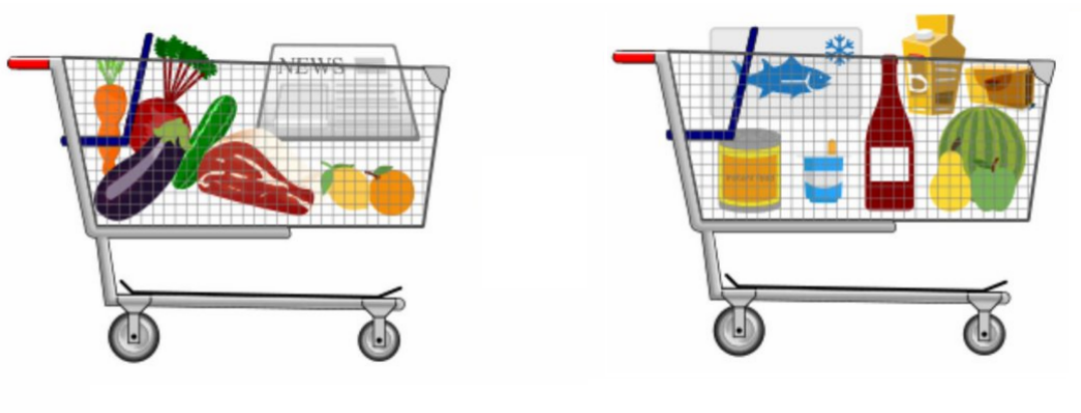
Ассоциативные правила

Виктор Китов

v.v.kitov@yandex.ru



Содержание

1 Решаемая задача

2 Основные меры качества

3 Алгоритм Apriori

4 Алгоритм FP-growth

5 Другие меры связи товаров

Решаемая задача

#транзакции	множество товаров	двоичное предст-ние
1	{хлеб, масло, молоко}	110010
2	{яйца, молоко, йогурт}	000111
3	{хлеб, сыр, яйца, молоко}	101110
4	{яйца, молоко, йогурт}	000111
5	{сыр, молоко, йогурт}	001011

Транзакционные данные

- Задача: извлечь все частые и достоверные правила.
  - например {яйца, молоко}=>{йогурт}
- Применения:
  - расположить йогурт вместе с хлебом и молоком
  - акции на йогурт покупающим хлеб и молоко
- Проблемы: множество товаров и транзакций велико.

Другие применения

Другие применения:

- анализ рыночных корзин (market basket analysis)
- медицинская диагностика
  - совстречаемость симптомов и болезней
- анализ ошибок в программах
  - совстречаемость событий в логах
- детекция событий (например, в службе реагирования)
  - совстречаемость событий и признаков

Расширение на другие типы данных

- Ассоциативные правила находятся только для бинарных признаков.
- Категориальные - бинаризовать через one-hot кодирование:

$$(\text{Gender=Male \& Date=7 марта}) \Rightarrow \text{цветы}$$

- Вещественные - в категориальные через дискретизацию:

$$(\text{Age} \in [60, 80]) \Rightarrow \text{тот самый чай}$$

Содержание

1 Решаемая задача

2 Основные меры качества

3 Алгоритм Apriori

4 Алгоритм FP-growth

5 Другие меры связи товаров

Ассоциативное правило

- $I = \{i_1, i_2, ...i_D\}$, $D \gg 1$ - все товары, ищем наборы из этих товаров
- $T = \{t_1, t_2, ...t_N\}$, $N \gg 1$ - все транзакции, t_i - подмножество I
- Ассоциативное правило $X \to Y$, где X, Y - наборы товаров, $X \cap Y = \emptyset$

Характеристики правил:

поддержка (support)	$P(X,Y) = \frac{\#\{X \cup Y\}}{N}$
уверенность (confidence)	$P(Y\|X) = \frac{\#\{X \cup Y\}}{\#\{X\}}$
значимость (lift)	$\frac{P(X,Y)}{P(X)P(Y)} = \frac{\#\{X \cup Y\} \cdot N}{\#\{X\}\#\{Y\}}$

Примеры расчётов

#транзакции	множество товаров	двоичное предст-ние
1	{хлеб, масло, молоко}	110010
2	{яйца, молоко, йогурт}	000111
3	{хлеб, сыр, яйца, молоко}	101110
4	{яйца, молоко, йогурт}	000111
5	{сыр, молоко, йогурт}	001011

- sup({яйца, молоко, йогурт})=
- sup({яйца, молоко})=
- conf({яйца, молоко}=>{йогурт})=
- lift({яйца, молоко}=>{йогурт})=

Примеры расчётов

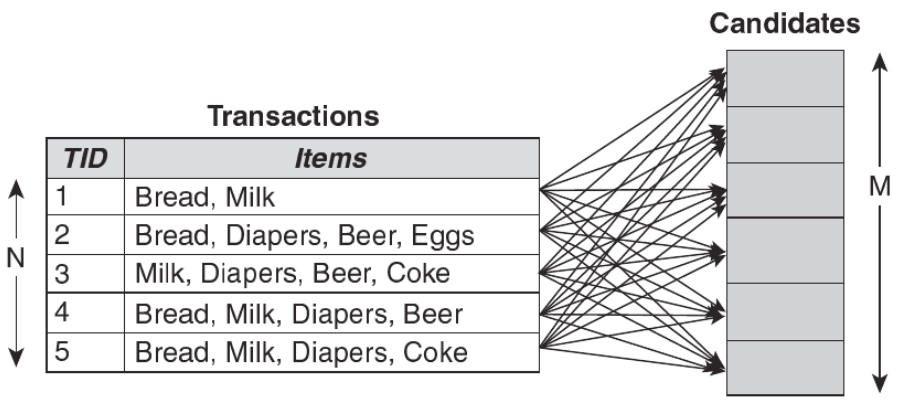
#транзакции	множество товаров	двоичное предст-ние
1	{хлеб, масло, молоко}	110010
2	{яйца, молоко, йогурт}	000111
3	{хлеб, сыр, яйца, молоко}	101110
4	{яйца, молоко, йогурт}	000111
5	{сыр, молоко, йогурт}	001011

- sup({яйца, молоко, йогурт})=2
- sup({яйца, молоко})=3
- conf({яйца, молоко}=>{йогурт})=2/3
- lift({яйца, молоко}=>{йогурт})=(2*5)/(3*3)=10/9

