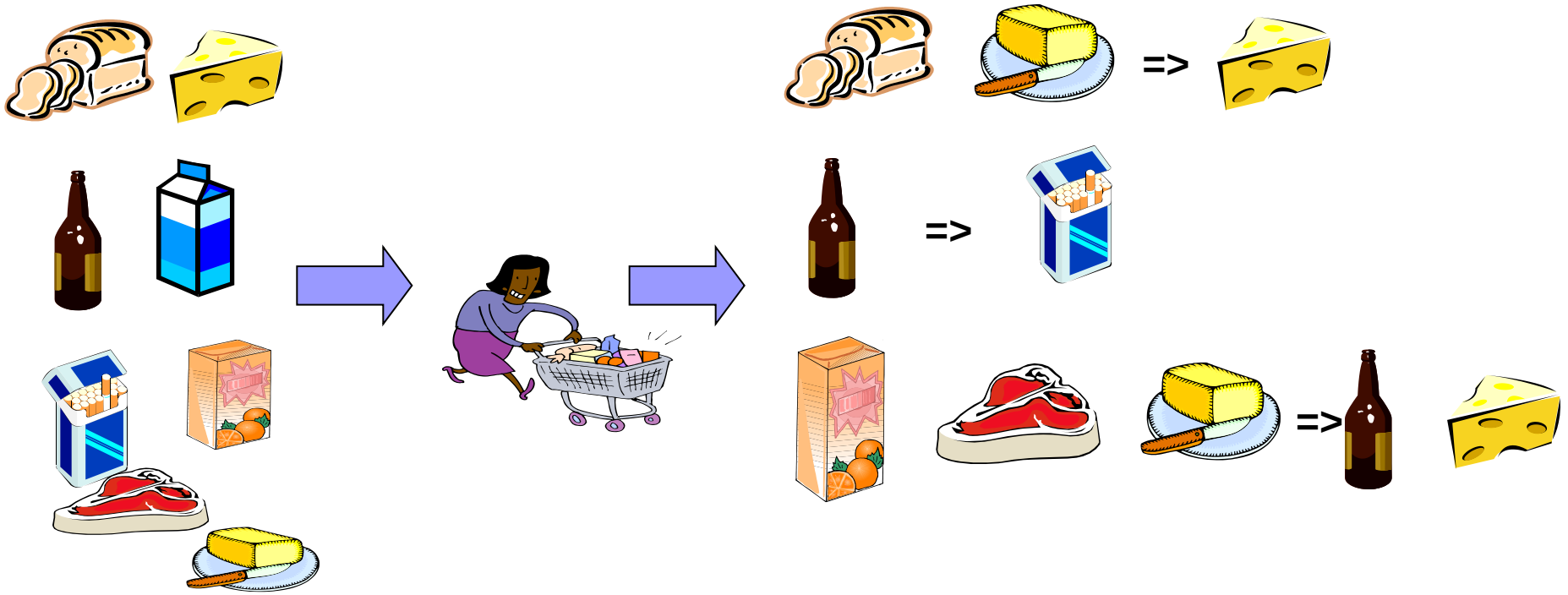
Лекция 20: Поиск ассоциативных правил

Типовая прикладная задача: анализ «корзины покупателя»

Ассортимент супермаркета

Интересные правила



Задача Определить интересные правила в предпочтениях покупателей при выборе товара

Ассоциативный анализ

- Правила с семантикой:
  - в s% случаях ЕСЛИ верно A_1, A_2 , …, A_k, ТО с достоверностью c будет верно B_1, B_2, …, B_m

$$\mathbf{A_1 \wedge A_2 \wedge \ldots \wedge A_k => B_1 \wedge B_2 \wedge \ldots \wedge B_m}$$

  - где A_1, A_2 , …, A_k, B_1, B_2, …, B_m – (различные!) предикаты,
  - s – поддержка (support), c – достоверность (confidence)
- Основная задача:
  - найти все интересные правила, с заданными ограничениями по s и c (возможно задание дополнительных ограничений на предикаты и сами правила)
- Основной математический аппарат:
  - дискретная математика, математическая логика, комбинаторная оптимизация (на основе метода «ветвей и границ» - вариации полного перебора с отсевом подмножеств допустимых решений, заведомо не содержащих оптимальных решений).

Ассоциативный анализ

- Прикладные задачи:
  - «Экономические»: анализ корзины, маркетинг
  - «Безопасность» и Web usage mining: модели поведения пользователя
  - Text mining: поиск ключевых слов, характеристик и тематик
  - Биоинформатика, медицина
- Задачи анализа:
  - Поиск самих правил
  - Поиск исключений (из правил)
  - Выделение признаков (на основе правил)
  - Классификация и прогнозирование (на базе правил)

Ассоциативный анализ

- Тип моделей:
  - Как правило, «описательный» (descriptive) Data mining => одна из задач - наглядное представление правил
- Тип обучения:
  - «без учителя» (unsupervised) => тренировочный набор не размечен
- Типы правил:
  - **Булевы**
  - Числовые – нужна дискретизация, интервалы как булевы предикаты
  - Иерархические – если определена иерархия для значений атрибутов
  - Временные – как правило, семантика «в s случаях если произошло A и B, то потом случится C и D с вероятностью c»)
  - Пространственные – предикаты определяют пространственные связи между объектами, например «рядом», «далеко» и т.п.

