

Методы поиска ассоциативных правил

К. В. Воронцов
vokov@forecsys.ru

Этот курс доступен на странице вики-ресурса
<http://www.MachineLearning.ru/wiki>
«Машинное обучение (курс лекций, К.В.Воронцов)»

ШАД Яндекс • 20 октября 2015

Содержание

- 1 Задачи поиска ассоциативных правил**
 - Определения и обозначения
 - Прикладные задачи
 - Связь с логическими закономерностями
- 2 Алгоритм APriori**
 - Этап 1: поиск частых наборов
 - Этап 2: выделение ассоциативных правил
 - Развитие алгоритмов индукции ассоциативных правил
- 3 Алгоритм FP-Growth**
 - Этап 1: построение префиксного FP-дерева
 - Этап 2: поиск частых наборов по FP-дереву

Определения и обозначения

X — пространство объектов;

$\mathcal{F} = \{f_1, \dots, f_n\}$, $f_j: X \rightarrow \{0, 1\}$ — бинарные признаки (items);

$X^\ell = \{x_1, \dots, x_\ell\} \subset X$ — обучающая выборка.

Каждому подмножеству $\varphi \subseteq \mathcal{F}$ соответствует конъюнкция

$$\varphi(x) = \bigwedge_{f \in \varphi} f(x), \quad x \in X.$$

Если $\varphi(x) = 1$, то «признаки из φ совместно встречаются у x ».

Частота встречаемости (поддержка, support) φ в выборке X^ℓ

$$\nu(\varphi) = \frac{1}{\ell} \sum_{i=1}^{\ell} \varphi(x_i).$$

Если $\nu(\varphi) \geq \delta$, то «набор φ частый» (frequent itemset).

Параметр δ — минимальная поддержка, MinSupp.

Определения и обозначения

Определение

Ассоциативное правило (*association rule*) $\varphi \rightarrow y$ — это пара непересекающихся наборов $\varphi, y \subseteq \mathcal{F}$ таких, что:

1) наборы φ и y совместно часто встречаются,

$$\nu(\varphi \cup y) \geq \delta;$$

2) если встречается φ , то часто встречается также и y ,

$$\nu(y|\varphi) \equiv \frac{\nu(\varphi \cup y)}{\nu(\varphi)} \geq \kappa.$$

$\nu(y|\varphi)$ — значимость (confidence) правила.

Параметр δ — минимальная поддержка, MinSupp.

Параметр κ — минимальная значимость, MinConf.

Классический пример

Анализ рыночных корзин (market basket analysis) [1993]

признаки — товары (предметы, items)

объекты — чеки (транзакции)

$f_j(x_i) = 1$ — в i -м чеке зафиксирована покупка j -го товара.

Пример: «если куплен хлеб φ , то будет куплено и молоко y с вероятностью $\nu(y|\varphi) = 60\%$; причём оба товара покупаются совместно с вероятностью $\nu(\varphi \cup y) = 2\%$ ».

Цели анализа:

- оптимизировать размещение товаров на полках,
- формировать персональные рекомендации,
- планировать рекламные кампании (промо-акции),
- более эффективно управлять ценами и ассортиментом.

Пример 2

Выявление тематики в коллекциях текстовых документов

признаки — термины (отдельные слова или выражения)

объекты — текстовые документы

$f_j(x_i) = 1$ — в i -м тексте (часто) употребляется j -й термин.

Тема — это совокупность терминов, совместно встречающихся в узком подмножестве документов, то есть *частый набор*.

Недостаток: слишком жёсткое требование, чтобы в тексте встречались **все** слова темы. Вероятностные модели адекватнее.

Цели анализа:

- получение признаков для выделения терминов;
- выделение наиболее чётких тем;
- формирование начальных приближений для Topic Models.