Нахождение правил

- Нахождение правил состоит из 2х этапов:
  1. генерация частых наборов ($support \geq minsup$)
  2. генерация уверенных правил по наборам ($confidence \geq minconf$)
- Первая задача вычислительно сложнее.
  - полный перебор: сложность $O(Nw(2^K - 1))$ для наборов длины k, #транзакций N и средней длины транзакции w.
  - решения: $\downarrow K$ (Apriori), $\downarrow w$ (hash-tree) $\downarrow$ число сравнений с транзакциями (FP-growth)

Полный перебор ($M = 2^k - 1$):



Применение для больших данных

- Если $|T| \gg 1$ можно искать правила по случайной $T' \subset T$.
- Возможные ошибочные правила (в контексте всей T):
  - ложно положительные: частые в T', редкие в T
    - решается перепроверкой по T
  - ложно отрицательные: редкие в T', частые в T
    - можно $\downarrow$строгость порогов в T', а потом перепроверять по T
    - если пороги слишком ослабим, получим слишком много кандидатов

Содержание

1 Решаемая задача

2 Основные меры качества

3 Алгоритм Apriori

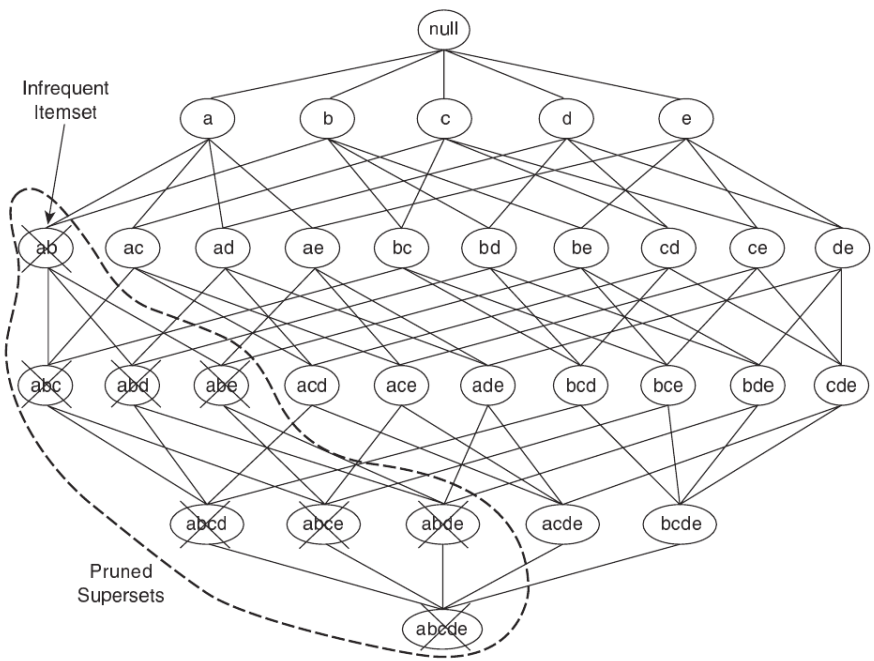
4 Алгоритм FP-growth

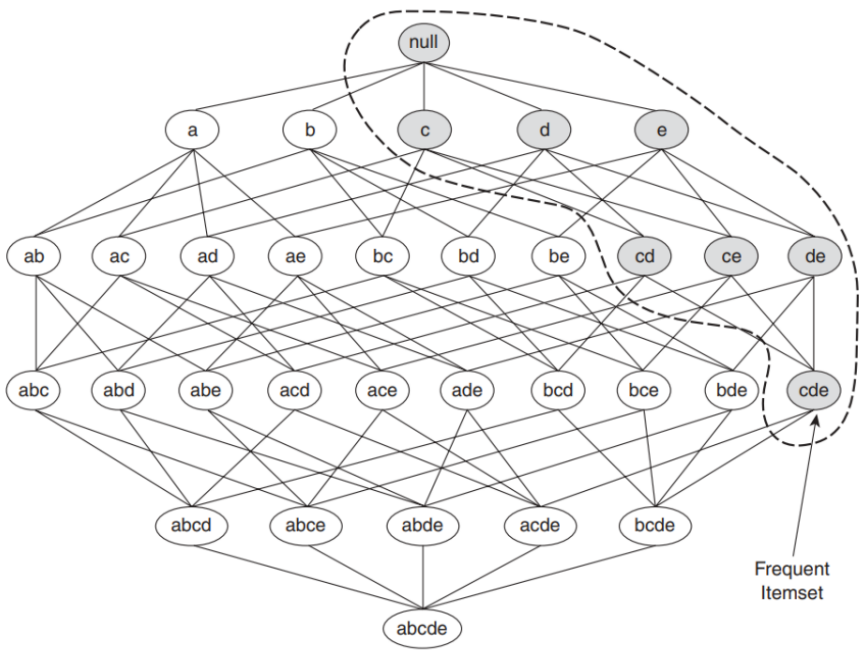
5 Другие меры связи товаров

Антимонотонность поддержки

- Поддержка удовлетворяет св-ву антимонотонности:
  - англ. downward closure property (DCP)

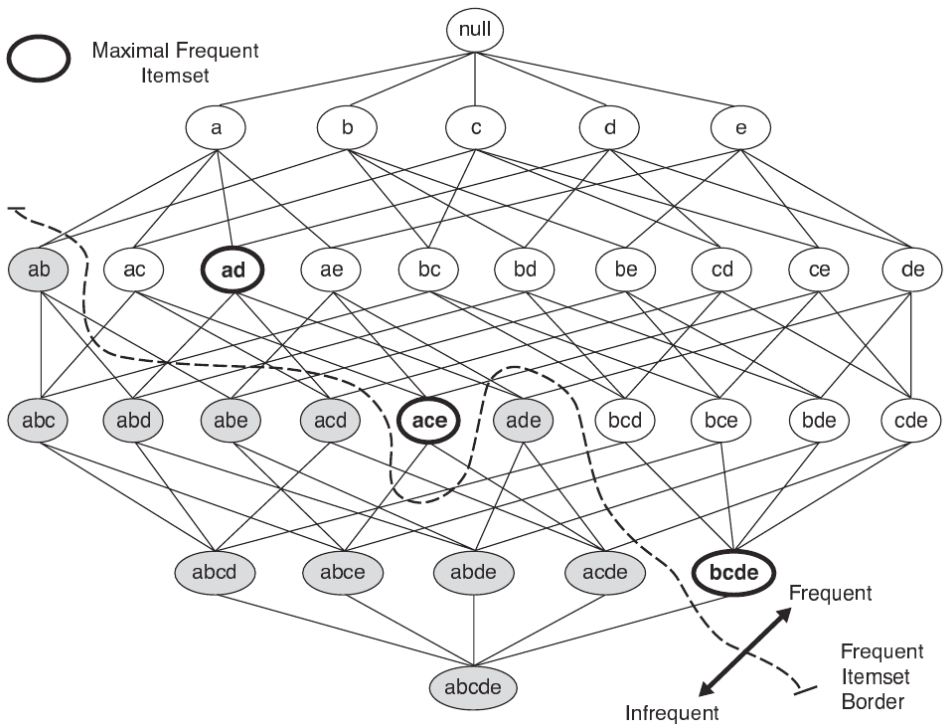
$$\forall X' \subset X \;\Rightarrow\; \sigma(X') \geq \sigma(X)$$