Булевы ассоциативные правила (определение через транзакции)

- Базовые определения:
  - $I = \{i_1, i_2, \ldots, i_n\}$ – множество атрибутов/элементов/бинарных признаков;
  - D – множество транзакций T, каждая T – множество элементов из I
- Эпизоды:
  - Транзакция T **поддерживает** набор (эпизод) $X \subseteq I$, если $X \subseteq T$
  - **Поддержка** эпизода X равна s, если s% транзакций из D содержат X
  - Набор X **часто встречаемый**, если $\text{support}(X) = p(X)$ выше порога
- Правило:
  - **Ассоциативное правило** - импликация $X \Rightarrow Y$, $X \subseteq I$, $Y \subseteq I$ и $X \cap Y = \emptyset$
  - Правило $X \Rightarrow Y$ имеет **поддержку (support) s**, если s% транзакций из D содержат X и Y: $\text{support}(X \Rightarrow Y) = p(X, Y)$
  - Правило $X \Rightarrow Y$ имеет **достоверность (confidence) c**, если c% транзакций из D, содержащих X, также содержат Y:

$$\text{confidence}(X \Rightarrow Y) = p(Y|X) = \frac{p(X,Y)}{p(X)} = \frac{\text{support}(X \cup Y)}{\text{support}(X)}$$

  - Правило **интересное**, если превышены заданные пороги на достоверность и поддержку

Булевы ассоциативные правила (определение через признаки)

- Базовые определения:
  - X – пространство объектов;
  - $X^l = \{x_1, \dots, x_l\} \subset X$ – обучающая выборка
  - $\mathcal{F} = \{f_1, \dots, f_n\}, f_j: X \to \{0,1\}$ – бинарные признаки (items)
- Эпизоды:
  - Каждому подмножеству $\varphi \subseteq \mathcal{F}$ соответствует конъюнкция

$$\varphi(x) = \Lambda_{f \in \varphi} f(x), x \in X$$

  - Если $\varphi(x) = 1$, то «признаки из φ совместно встречаются у x».
  - Частота встречаемости (поддержка, support) φ в выборке X^l

$$v(\varphi) = \frac{1}{l} \sum_{i=1}^{l} \varphi(x_i)$$

  - Если $v(\varphi) \geq \delta$, то «набор φ частый» (frequent itemset)
  - Параметр δ – минимальная поддержка, MinSupp

Булевы ассоциативные правила (определение через признаки)

- **Интересное** ассоциативное правило $\varphi \to y$ – это пара непересекающихся наборов $\varphi, y \subseteq \mathcal{F}$ таких, что:
  - наборы φ и y совместно часто встречаются, $v(\varphi \cup y) \geq \delta$;
  - если встречается φ, то часто встречается также и y,

    $v(y|\varphi) \equiv \frac{v(\varphi \cup y)}{v(\varphi)} \geq \varkappa$, где $v(y|\varphi)$ – значимость (confidence) правила.
  - Параметр $\varkappa$ – минимальная значимость, MinConf.
- **Логическая** интерпретация:
  - Предикат $\varphi(x)$ – логическая закономерность класса $c \in Y$

    $$\text{support}(\varphi) = \frac{p(\varphi)}{l} \geq \delta; \quad \text{confidence}(\varphi) = \frac{p(\varphi)}{p(\varphi)+n(\varphi)} \geq \varkappa$$
  - $p(\varphi) = \#\{x_i \in X^l : \varphi(x_i) = 1 \text{ и } y(x_i) = c\}$ +примеры класса c
  - $n(\varphi) = \#\{x_i \in X^l : \varphi(x_i) = 1 \text{ и } y(x_i) \neq c\}$ -примеры класса c
  - Для «$\varphi \Rightarrow y$» возьмем целевой признак $y(x) = \Lambda_{f \in y} f(x)$.
  - Тогда $v(\varphi \cup y) \equiv Supp_1(\varphi) \geq \delta$; $\frac{v(\varphi \cup y)}{v(\varphi)} \equiv Conf_1(\varphi) \geq \varkappa$

Пример

D=

t	Хлеб	Кефир	Пиво	Чипсы
1	1	0	0	0
2	1	1	0	0
3	0	1	1	1
4	0	1	1	1
5	1	1	0	0
6	1	0	1	0
7	1	1	1	1
8	1	0	0	0
9	0	0	1	0
10	0	0	1	0

I={Хлеб, Кефир, Пиво, Чипсы}
supp(Хлеб) = 60%
supp(Кефир) = 50%
supp(Пиво) = 60%
supp(Чипсы) = 30%

Пример правила: Пиво=>Чипсы
supp(П=>Ч) = 30%
conf(П=>Ч) = 50%

Задача: Найти правила с параметрами:

minsupp = 30%,

minconf = 60%

Критика Support и Confidence

- Пример: (Aggarwal & Yu, PODS98)
  - Среди 5000 студентов
    - 3000 играют баскетбол
    - 3750 любят черный хлеб
    - 2000 и то и другое

	basketball	not basketball	sum(row)
bread	2000	1750	3750
not bread	1000	250	1250
sum(col.)	3000	2000	5000

  - *basketball* $\Rightarrow$ *bread* [40%, 66.7%] вводит в заблуждение, поскольку процент любителей хлеба 75% выше support 66.7%.
  - *basketball* $\Rightarrow$ *not bread* [20%, 33.3%] более полезное, хотя support and confidence ниже
- Пример:
  - X и Y: положительно коррелированы,
  - X и Z, отрицательно коррелированы
  - support и confidence больше у X=>Z

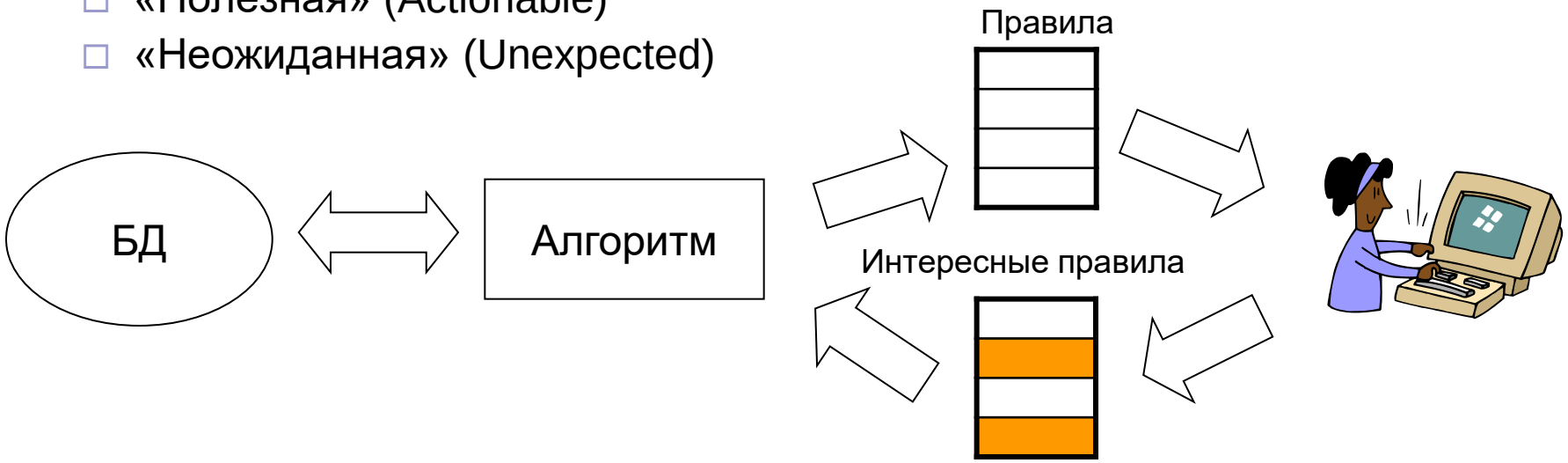
X	1	1	1	1	0	0	0	0
Y	1	1	0	0	0	0	0	0
Z	0	1	1	1	1	1	1	1