Ассоциативные правила — это логические закономерности

Определение

Предикат $\varphi(x)$ — логическая ε, δ -закономерность класса $c \in Y$

$$D_c(\varphi, X^\ell) = \frac{p_c(\varphi)}{\ell} \geq \delta; \quad E_c(\varphi, X^\ell) = \frac{n_c(\varphi)}{p_c(\varphi) + n_c(\varphi)} \leq \varepsilon,$$

$$p_c(\varphi) = \#\{x_i: \varphi(x_i) = 1 \text{ и } y(x_i) = c\} \quad + \text{примеры класса } c;$$

$$n_c(\varphi) = \#\{x_i: \varphi(x_i) = 1 \text{ и } y(x_i) \neq c\} \quad - \text{примеры класса } c.$$

Для « $\varphi \rightarrow y$ » возьмём целевой признак $y(x) = \bigwedge_{f \in y} f(x)$. Тогда

$$\nu(\varphi \cup y) \equiv D_1(\varphi) \geq \delta; \quad \frac{\nu(\varphi \cup y)}{\nu(\varphi)} \equiv 1 - E_1(\varphi) \geq 1 - \varepsilon \equiv \kappa.$$

Вывод: различия двух определений — чисто терминологические.

Два этапа построения правил. Свойство антимонотонности

Поскольку $\varphi(x) = \bigwedge_{f \in \varphi} f(x)$ — конъюнкция, имеет место

Свойство антимонотонности:

для любых $\psi, \varphi \subset \mathcal{F}$ из $\varphi \subset \psi$ следует $\nu(\varphi) \geq \nu(\psi)$.

Следствия:

- 1 если ψ частый, то все его подмножества $\varphi \subset \psi$ частые.
- 2 если φ не частый, то все наборы $\psi \supset \varphi$ также не частые.
- 3 $\nu(\varphi \cup \psi) \leq \nu(\varphi)$ для любых φ, ψ .

Два этапа поиска ассоциативных правил:

- 1 поиск частых наборов
(многократный просмотр транзакционной базы данных).
- 2 выделение ассоциативных правил
(простая эффективная процедура в оперативной памяти).

Алгоритм APriory (основная идея — поиск в ширину)

Вход: X^ℓ — обучающая выборка;

минимальная поддержка δ ; минимальная значимость κ ;

Выход: $R = \{(\varphi, y)\}$ — список ассоциативных правил;

1: множество всех частых исходных признаков:

$$G_1 := \{f \in \mathcal{F} \mid \nu(f) \geq \delta\};$$

2: **для всех** $j = 2, \dots, n$

3: множество всех частых наборов мощности j :

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

4: **если** $G_j = \emptyset$ **то**

5: **выход** из цикла по j ;

6: $R := \emptyset$;

7: **для всех** $\psi \in G_j, j = 2, \dots, n$

8: AssocRules (R, ψ, \emptyset);

Выделение ассоциативных правил

Этап 2. Простой алгоритм, выполняемый быстро, как правило, полностью в оперативной памяти.

Вход и Выход: R — список ассоциативных правил;
 (φ, y) — ассоциативное правило;

- 1: **ПРОЦЕДУРА** AssocRules (R, φ, y);
 - 2: **для всех** $f \in \varphi$: $\text{id}_f > \max_{g \in y} \text{id}_g$ (чтобы избежать повторов y)
 - 3: $\varphi' := \varphi \setminus \{f\}$; $y' := y \cup \{f\}$;
 - 4: **если** $\nu(y' | \varphi') \geq \kappa$ **то**
 - 5: добавить ассоциативное правило (φ', y') в список R ;
 - 6: **если** $|\varphi'| > 1$ **то**
 - 7: AssocRules (R, φ', y');
-

id_f — порядковый номер признака f в $\mathcal{F} = \{f_1, \dots, f_n\}$

Модификации алгоритмов индукции ассоциативных правил

- Более эффективные структуры данных для быстрого поиска частых наборов.
- Сэмплинг с последующей проверкой правил на полной выборке.
- Иерархические алгоритмы, учитывающие иерархию признаков (например, товарное дерево).
- Учёт времени: инкрементные и декрементные алгоритмы.
- Учёт времени: поиск последовательных шаблонов (sequential pattern).
- Учёт информации о клиентах.

