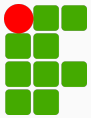
Mineração de Dados

07 - Mineração de Conjuntos de Itens

Marcos Roberto Ribeiro



Instituto Federal Minas Gerais - Campus Bambuí

2018

Introdução

- A mineração de um conjunto de itens frequentes (ou *itemset*) é um dos tópicos de pesquisa mais ativos em descoberta de conhecimento em bases de dados
- Um trabalho pioneiro nessa área foi a análise de carrinhos de compras de clientes
- O objetivo é descobrir grupos de produtos que frequentemente são comprados em conjunto
- Desde então, um grande número de algoritmos eficientes tem sido desenvolvido

Mineração de Conjuntos de Itens Frequentes

- Seja $A = \{a_I, ..., a_m\}$ o universo de m itens
- Qualquer subconjunto $I \subseteq A$ é chamado um conjunto de itens (ou *itemset*)
- Um conjunto de itens diz respeito a qualquer conjunto de produtos que podem ser comprados juntos
- Seja $T = (t_1, ..., t_n)$ um conjunto de n transações
- Cada transação é um par $\langle id, k - items \rangle$, em que id é o identificador da transação e $k - items$ é um conjunto de k itens
- Exemplos de conjunto de transações são os carrinhos de compras dos cliente de um supermercado, o conjunto de páginas visitadas por por usuários de um site, etc
- Uma transação $t \in T$ suporta o conjunto de itens I ou I está contido no $k - item$ da transação t, se e somente se, a transação t contém todos os elementos de I

Itemsets Frequentes e Regras de Associação

- O objetivo da tarefa de mineração é encontrar *itemsets* para, posteriormente, derivar regras de associações a partir destes *itemsets*
- As regras de associação têm o seguinte formato:

$$\text{item}_1, ..., \text{item}_j \rightarrow \text{item}_l$$

Exemplo:

- Itemset: {pão, leite, manteiga, vinho, queijo, ... }
- Regras de associação:
  - pão, leite $\rightarrow$ manteiga
  - vinho $\rightarrow$ queijo

Medidas de Interesse

- Suporte de $\{A, B, C, D\} = \frac{\text{Número de transações contendo } \{A,B,C,D\}}{\text{Número total de transações}}$
- Confiança de $A, B, C \rightarrow D = \frac{\text{Número de transações contendo } \{A,B,C,D\}}{\text{Número de transações contendo } \{A,B,C\}}$

Exemplo

ID	Itens
1	$\{a, d, e\}$
2	$\{b, c, d\}$
3	$\{a, c, e\}$
4	$\{a, c, d, e\}$
5	$\{a, e\}$
6	$\{a, c, d\}$
7	$\{b, c\}$
8	$\{a, c, d, e\}$
9	$\{b, c, e\}$
10	$\{a, d, e\}$

- Itemsets frequentes considerando suporte mínimo de 30%
  - 1-itemset: $\{a\} : 7$, $\{b\} : 3$, $\{c\} : 7$, $\{d\} : 6$, $\{e\} : 7$
  - 2-itemset: $\{a, c\} : 4$, $\{a, d\} : 5$, $\{a, e\} : 6$, $\{b, c\} : 3$, $\{c, d\} : 4$, $\{c, e\} : 4$, $\{d, e\} : 4$
  - 3-itemset: $\{a, c, d\} : 3$, $\{a, c, e\} : 3$, $\{a, d, e\} : 4$

Definição do Problema

Dados

- Um conjunto $A = \{a_1, ..., a_m\}$ de itens
- Um conjunto $T = (t_1, ..., t_n)$ de transações contendo os itens de A
- Um suporte mínimo σ_{min} tal que $0 < \sigma_{\text{min}} \leq 1$
- Uma confiança mínima c_{min} tal que $0 < c_{\text{min}} \leq 1$