Rule	Support	Confidence
X=>Y	25%	50%
X=>Z	37,50%	75%

- Нужна новая **мера зависимости**

Интересность

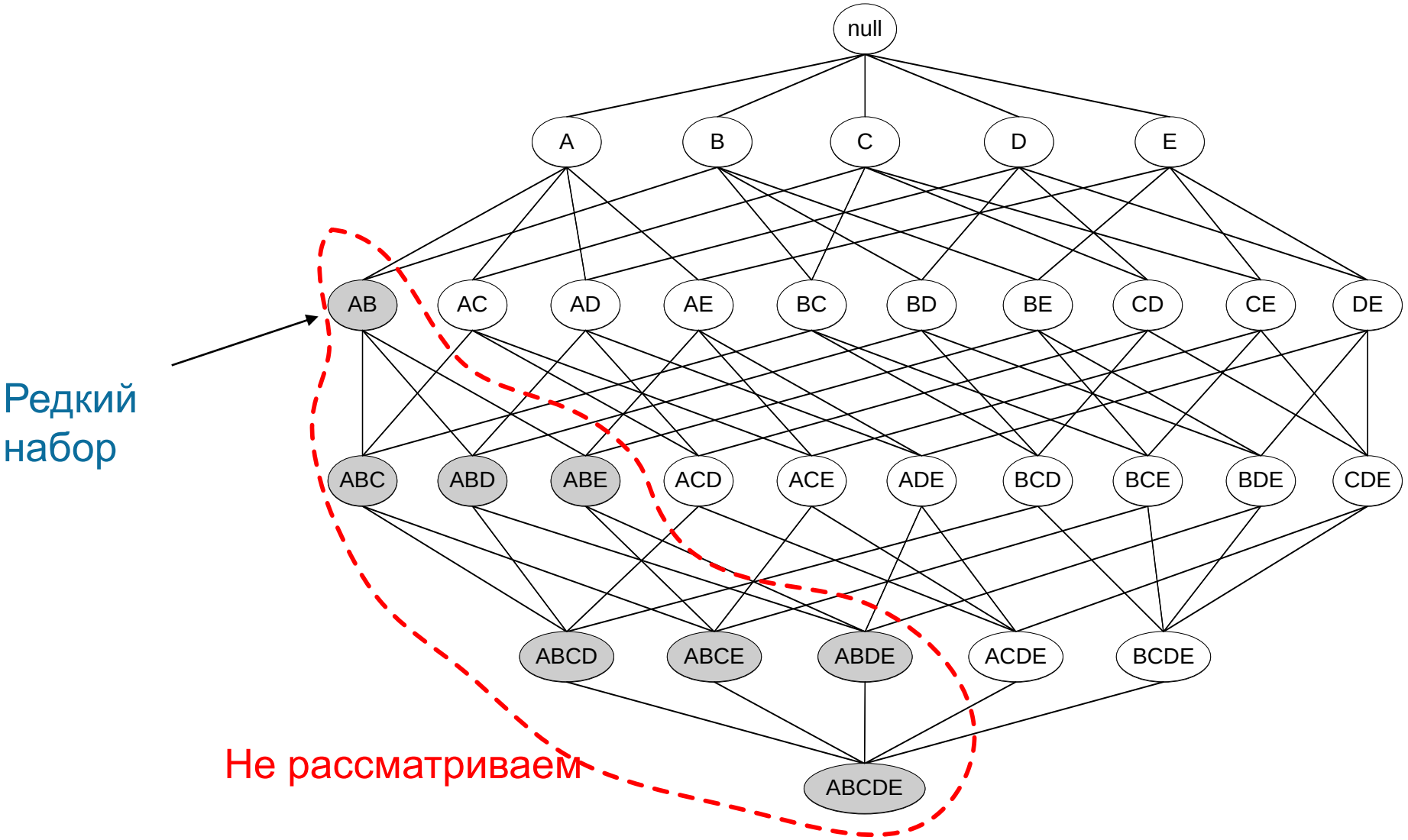
- Объективная:
  - $\text{support}(X \Rightarrow Y), \text{confidence}(X \Rightarrow Y), \text{generality}(X \Rightarrow Y) = \text{P(Y)}$
  - $\text{lift}(X \Rightarrow Y) = \frac{P(Y|X)}{P(Y)} = \frac{\text{confidence}(X \Rightarrow Y)}{\text{generality}(X \Rightarrow Y)}$
  - $\text{RI}(X \Rightarrow Y) = \text{P(Y|X)} - \text{P(Y)} = \text{confidence}(X \Rightarrow Y) - \text{generality}(X \Rightarrow Y)$
  - $J(X \Rightarrow Y) = P(Y)\left[P(Y|X)\log_2\frac{P(Y|X)}{P(Y)} + (1 - P(Y|X))\log_2\frac{1-P(Y|X)}{1-P(Y)}\right]$
- Субъективная (на основе информации, заданной экспертом)
  - «Полезная» (Actionable)
  - «Неожиданная» (Unexpected)