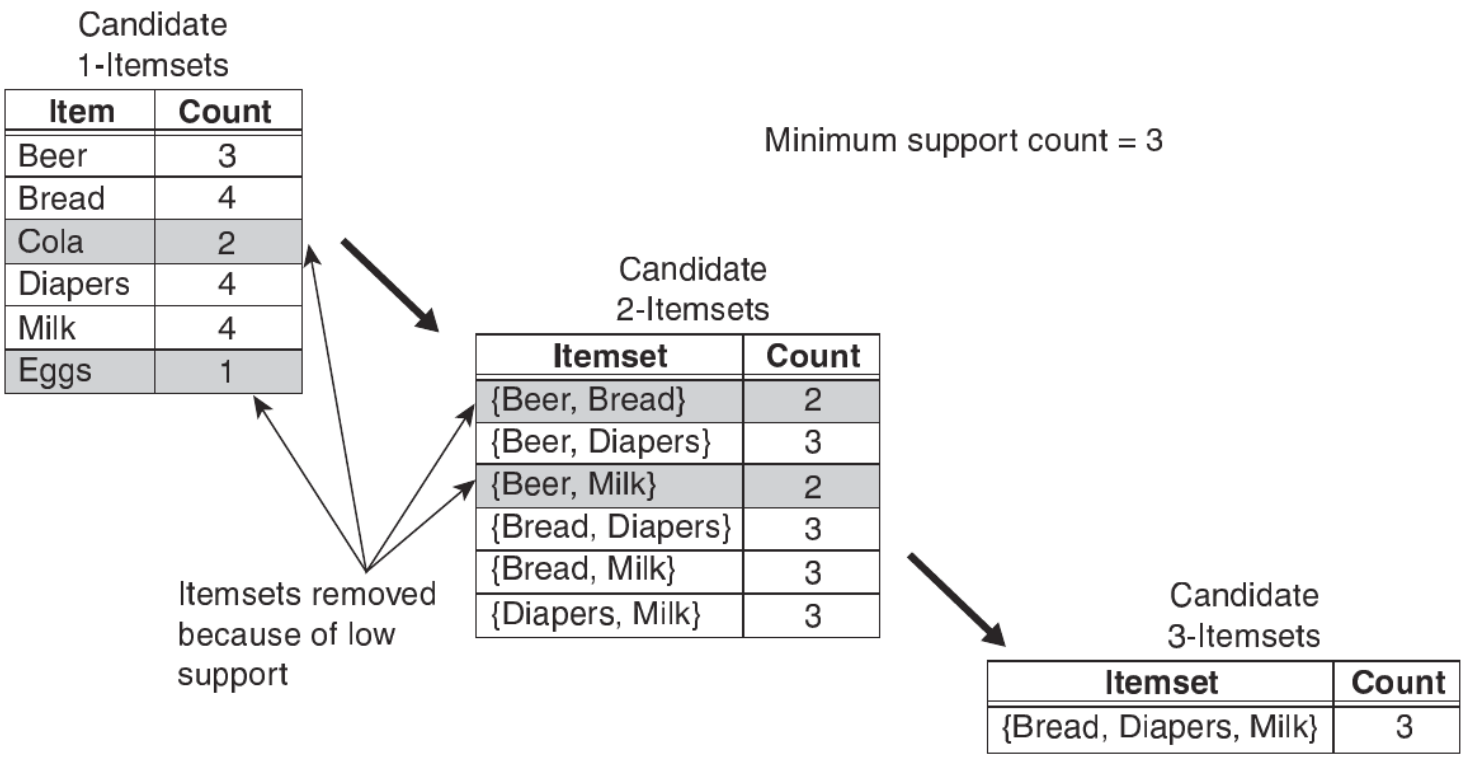
Эффективное хранение частых наборов

- Если набор частый, то все его поднаборы тоже частые.
- Можем хранить не все частые наборы, а только максимальные нерасширяемые (maximal frequent itemsets).
- Все поднаборы частых нерасширяемых - тоже частые. Но теряется информация о поддержке.



Генерация кандидатов в частые наборы в Apriori

- Тестируем $(K+1)$-наборы только из комбинаций частых K-наборов.



- Экономия перебора: $C_1^6 + C_2^6 + C_3^6 \to C_1^6 + C_1^4 + 1$.

Генерация C_{k+1} по F_k

- Генерация $\underline{F_k \times F_1}$: комбинация всех F_k и F_1
- возможны дублирования:

$$\{a,b,c\} = \{a,b\} + \{c\} = \{a,c\} + \{b\} = \{c,b\} + \{a\}$$

- поэтому комбинируем только по $\uparrow$ порядка

$$\text{OK:}\{a,b\} + \{c\} \quad \text{NO:}\{a,c\} + \{b\}, \{c,b\} + \{a\}$$

  - нужно предварительно упорядочить наборы $aaa, aab, aac, aba, abb, abc, ...$
- сложность $O(|F_k| \times |F_1|)$, недостаток $|F_1| \gg 1$

Генерация C_{k+1} по F_k

- $\underline{F_k \times F_k}$: объединяются всевозможные частые K-наборы
- избавляет от дублирования
  $\{a,b,c,d\} = \{a,b,c\}+\{d\} = \{a,b\}+\{c,d\} = \{a\}+\{b,c,d\}$
- могут генерироваться частые (K+2),(K+3),... наборы:

$$\text{OK}: \{a,b,c\} + \{a,b,d\} = \{a,b,c,d\}$$
$$\text{NO}: \{a,b,c\} + \{b,d,e\} = \{a,b,c,d,e\}$$
$$\text{NO}: \{a,b,c\} + \{d,e,f\} = \{a,b,c,d,e,f\}$$