Префиксное FP-дерево (FP — frequent pattern)

В каждой вершине v дерева T задаются:

- признак $f_v \in \mathcal{F}$;
- множество дочерних вершин $S_v \subset T$;
- поддержка $c_v = \nu(\varphi_v)$ набора признаков $\varphi_v = \{f_u : u \in [v_0, v]\}$, 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 f .

$C(T, f) = \sum_{v \in V(T, f)} c_v$ — суммарная поддержка признака f .

Свойства FP-дерева T , построенного по всей выборке X^ℓ :

- 1 T содержит полную информацию о всех $\nu(\varphi)$, $\varphi \subseteq \mathcal{F}$.
- 2 $C(T, f) = \nu(f)$ для всех $f \in \mathcal{F}$.

Пример: построение префиксного FP-дерева

Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

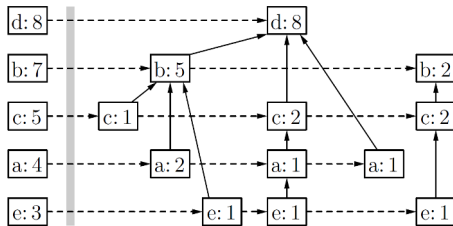
Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a

(корень v_0 не показан)



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

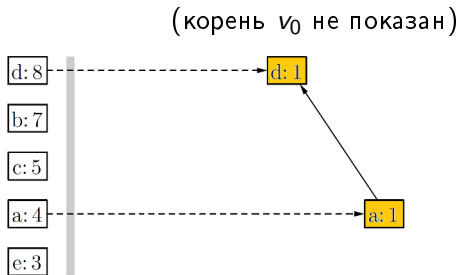
Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

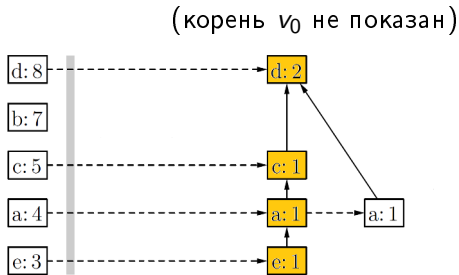
Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

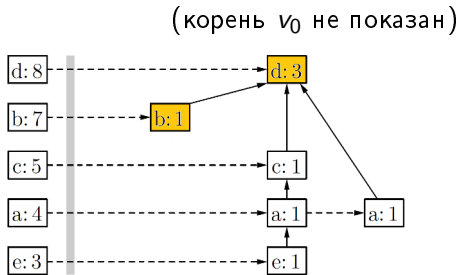
Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

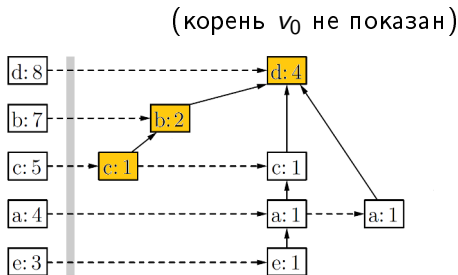
Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

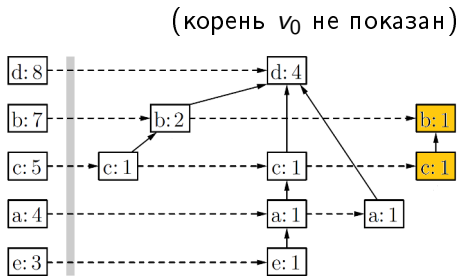
Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

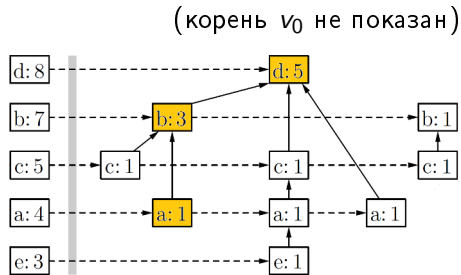
Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