Алгоритм Apriori

- Основной принцип (анти-монотонность):
  - Любое подмножество часто встречаемого набора является часто встречаемым набором
- Формально:
  - Поддержка любого набора элементов не может превышать минимальной поддержки всех его подмножеств
  - Необходимое условие частой встречаемости k-элементного набора – частая встречаемость всех его (k-1)-элементных подмножеств
  - В примере: supp({Хлеб, Кефир, Чипсы}) не больше чем supp({Хлеб, Кефир}), supp({Хлеб, Чипсы}),supp({Кефир, Чипсы}), supp({Кефир}), supp({Чипсы}),supp({Хлеб})
- Этапы алгоритма:
  - Генерация множества часто встречаемых наборов (supp >= minsupp): метод «ветвей и границ» - направленный перебор от простых (коротких) наборов к сложным (длинным) с отсечением
  - Генерация правил по найденным наборам (conf >= minconf)

Идея метода ветвей и границ для Apriori



Пример генерации кандидатов с отсечением

- L_3={*abc, abd, acd, ace, bcd*}
- Join: L_3*L_3
  - *abcd* = *abc* + *abd*
  - *acde* = *acd* + *ace*
- Pruning:
  - *acde* удален, т.к. *ade* не в L_3
- C_4={*abcd*}

Алгоритм Apriori (формально)

- Вход: X^l – обучающая выборка; $\delta = MinSupp$; $\varkappa = MinConf$;
- Выход: $R = \{(\varphi, y)\}$ – список ассоциативных правил;
- Множество всех частых исходных признаков:
  - $G_1 := \{f \in \mathcal{F} | v(f) \geq \delta\}$;
- Для всех $j = 2, \ldots, n$
  - Множество всех частых наборов мощности j:

$$G_j := \{\varphi \cup \{f\} | \varphi \in G_{j-1}, f \in G_1 \setminus \varphi, v(\varphi \cup \{f\}) \geq \delta\};$$

  - Если $G_j = \emptyset$, то Выход из цикла по j;
- $R := \emptyset$;
- Для всех $\psi \in G_j, j = 2, \ldots, n$ AssocRules$(R, \psi, \emptyset)$;

Пример

D=

t	Хлеб	Кефир	Пиво	Чипсы
1	1	0	0	0
2	1	1	0	0
3	0	1	1	1
4	0	1	1	1
5	1	1	0	0
6	1	0	1	0
7	1	1	1	1
8	1	0	0	0
9	0	0	1	0
10	0	0	1	0

Построение L1
supp(Хлеб) = 60%
supp(Кефир) = 50%
supp(Пиво) = 60%
supp(Чипсы) = 30%
L1 = {{Х}, {К}, {П}, {Ч}}

Построение L2
{Х, К}, {Х, П}, {Х, Ч}
{К, П}, {К, Ч}, {П, Ч}
supp({Х, К}) = 30%
supp({Х, П}) = 20%
supp({Х, Ч}) = 10%
supp({К, П}) = 30%
supp({К, Ч}) = 30%
supp({П, Ч}) = 30%
L2={{Х,К}, {К,П}, {К,Ч}, {П,Ч}}

Пример

D=

t	Хлеб	Кефир	Пиво	Чипсы
1	1	0	0	0
2	1	1	0	0
3	0	1	1	1
4	0	1	1	1
5	1	1	0	0
6	1	0	1	0
7	1	1	1	1
8	1	0	0	0
9	0	0	1	0
10	0	0	1	0

L2={{Х,К}, {К,П}, {К,Ч}, {П,Ч}}

Формируем L3

{К, П, Ч}

supp({К, П, Ч}) = 30%

L3 = {{К, П, Ч}}

Результат=
{{Х}60%,{К}50%,{П}60%,{Ч}30%,
{Х,К}30%,{К,П}30%,{К,Ч}30%,
{П,Ч}30%,{К, П, Ч}30%}

Генерация правил

- Дано и условия:
  - $\text{confidence}(X \Rightarrow Y) = \frac{\text{support}(X,Y)}{\text{support}(X)}$
  - Должно выполняться $\text{confidence}(X \Rightarrow Y) \geq \varkappa$
  - все $\text{support}(X,Y) \geq \delta$ и известны с 1-го этапа – генерации эпизодов
- Принцип:
  - Если правило $\{A\} => \{B,C\}$ интересно, то и $\{A,B\} => \{C\}$ интересно
- Доказательство:

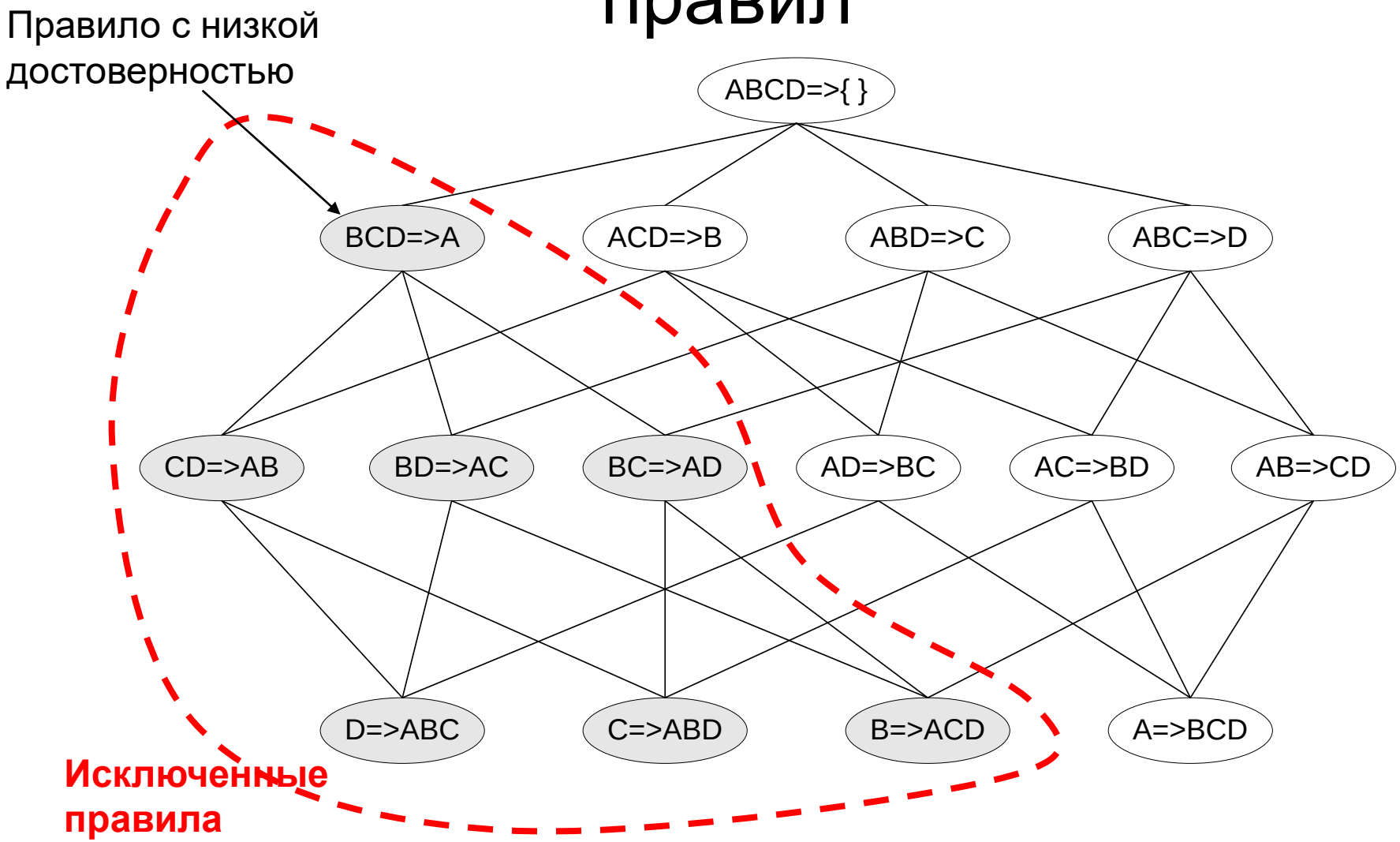
$$\text{confidence}(\{A\} => \{B,C\}) = \frac{\text{support}(\{A,B,C\})}{\text{support}(\{A\})} \geq \varkappa$$

т.к. $\text{support}(\{A,B\}) \leq \text{support}(\{A\})$, то:

$$\text{confidence}(\{A,B\} => \{C\}) = \frac{\text{support}(\{A,B,C\})}{\text{support}(\{A,B\})} \geq \frac{\text{support}(\{A,B,C\})}{\text{support}(\{A\})} \geq \varkappa$$

- Алгоритм:
  - Для каждого часто встречаемого набора проверять правила на интересность, начиная со случая, когда в правой части правила находится один атрибут и постепенно добавлять/убавлять атрибуты в/из правую/левой часть(и).

Метод ветвей и границ для генерации правил



Выделение ассоциативных правил

- Этап 2. Простой рекурсивный алгоритм, выполняемый быстро, как правило, полностью в оперативной памяти.
- Функция AssocRules(R, φ, y)
  - Вход: (φ, y) – ассоциативное правило;
  - Выход: R – список ассоциативных правил;
  - Для всех $f \in \varphi$: $id_f > \max\limits_{g \in y} id_g$ (чтобы избежать повторов y)
    - $\varphi' := \varphi \setminus \{f\}$; $y' := y \cup \{f\}$;
    - если $v(y'|\varphi') \geq \varkappa$ то добавить ассоциативное правило (φ', y') в список R;
    - Если $|\varphi'| > 1$ то AssocRules(R, φ', y');
  - id_f – порядковый номер признака f в $\mathcal{F} = \{f_1, \ldots, f_n\}$

Пример

D=

t	Хлеб	Кефир	Пиво	Чипсы
1	1	0	0	0
2	1	1	0	0
3	0	1	1	1
4	0	1	1	1
5	1	1	0	0
6	1	0	1	0
7	1	1	1	1
8	1	0	0	0
9	0	0	1	0
10	0	0	1	0

Наборы:

- {Х}60%,{К}50%,{П}60%, {Ч}30%, **{Х,К}30%,{К,П}30%,{К,Ч}30%,{П,Ч}30%, {К, П, Ч}30%**

Правила:

conf({Х}=>{К})=50%

conf({К}=>{Х})=60%

conf({К}=>{П})=60%

conf({П}=>{К})=50%

conf({К}=>{Ч})=60%

conf({Ч}=>{К})=100%

conf({П}=>{Ч})=50%

conf({Ч}=>{П})=100%

conf({К, П}=>{Ч})=100%

conf({К}=>{П, Ч})=60%

conf({П}=>{К, Ч})=50%

conf({К, Ч}=>{П})=100%

conf({Ч}=>{К, П})=100%

conf({П, Ч}=>{К})=100%

Недостатки Apriori

- Суть алгоритма Apriori:
  - Использовать часто встречаемые наборы размера (k – 1) для генерации кандидатов часто встречаемых наборов размера k
  - Использовать dbscan и сравнения подмножеств атрибутов для расчета поддержки кандидатов
- Слабое место – генерация кандидатов
  - Огромное число кандидатов: 10^4 1-элементных наборов приводят к 10^7 2-элементным наборам, если надо найти наборы размера 100 $\{a_1, a_2, \ldots, a_{100}\}$, нужно сгенерировать $2^{100} \approx 10^{30}$ кандидатов.
  - Множественные dbscan: (n +1) сканирований, где n - длина наибольшего набора
- Пути решения:
  - Хэш-деревья для хранения наборов и счетчиков поддержки
  - Удаление неинформативных транзакций из базы
  - Разбиение базы и sampling - набор будет часто встречаемым, если он часто встречаемый на каком-то подмножестве транзакций, но: необходима оценка полноты и достоверности