- поэтому объединяем X и Y только при условии

$$x_1 = y_1,\ x_2 = y_2, \ldots x_{k-1} = x_{k-1}$$
$$\text{но } x_k \neq y_k$$

  - нужно предварительно упорядочить наборы
    $aaa, aab, aac, aba, abb, abc, \ldots$

Важность упорядочивания и перепроверка

- Без упорядочивания наборов появлялись бы дублирования:

$$abc + abd = abcd$$
$$abc + acd = abcd$$
$$abc + bcd = abcd$$
$$abd + acd = abcd$$
$$abd + bcd = abcd$$
$$acd + bcd = abcd$$

- С упорядочиванием $abcd$ генерируется однокртано как $abc + abd$.
- После склеивания K-наборов важно проверить, что новый K-набор частый.
  - проверить bcd для $abcd = abc + abd$
  - минимизируем #обращений к базе транзакций

Алгоритм Apriori

- F_k- частые k-наборы
- C_k - кандидаты в частые k-наборы
- T - БД транзакций.

Алгоритм Apriori.

$k = 1$
$F_1 = \{$все частые наборы из 1 товара$\}$
ПОКА $F_k \neq$ ПОВТОРЯТЬ:
    сегенрировать C_{k+1} комбинациями F_k
    исключить элементы C_{k+1}, содержащие редкие k-наборы
    определить по F_{k+1} по C_{k+1} подсчётом по T
    $k := k + 1$
ВЕРНУТЬ $\cup_{i=1}^{k} F_k$

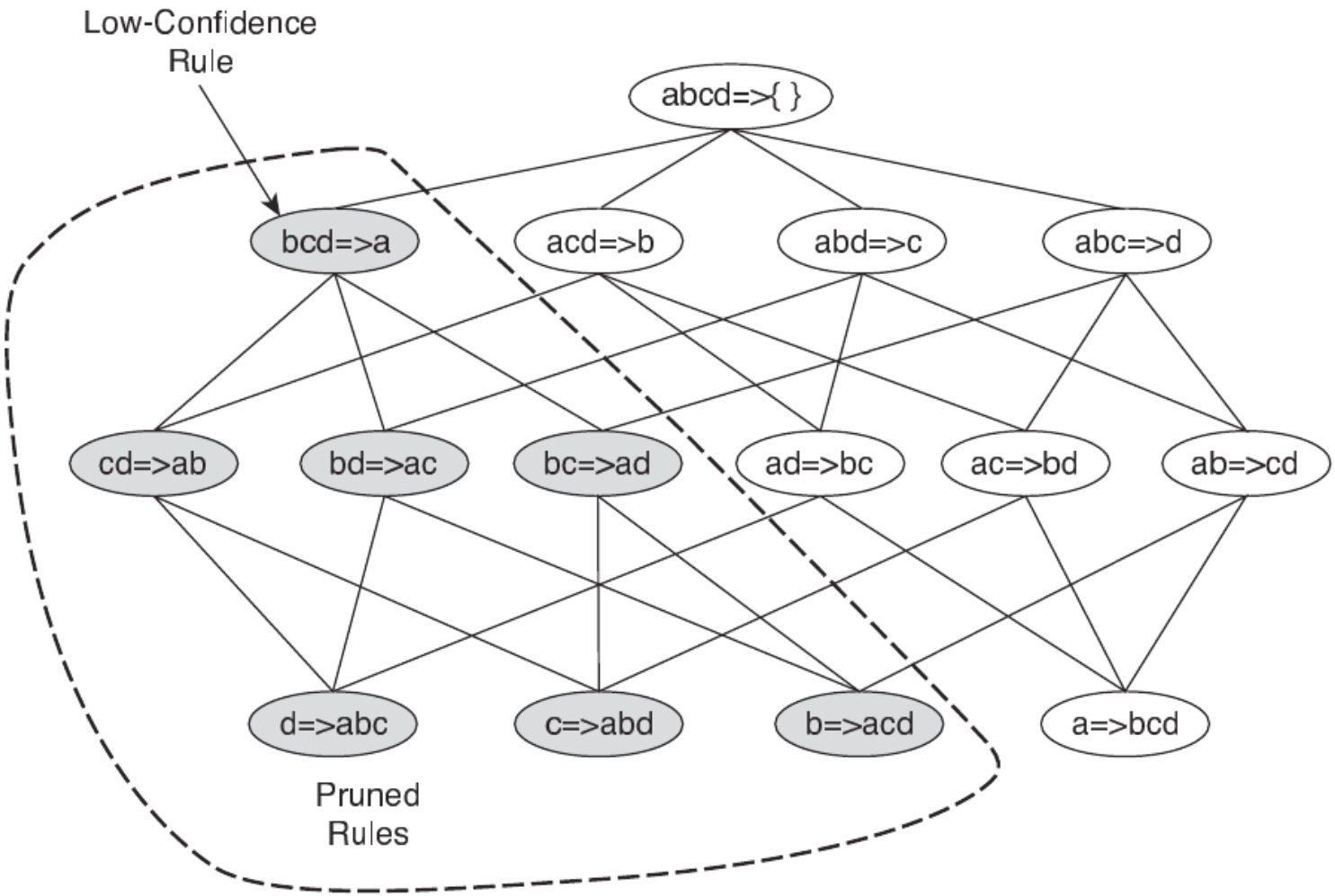
- Число полных проходов по T равно $k - 1$.
    - это длина самого длинного поднабора

Генерация правил

- Для k-набора существует $2^k - 2$ невырожденных правил
  - по набору X генерируем $Y \to X - Y \; \forall Y \subset X$
  - X - частый, значит и поднаборы $X - Y, Y$ - частые.
- Интересуют все правила с уверенностью выше порога.