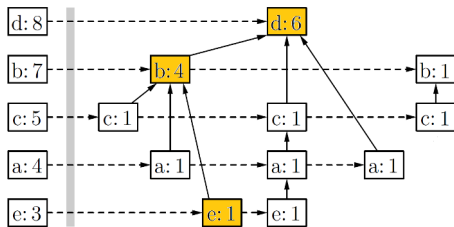
Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a

(корень v_0 не показан)



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

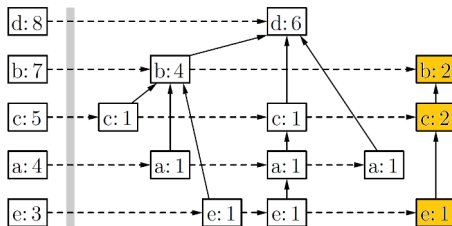
Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a

(корень v_0 не показан)



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

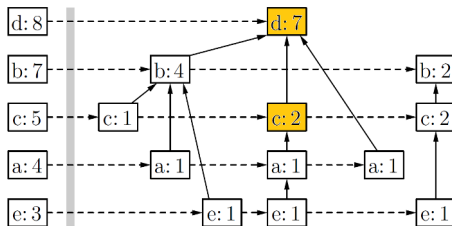
Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a

(корень v_0 не показан)



при $\delta = 3$ признаки f, g не частые

Пример: построение префиксного FP-дерева

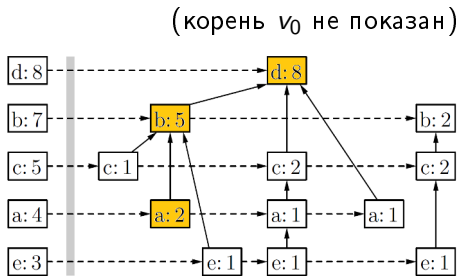
Упорядочим все признаки $f \in \mathcal{F}: \nu(f) \geq \delta$ по убыванию $\nu(f)$.

Тогда каждый объект описывается словом в алфавите \mathcal{F} ;

FP-дерево — это эффективный способ хранения словаря;

уровни дерева соответствуют признакам, по убыванию $\nu(f)$.

матрица	слова
a - - d - f -	d a
a - c d e - -	d c a e
- b - d - - -	d b
- b c d - - -	d b c
- b c - - - -	b c
a b - d - - -	d b a
- b - d e - -	d b e
- b c - e - g	b c e
- - c d - f -	d c
a b - d - - -	d b a



при $\delta = 3$ признаки f, g не частые

Алгоритм FP-growth

Вход: X^ℓ — обучающая выборка;

Выход: FP-дерево T , $\langle f_v, c_v, S_v \rangle_{v \in T}$;

-
- 1: упорядочить признаки $f \in \mathcal{F}$: $\nu(f) \geq \delta$ по убыванию $\nu(f)$;
ЭТАП 1: построение FP-дерева T по выборке X^ℓ
 - 2: **для всех** $x_i \in X^\ell$
 - 3: $v := v_0$;
 - 4: **для всех** $f \in \mathcal{F}$ таких, что $f(x_i) \neq 0$
 - 5: **если** нет дочерней вершины $u \in S_v$: $f_u = f$ **то**
 - 6: **создать** новую вершину u ; $S_v := S_v \cup \{u\}$;
 $f_u := f$; $c_u := 0$; $S_u := \emptyset$;
 - 7: $c_u := c_u + 1/\ell$; $v := u$;
 - 8: *ЭТАП 2: рекурсивный поиск частых наборов по FP-дереву T*
FP-find (T, \emptyset, \emptyset);

Этап 2: рекурсивный поиск частых наборов по FP-дереву

Вход: FP-дерево T , набор $\varphi \in \mathcal{F}$, список частых наборов R ;

Выход: добавить в R все частые наборы, содержащие φ ;