Поиск частых наборов без кандидатов

- Основная задача, решаемая методом **Frequent-Pattern tree** :
  - «сжать» информацию о транзакциях и представить в «компактном» виде с быстрым поиском частых наборов
  - уйти от частых сканирований БД, не генерировать кандидатов, а искать их динамически по структуре FP-tree
  - стратегия «разделяй и властвуй»: декомпозиция задачи поиска на более мелкие подзадачи – рекурсивное построение «пути» частых наборов в FP-tree дереве
- Свойства и требования к структуре:
  - «сжатая» информация для поиска наборов должна быть полной
  - размер вспомогательных структур не должен превосходить размер БД
  - не должно быть несодержательной информации, например
  - при поиске обратная упорядоченность по частоте наборов и атрибутов – более часто встречаемые атрибуты с большой вероятностью являются частью частых наборов

Построение FP-tree

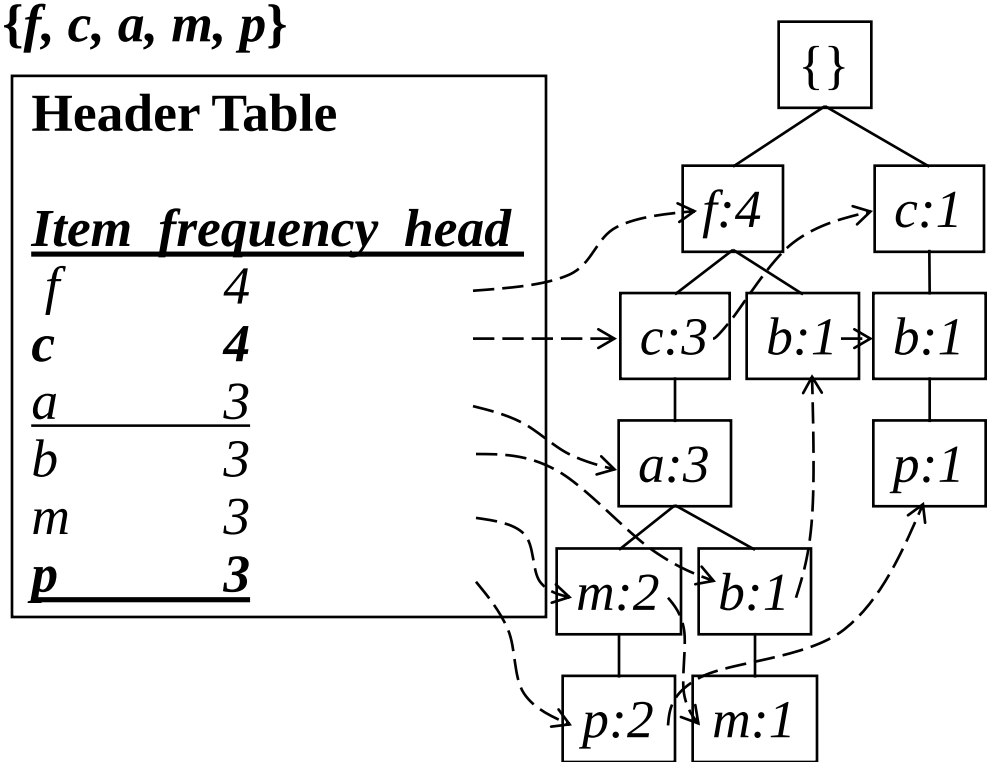
TID	*Items*	*frequent items*
100	{*f, a, c, d, g, i, m, p*}	{*f, c, a, m, p*}
200	{*a, b, c, f, l, m, o*}	{*f, c, a, b, m*}
300	{*b, f, h, j, o*}	{*f, b*}
400	{*b, c, k, s, p*}	{*c, b, p*}
500	{*a, f, c, e, l, p, m, n*}	{*f, c, a, m, p*}

min_support = 0.5

Шаги:

1. Первое сканирование БД и построение частых 1-наборов
2. Обратная сортировка по частоте
3. Второе сканирование и построение FP-tree

По заголовочной таблице можно проверить является эпизод частым: {p,a}? {p,c}?



Префиксное FP-дерево

- В каждой вершине v дерева T задается тройка $\langle f_v, c_v, S_v \rangle$:
  - Признак $f_v \in \mathcal{F}$;
  - Множество дочерних вершин $S_v \subset T$;
  - Счетчик поддержки $c_v = lv(\varphi_v)$ набора $\varphi_v = \{f_u : u \in [v_0, v]\}$,

  где $[v_0, v]$ – путь от корня дерева v_0 до вершины v.
- Обозначения:
  - $V(T, f) = \{v \in T : f_v = f\}$ – все вершины признака (уровня) f.
  - $C(T, f) = \sum_{v \in V(T,f)} c_v$ – сумма счетчиков поддержки признака f.
- Свойства FP-дерева T, построенного по всей выборке X^l:
  - $\boldsymbol{\frac{1}{l} C(T, f) = v(f)}$ **– поддержка признака $\boldsymbol{f}$.**
  - **$\boldsymbol{T}$ содержит информацию о $\boldsymbol{v(\varphi)}$ всех частых наборов $\boldsymbol{\varphi}$.**