- Оптимизация перебора правил: если $Y \to X - Y$ малой уверенности, то любое $Y' \to X - Y'$, $Y' \subset Y$ - тоже малой уверенности, т.к.
  - $Y' \subset Y \;\Rightarrow\; \sup(Y') \geq \sup(Y)$
  - $\text{conf}(Y \to X - Y) = \frac{\sup(X)}{\sup(Y)} \geq \frac{\sup(X)}{\sup(Y')} = \text{conf}(Y' \to X - Y')$

Эффективная генерация правил по {a,b,c,d}



Содержание

1 Решаемая задача

2 Основные меры качества

3 Алгоритм Apriori

4 Алгоритм FP-growth

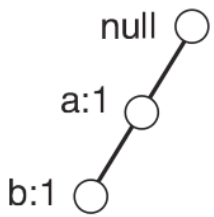
5 Другие меры связи товаров

FP-growth

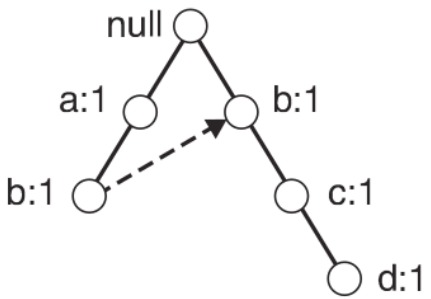
FP-growth алгоритм использует эффективную структуру данных для компактного представления наборов и их поддержек без дублирования.

Построение FP-дерева:

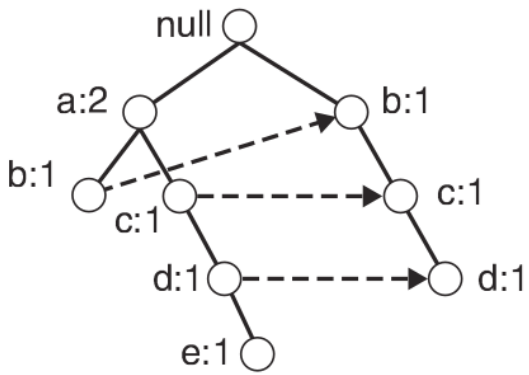
TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}



(i) After reading TID=1



(ii) After reading TID=2



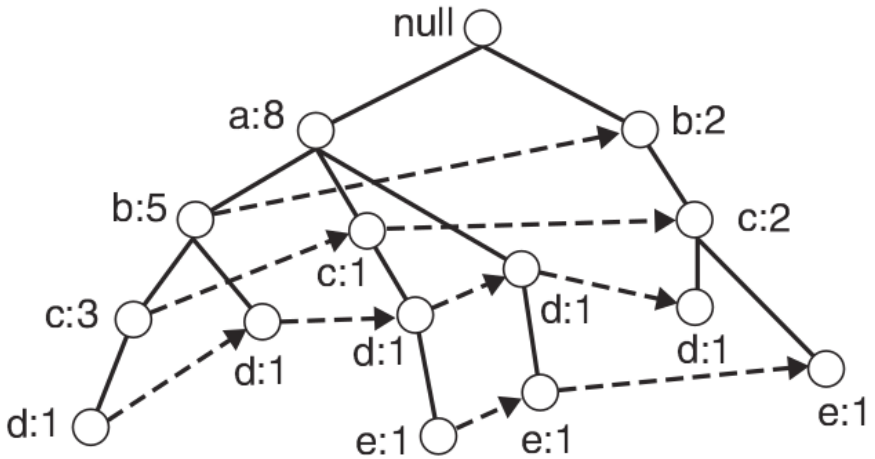
(iii) After reading TID=3

Полное FP-дерево

Полное FP-дерево

Transaction Data Set

TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}



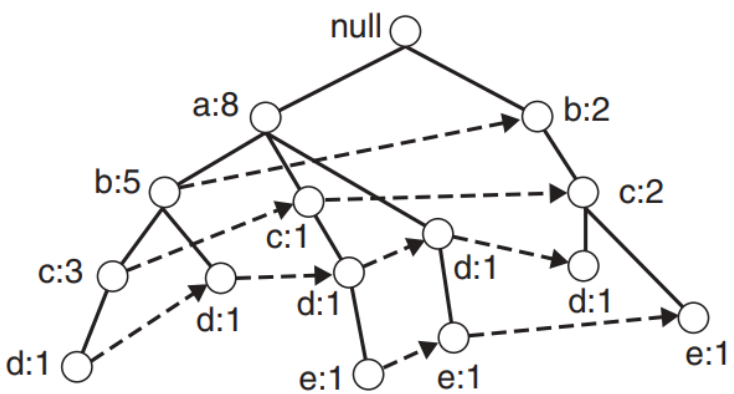
(iv) After reading TID=10

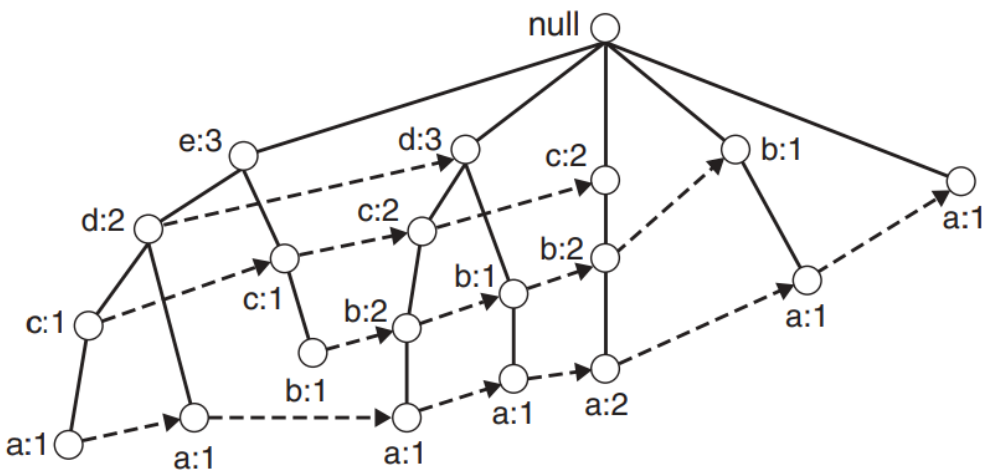
Сканируется FP-дерево, а не вся T

- поэтому эффективнее Apriori.