- 1: **ПРОЦЕДУРА** FP-find (T, φ, R);
- 2: **для всех** $f \in \mathcal{F} : V(T, f) \neq \emptyset$ по уровням **снизу вверх**
- 3: **если** $C(T, f) \geq \delta$ **то**
- 4: добавить частый набор $\varphi \cup \{f\}$ в список R :
 $R := R \cup \{\varphi \wedge f\}$;
- 5: построить **условное FP-дерево** $T' := T|f$, а именно:
 $T' :=$ FP-дерево по подвыборке $\{x_i \in X^l : f(x_i) = 1\}$;
- 6: найти по T' все частые наборы, включающие φ и f :
 FP-find ($T', \varphi \cup \{f\}, R$);

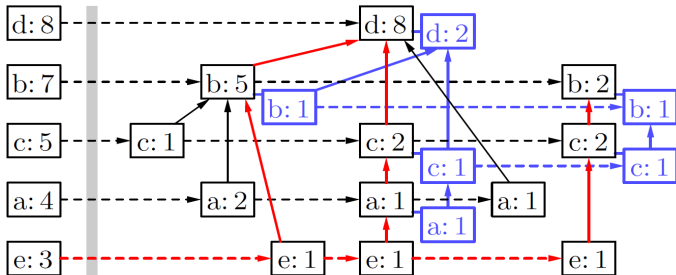
Условное FP-дерево $T' := T|f$ можно построить быстро, используя только FP-дерево T и не заглядывая в выборку.

Условное FP-дерево

Пусть FP-дерево T построено по выборке X^ℓ .

Опр. Условное FP-дерево (conditional FP-tree) — это FP-дерево $T' := T|f$, построенное по подвыборке $\{x_i \in X^\ell : f(x_i) = 1\}$, из которого удалены все вершины $v \in V(T', f)$ и все их потомки.

Продолжение примера: CFP-дерево $T|“e”$



Быстрое построение условного FP-дерева $T' = T|f$

Вход: FP-дерево T , признак $f \in \mathcal{F}$;

Выход: условное FP-дерево $T' = T|f$;

-
- 1: оставить в дереве только вершины на путях из вершин v признака f снизу вверх до корня v_0 :

$$T' := \bigcup_{v \in V(T, f)} [v, v_0];$$

- 2: поднять значения счётчиков c_v от вершин $v \in V(T', f)$ снизу вверх по правилу

$$c_u := \sum_{w \in S_u} c_w \text{ для всех } u \in T';$$

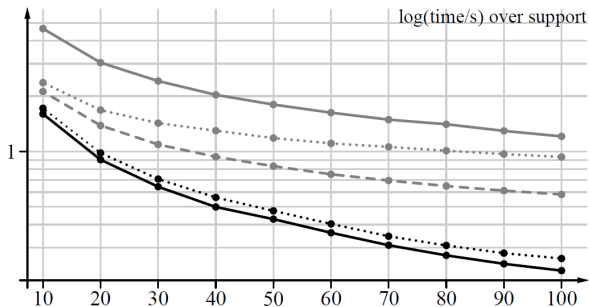
- 3: удалить из T' все вершины признака f ;
их поддеревья также не нужны и даже не создаются, т.к.

в момент вызова FP-find все наборы, содержащие признаки ниже f , уже просмотрены.

Эффективность алгоритма FP-Growth

Одна из типичных зависимостей \log времени работы алгоритма от MinSupp (на выборке данных census).

Нижние кривые — две разные реализации FP-growth.



Christian Borgelt. An Implementation of the FP-growth Algorithm. 2005.

Резюме в конце лекции

- Поиск ассоциативных правил — обучение без учителя.
- Ассоциативное правило (по определению) — почти то же самое, что логическая ϵ, δ -закономерность.
- Простые алгоритмы типа APriori вычислительно неэффективны на больших данных.
- FP-growth — один из самых эффективных алгоритмов поиска ассоциативных правил.
- Для практических приложений часто используются его инкрементные и/или иерархические обобщения.