Алгоритм FP-growth

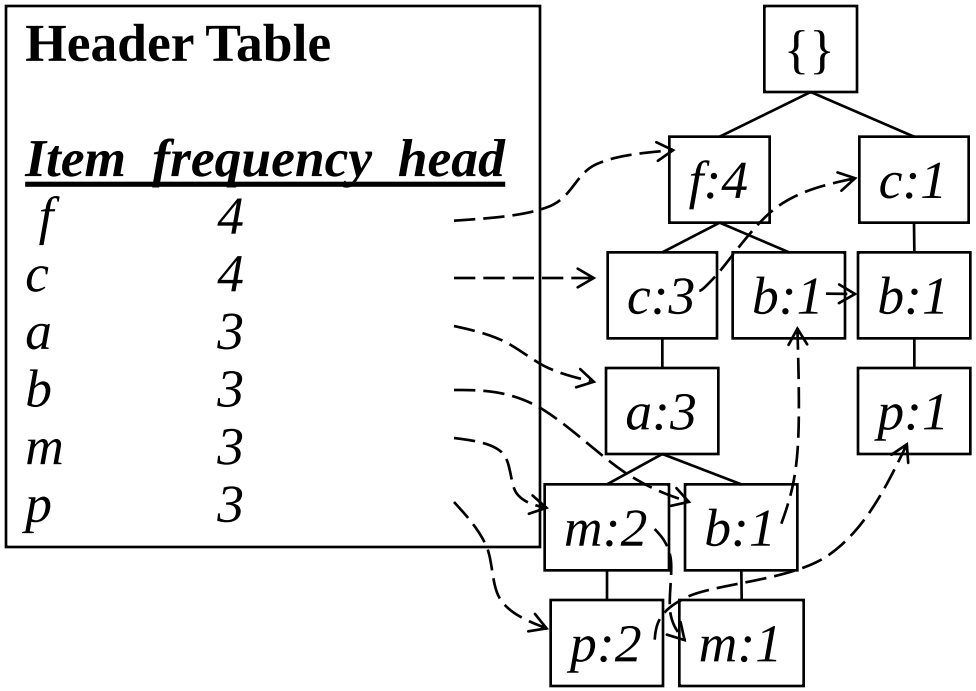
- Вход: X^l – обучающая выборка;
- Выход: FP-дерево T; $\langle f_v, c_v, S_v \rangle$ для всех вершин $v \in T$;
- Упорядочить признаки $f \in \mathcal{F}$: $v(f) \geq \delta$ по убыванию $v(f)$;
- ЭТАП 1: Построение FP-дерева T по выборке X^l
- Для всех $i := 1, \ldots, l$
  - $v := v_0$
  - Для всех $f \in \mathcal{F}$ таких, что $f(x_i) = 1$, по убыванию $v(f)$
    - Если нет дочерней вершины $u \in S_v$: $f_u = f$ то
      - Создать новую вершину u; $S_v := S_v \cup \{u\}$;
      - $f_u := f$; $c_u := 0$; $S_u := \emptyset$;
    - $c_u := c_u + 1$; $v := u$;
- ЭТАП 2: Рекурсивный поиск частых наборов по FP-дереву T
- FP-find$(T, \emptyset, \emptyset)$;

Поиск частых наборов с FP-tree

- Метод:
  - Для каждого элемента (начиная с самых редких) найти его **условный базовый набор**
  - На основе условного базового набора построить новое **условное FP-tree поддерево** для каждого элемента, рассматривая каждый путь как отдельную транзакцию
  - **Повторить процесс** для элементов каждого вновь созданного условного FP-tree поддерева
  - До тех пор пока результирующее FP-tree не будет **пусто или** не будет содержать **единственный путь**
  - **Единственный путь** генерирует все комбинации подпутей, каждый из которых есть **частый набор**

Шаг 1: От FP-tree к условному базовому набору

- Для каждого элемента проход FP-tree «вверх» по дугам с запоминанием «условного» пути и его поддержки
- В результате с каждым элементом связан условный базовый набор (набор возможных путей к вершине с поддержкой)



***Conditional* pattern bases**

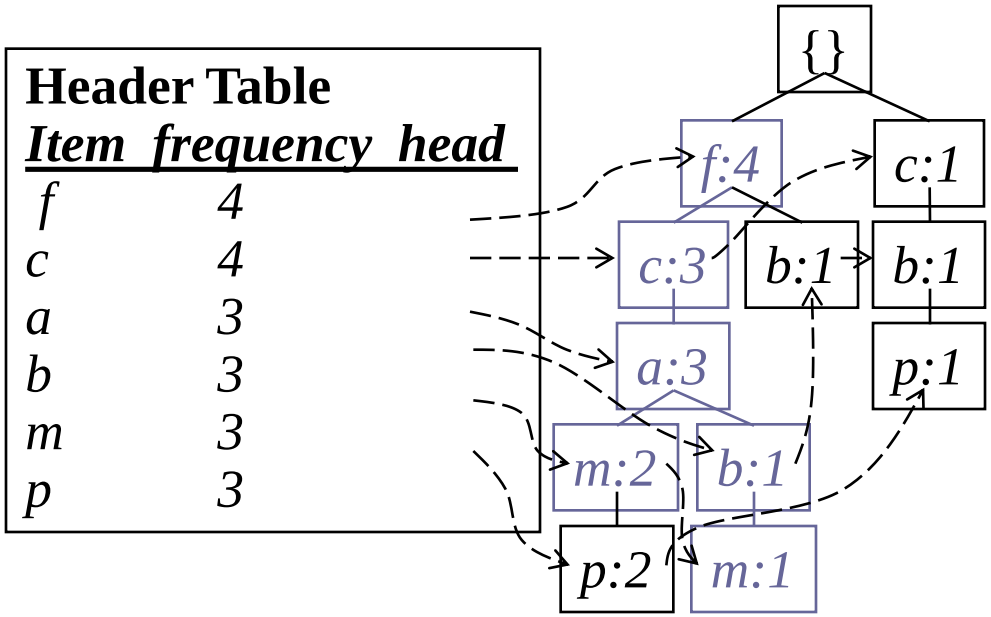
item	***cond. pattern base***
c	***f:3***
a	***fc:3***
b	***fca:1, f:1, c:1***
m	***fca:2, fcab:1***
p	***fcam:2, cb:1***

Свойства условного FP-tree (без доказательства)

- Свойство «узел-связь»:
  - Для каждого частого элемента *a*, все возможные частые наборы, содержащие его, могут быть получены обходом по пути «узел-связь» от заголовочного элемента к корню FP-tree
- Свойство префикса:
  - Для поиска частых наборов для узла *a* в пути *P*, необходимо рассматривать только префикс под-пути от *a* в *P*, его поддержка должна быть равна поддержке узла *a*.