Упорядочивание товаров

- Для ↑ эффективности сканов товары, д. быть упорядочены по ↓поддержки.
  - требуется предварительный проход по T, чтобы оценить встречаемость отдельных товаров.
- Слева-упорядочивание по ↓поддержки (компактнее), а справа - по ↑:





Поиск частных наборов

- Частые наборы из 1 товара уже знаем.
- Сначала ищем частые наборы, заканчивающееся на {a},{b},{c},{d},{e}.
- В контексте каждого рекурсивно ищем расширения на 1 товар (по условному FP-дереву|e)

$$\{e\} \to \{de\}, \{ce\}, \{be\}, \{ae\}.$$

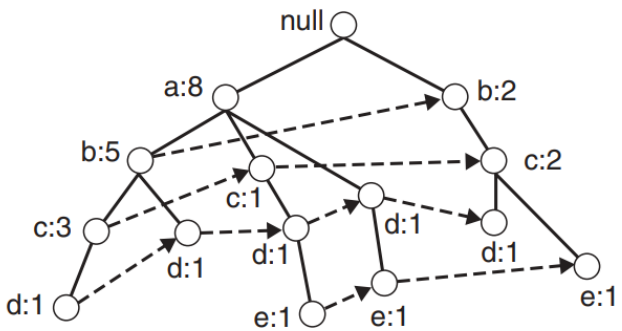
- К контексте каждой пары также рекурсивно ищем расширения на 1 товар (по условному FP-дереву|de)

$$\{de\} \to \{cde\}, \{bde\}, \{ade\}$$

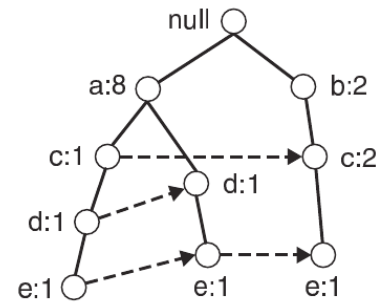
- И т.д. пока условые FP-деревья не станут пустыми
  - рекурсивная стратегия "разделяй и властвуй".

Визуализация путей, оканчивающихся на e,d,c,b,a

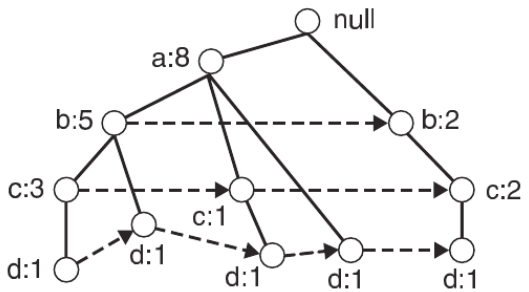
Для каждлого суффикса e,d,c,b,a пробуем его расширить на 1,2,... товара:



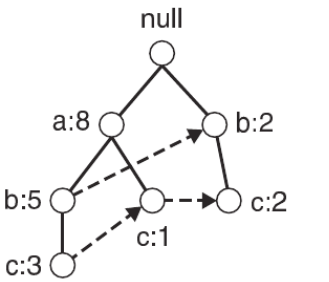
Suffix	Frequent Itemsets
e	{e}, {d,e}, {a,d,e}, {c,e},{a,e}
d	{d}, {c,d}, {b,c,d}, {a,c,d}, {b,d}, {a,b,d}, {a,d}
c	{c}, {b,c}, {a,b,c}, {a,c}
b	{b}, {a,b}
a	{a}



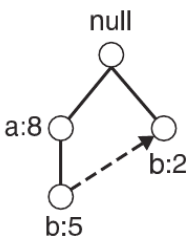
(a) Paths containing node e



(b) Paths containing node d



(c) Paths containing node c



(d) Paths containing node b

(e) Paths containing node a

- Но это еще не условное FP-дерево.

Построение условного FP-дерева

Условное FP-дерево транз-ций, заканчивающихся на e (рис.b):

1. Выделяем пути, оканчивающиеся на e (рис. a)
2. Пересчитываем поддержки всех внутренних узлов

   (суммируем снизу-вверх от листов "e" их поддержку)
3. Удаляем редкие товары ($\text{sup} < \text{sup}_{min}$)

   например, "b", т.к. только одна транзакция с "b", оканчивающаяся на "e" (bce)
4. Оставшиеся частые наборы из 2х элементов, заканчивающизся на "e" добавляем в список всех частых наборов

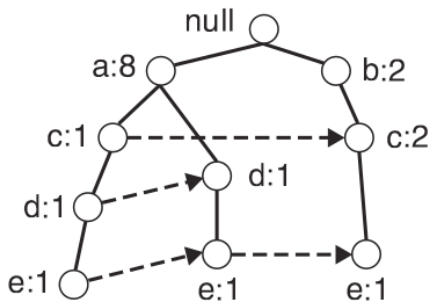
   например, для $\text{sup}_{min} = 2$ это de,ce,ae.
5. Удаляем листы с "e"

Для каждого частого набора (например, de) строим свое условное (FP-дерево|de), по которому далее рекурсивно ищем частые наборы из 3х товаров (ade).

