfa structure: suffix tree

rolls
S1: CAGAAAGTF

extra sentinel character

O(n) time!!!

AC: 2, 5

j(p(AC)) = 2

O((D) - n) ← build the suffix tree

O(m) time we can answer any query.

m is length of query string, independent of dij "D".

+ time to output the answer.