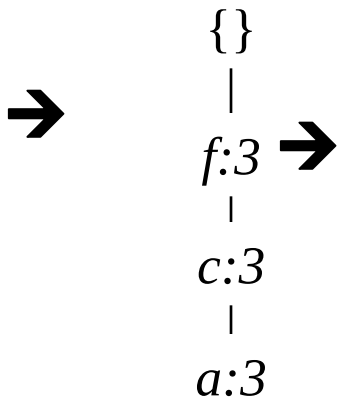
Шаг 2: Построение условного FP-tree

- Для каждого условного базового набора построить условное FP-tree, содержащее только пути из базового набора



***m-conditional* pattern base:**

fca:2, fcab:1



***m-conditional* FP-tree**

All frequent patterns concerning *m*

m,

fm, cm, am,

fcm, fam, cam,

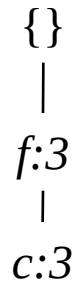
fcam

Поиск частых наборов по условным базовым наборам

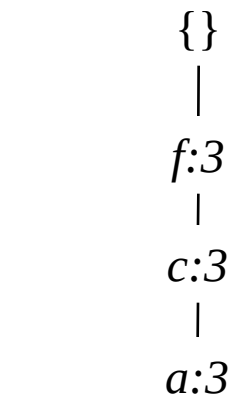
Item	Условный базовый набор	Условное FP-tree
p	{(fcam:2), (cb:1)}	{(c:3)}|p
m	{(fca:2), (fcab:1)}	{(f:3, c:3, a:3)}|m
b	{(fca:1), (f:1), (c:1)}	Empty
a	{(fc:3)}	{(f:3, c:3)}|a
c	{(f:3)}	{(f:3)}|c
f	Empty	Empty

Шаг 3: Рекурсивная обработка условного FP-tree

Условный базовый набор для “am”: (fc:3)

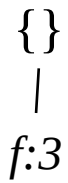


***am-conditional* FP-tree**



***m-conditional* FP-tree**

Условный базовый набор для“cm”: (f:3)



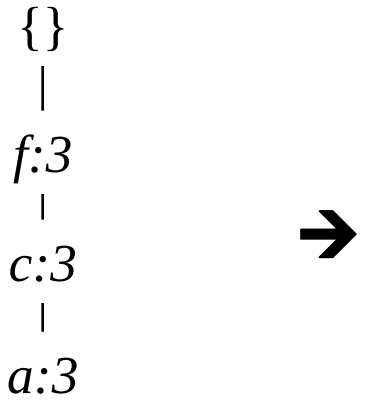
***cm-conditional* FP-tree**

Условный базовый набор для“cam”: (f:3)

***cam-conditional* FP-tree**

Единственный путь в FP-tree

- Предположим в FP-tree есть единственный путь P
- Полное множество частых наборов из T могут быть получены перебором всех возможных комбинаций подпутей из P



→

All frequent patterns concerning *m*

m,

fm, cm, am,

fcm, fam, cam,

fcam

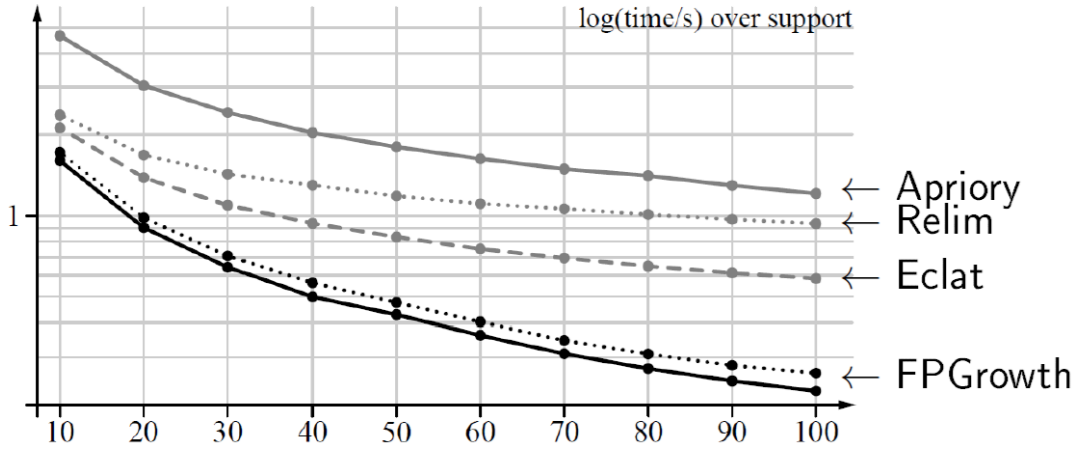
***m-conditional* FP-tree**

Этап 2: Рекурсивный поиск частых наборов по FP-дереву

- FP-find(T, φ, R) находит по FP-дереву T все частые наборы, содержащие частый набор φ, и добавляет их в список R.
- Две идеи эффективной реализации FP-find:
  - Вместо T достаточно передать условное FP-дерево $T|\varphi$, это FP-дерево, порождаемое подвыборкой $\{x_i \in X^l : \varphi(x_i) = 1\}$
  - Будем добавлять в φ только те признаки, которые находятся выше в FP-дереве. Так мы переберем все подмножества $\varphi \subseteq \mathcal{F}$
- Функция FP-find(T, φ, R)
  - Вход: FP-дерево T, частый набор φ, список наборов R;
  - Выход: Добавить в R все частые наборы, содержащие φ;
  - Для всех $f \in \mathcal{F}$: $V(T, f) \neq \emptyset$ по уровням снизу вверх, если $C(T, f) \geq l\delta$:
    - Добавить частый набор $\varphi \cup \{f\}$ в список R;
    - $T' := T|f$ – условное FP-дерево, найти по T' все частые наборы, включающие φ и f: FP-find($T', \varphi \cup \{f\}, R$);

Преимущества FP-tree перед Apriori

- Экспериментально:
  - FP-tree значительно быстрее Apriori



- Причина
  - Нет генерации и проверки кандидатов, нет повторяющихся сканирований БД, используется компактная структура для поиска частых наборов и расчета поддержки, основные операции – суммирование и построение дерева

Использование ограничений

- Проблема итеративного анализа больших объемов данных:
  - невозможно без использования ограничений
- Типы ограничений:
  - стандартные: на support и confidence
  - на меры объективной интересности
  - на выборку «горизонтально» - подмножества транзакций
  - на выборку «вертикально» - подмножества атрибутов
  - на значения отдельных атрибутов (с точки зрения алгоритма аналогично «вертикальному»)
  - шаблоны правил для поиска – метаправила (задаются экспертом, учитываются методом «ветвей и границ» при генерации наборов и правил из них, сокращается перебор)
  - шаблоны «неинтересных» правил (поиск «неожиданных» правил, нарушающих шаблон) – для оценки субъективной интересности