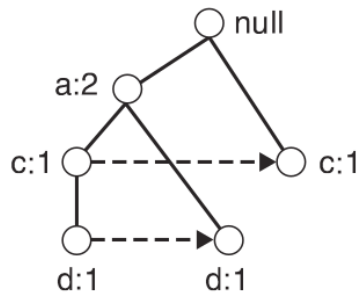
Построение условного FP-дерева

Transaction Data Set

TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

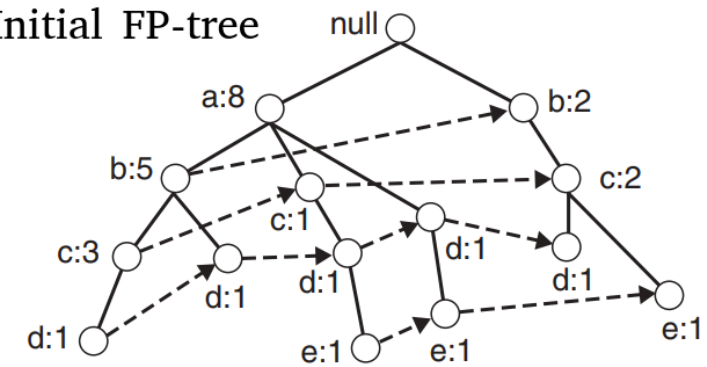


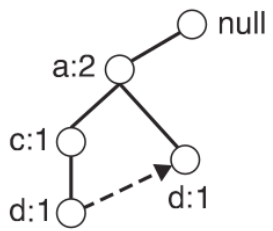
(a) Prefix paths ending in e



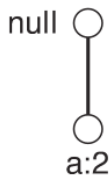
(b) Conditional FP-tree for e

Initial FP-tree





(c) Prefix paths ending in de



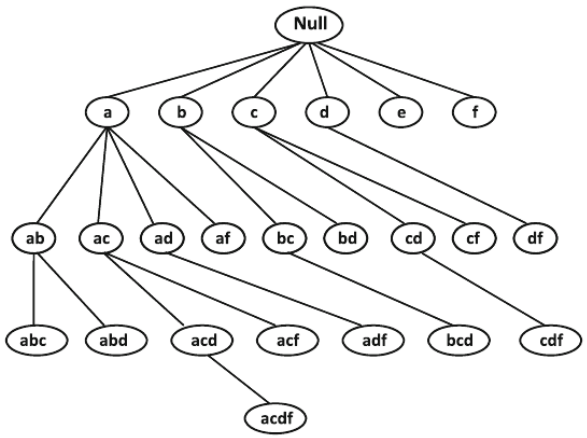
(d) Conditional FP-tree for de

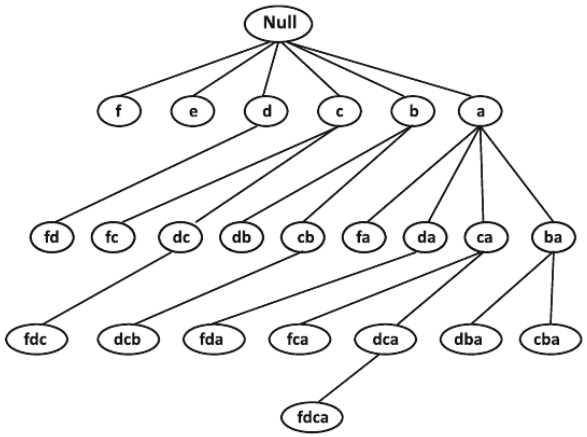
Алгоритм поиска частых наборов FP-growth

Algorithm *FP-growth*(FP-Tree of frequent items: $\mathcal{FPT}$, Minimum Support: $minsup$, Current Suffix: P)
begin
  if $\mathcal{FPT}$ is a single path
    then determine all combinations C of nodes on the path, and report $C \cup P$ as frequent;
  else (Case when $\mathcal{FPT}$ is not a single path)
  for each item i in $\mathcal{FPT}$ **do begin**
    report itemset $P_i = \{i\} \cup P$ as frequent;
    Use pointers to extract conditional prefix paths from $\mathcal{FPT}$ containing item i;
    Readjust counts of prefix paths and remove i;
    Remove infrequent items from prefix paths and reconstruct conditional FP-Tree $\mathcal{FPT}_i$;
    if $(\mathcal{FPT}_i \neq \phi)$ **then** *FP-growth*$(\mathcal{FPT}_i, minsup, P_i)$;
  end
end

Использована оптимизация: если все узлы условного FP-дерева лежат на одной линии, то можно найти оставшиеся более длинные частые наборы без рекурсии.

FP-growth vs. Apriori





- Apriori и FP-growth гарантируют нахождение всех частых наборов.
- Apriori использует поиск в ширину по префиксам.
- FP-growth использует поиск в глубину по суффиксам (окончаниям).
- FP-growth эффективнее Apriori:
  1. за счет использования компактного представления T
     - FP-дерево не содержит повторений для идентичных транзакций
  2. Из условное FP-дерева убирается уже выполненная информация о суффиксах
     - не нужно сверяться с полными транзакциями

Содержание

1 Решаемая задача

2 Основные меры качества

3 Алгоритм Apriori

4 Алгоритм FP-growth

5 Другие меры связи товаров

Учёт наличие и отсутствие товара

- Метрики support, confidence учитывают только присутствие товара.
- Более точные метрики учитывают как наличие, так и отсутствие товара.
- Связь {хлеб}<->{молоко} по
  - одновременному присутствию {хлеб, молоко}
  - одновременному отсутствию {¬хлеб, ¬молоко}
- Отсутствие товара - тоже значимый признак
  - поиск взаимосвязей с фактом отсутствия - negative pattern mining
- Не все удовлетворяют антимонотонности и могут быть эффективно вычислены.

Корреляция

$$\text{corr}(X,Y) = \frac{\mathbb{E}\{[X-\mathbb{E}X][Y-\mathbb{E}Y]\}}{\sigma(X)\sigma(Y)} = \frac{\mathbb{E}\{XY\}-\mathbb{E}X\cdot\mathbb{E}Y}{\sigma(X)\sigma(Y)}$$

- $X=\mathbb{I}[i\in t]$, $Y=\mathbb{I}[j\in t]$ для товаров i,j и случайной транзакции t.
- $\mathbb{E}X=\text{sup}\{i\}$, $\mathbb{D}X=\text{sup}\{i\}\cdot(1-\text{sup}\{i\})$

$$\text{corr}(i,j) = \frac{\text{sup}\{i,j\}-\text{sup}\{i\}\,\text{sup}\{j\}}{\sqrt{\text{sup}\{i\}(1-\text{sup}\{i\})\,\text{sup}\{j\}(1-\text{sup}\{j\})}}$$

Мера χ^2

- Тест χ^2 на независимость сл. вел. $X \in \{1,2,...n\}, Y \in \{1,2,...m\}$ по счётчикам их совместных значений:

$$\sum_{x=1}^{n}\sum_{y=1}^{m}\frac{\left(N\frac{N_{xy}}{N}-N\frac{N_x}{N}\frac{X_y}{N}\right)^2}{\left(N\frac{N_x}{N}\frac{X_y}{N}\right)^2}\to\chi^2\left((n-1)(m-1)\right)$$

- $X=\mathbb{I}[i\in t]$, $Y=\mathbb{I}[j\in t]$ и обобщается на k товаров
  - нужно суммировать по всем включениям-исключениям
- Удовлетворяет антимонотонности[1].
  - возможна Apriori-оптимизация
- Чем больше, тем выше зависимость.
  - не различает положит. и отрицат. зависимость

[1] Aggrawal. Data Mining: the textbook.

Мера интереса

- Мера интереса (interest ratio) - тестирует независимость бинарных сл. вел.:

$$I(X_1, ...X_K) = \frac{P(X_1 = 1, X_2 = 1, ...X_K = 1)}{P(X_1 = 1) \cdot P(X_2 = 1) \cdot ... \cdot P(X_K = 1)}$$

  $= 1$: независимость
  > 1: положительная зависимость
  $\in (0, 1)$: отрицательная зависимость

- В терминах товаров:

$$I(\{i_1, ...i_K\}) = \frac{\sup\{i_1, ...i_K\}}{\prod_{k=1}^{K} \sup\{i_k\}}$$

- Для редких товаров нерепрезентативна.

Симметричная уверенность

- Симметричная уверенность
  - для 2х наборов X и Y - степень взаимной связи:

$$\text{conf}_{sym}(X \Rightarrow Y) = \frac{\text{conf}(X \Rightarrow Y) + \text{conf}(Y \Rightarrow X)}{2}$$

  - Степень взаимосвязи между всеми поднаборами X:

$$\text{conf}_{sym}(X) = \text{Avg}\left(\{\text{conf}(X \setminus Z \Rightarrow Z)\}_{Z \subset X}\right)$$

Косинусная мера близости

- Косинусная мера
  - по близости между столбцами i, j (отвечающие товарам) в бинарной матрице чеков:

$$\text{cos-sim}(i,j) = \frac{\sup\{i,j\}}{\sqrt{\sup\{i\}} \cdot \sqrt{\sup\{j\}}} = \sqrt{\text{conf}\{i \Rightarrow j\}\,\text{conf}\{j \Rightarrow i\}}$$

- Близость Жаккарда:
  - обозначим $S_i = \{n : i \in t_n\}$ - множество чеков, содержащих i-й товар.

$$J(S_1, \dots S_K) = \frac{\left|\cap_{k=1}^K S_k\right|}{\left|\cup_{k=1}^K S_k\right|}$$

  - удовлетворяет антимонотонности $J(S_1, \dots S_K, S_{K+1}) \leq J(S_1, \dots S_K)$
    - может эффективно находиться через Apriori

Collective strength

- Для набора X определим "нарушение": в транзакции t часть товаров X присутствует, а часть - нет.
- $p_i = \#[i \in t]/N$ - частота встречи товара i в транзакциях
- $v(X)$ - частота нарушений (предп. независимость встреч)

$$\mathbb{E}\{v(X)\} = 1 - \underbrace{\prod_{i\in X} p_i}_{\text{все встр-сь}} - \underbrace{\prod_{i\in X}(1-p_i)}_{\text{никто не встр-ся}}$$

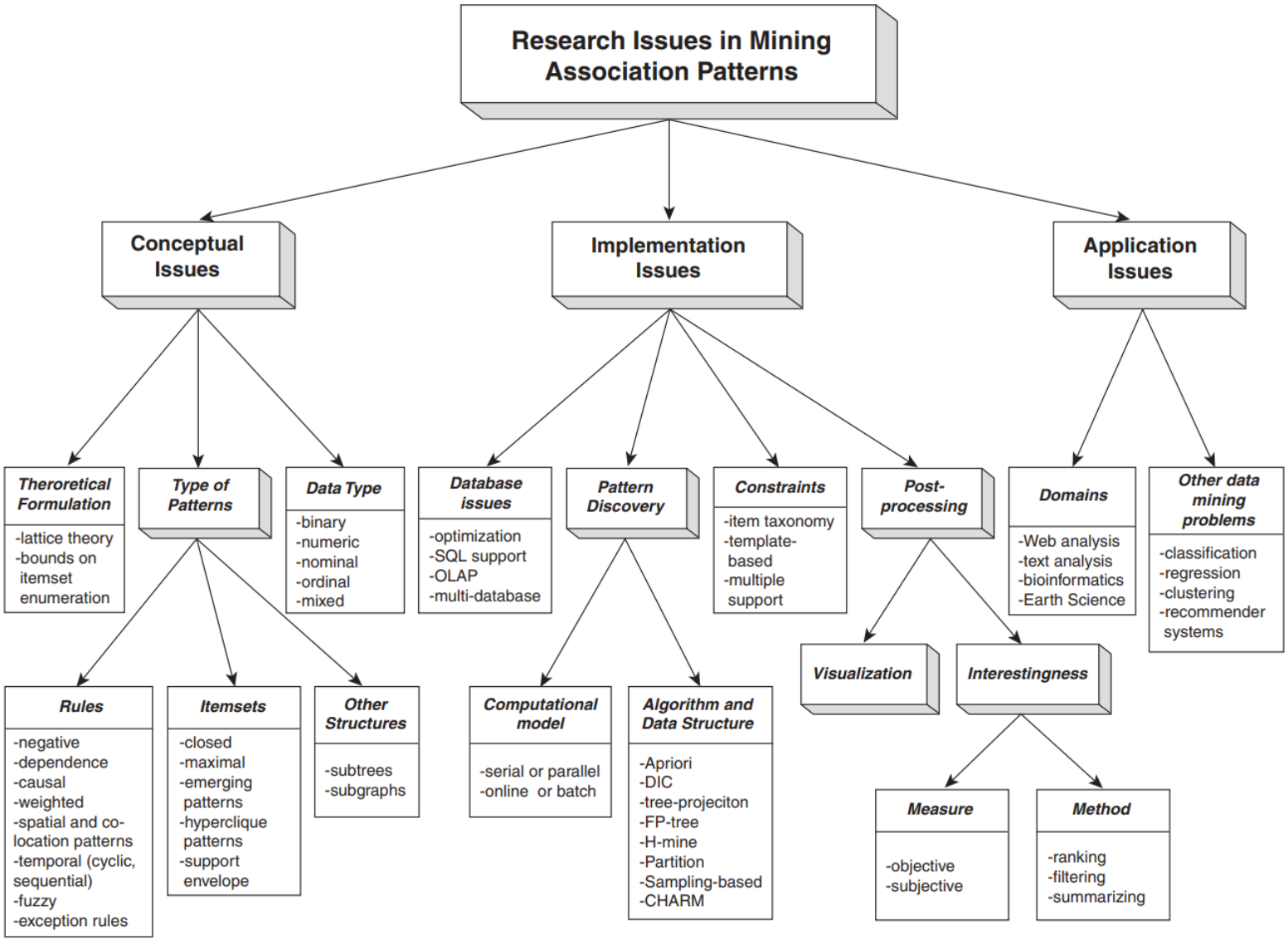
- По смыслу $v(X) = p(\text{успех})$, $1 - v(X) = p(\text{неудача})$
- Мера Collective Strength связи товаров в X

$$CS(X) = \frac{1 - v(X)}{1 - \mathbb{E}\{v(X)\}} \cdot \frac{\mathbb{E}\{v(X)\}}{v(X)} \geq 0$$

0: полностью отрицательная связь

$+\infty$: полностью положительная связь

Темы исследований по ассоциативным правилам



Заключение

- Поиск ассоциативных правил состоит из 2 шагов:
  - обнаружение наборов с высокой поддержкой
    - Apriori, FP-growth.
  - генерация по ним правил с высокой уверенностью
- Для большой T - искать по подвыборке.
- Кроме бинарных данных (факт покупки) можно применять к категориальным и вещественным признакам.
- Развернутый обзор темы.