Objetivos

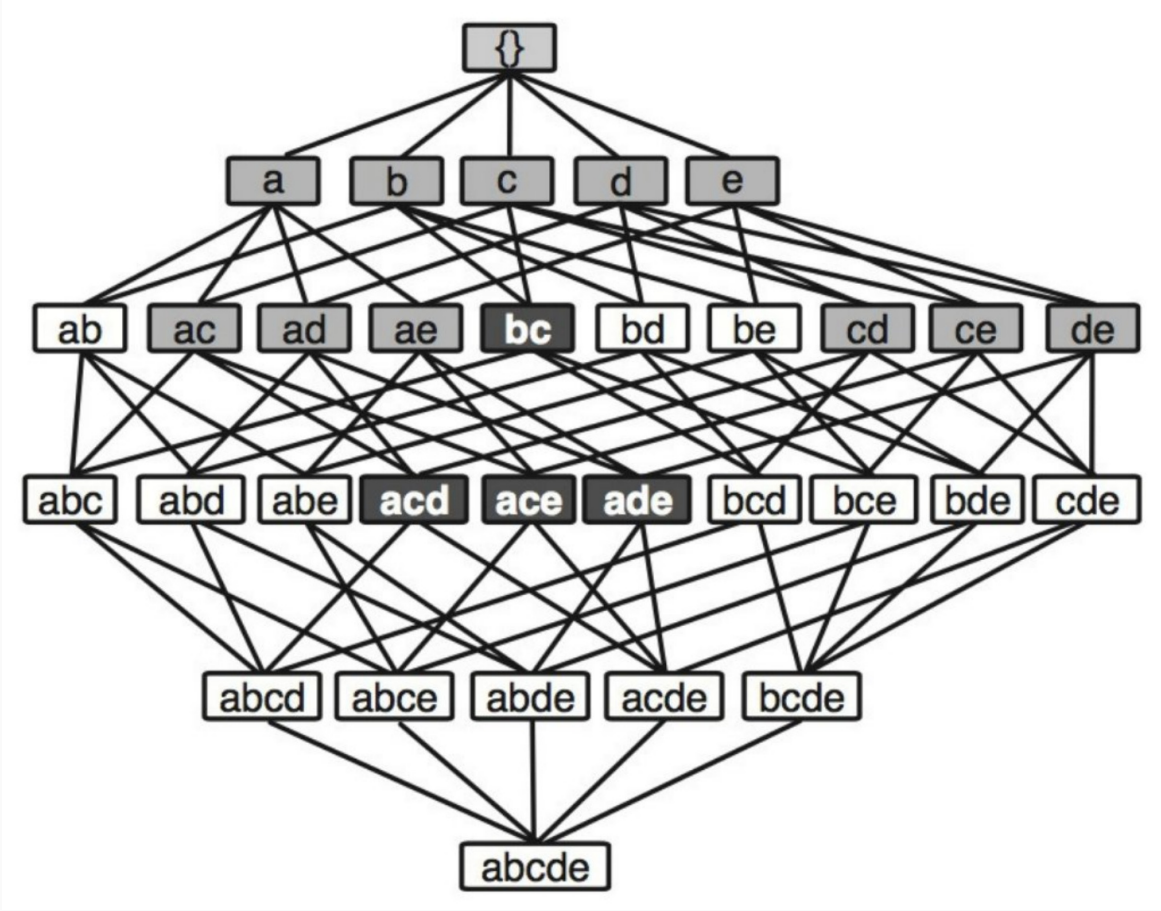
- Encontrar o conjunto de itens frequentes, tais que o suporte relativo de cada conjunto de itens é maior ou igual a σ_{min}
- Encontrar o conjunto de regras de associação com confiança maior que c_{min}

Histórico

- Desde sua introdução em [Agrawal et al., 1993], a mineração de itemsets e regras de associação receberam uma grande atenção
- Na década de 1990, diversas pesquisas desenvolveram novos algoritmos ou melhorias nos algoritmos existentes para resolver esse problema de forma mais eficiente

O Espaço de Busca I

- Dado um conjunto de itens A, temos $2^{|A|}$ *itemsets* possíveis



O Espaço de Busca II

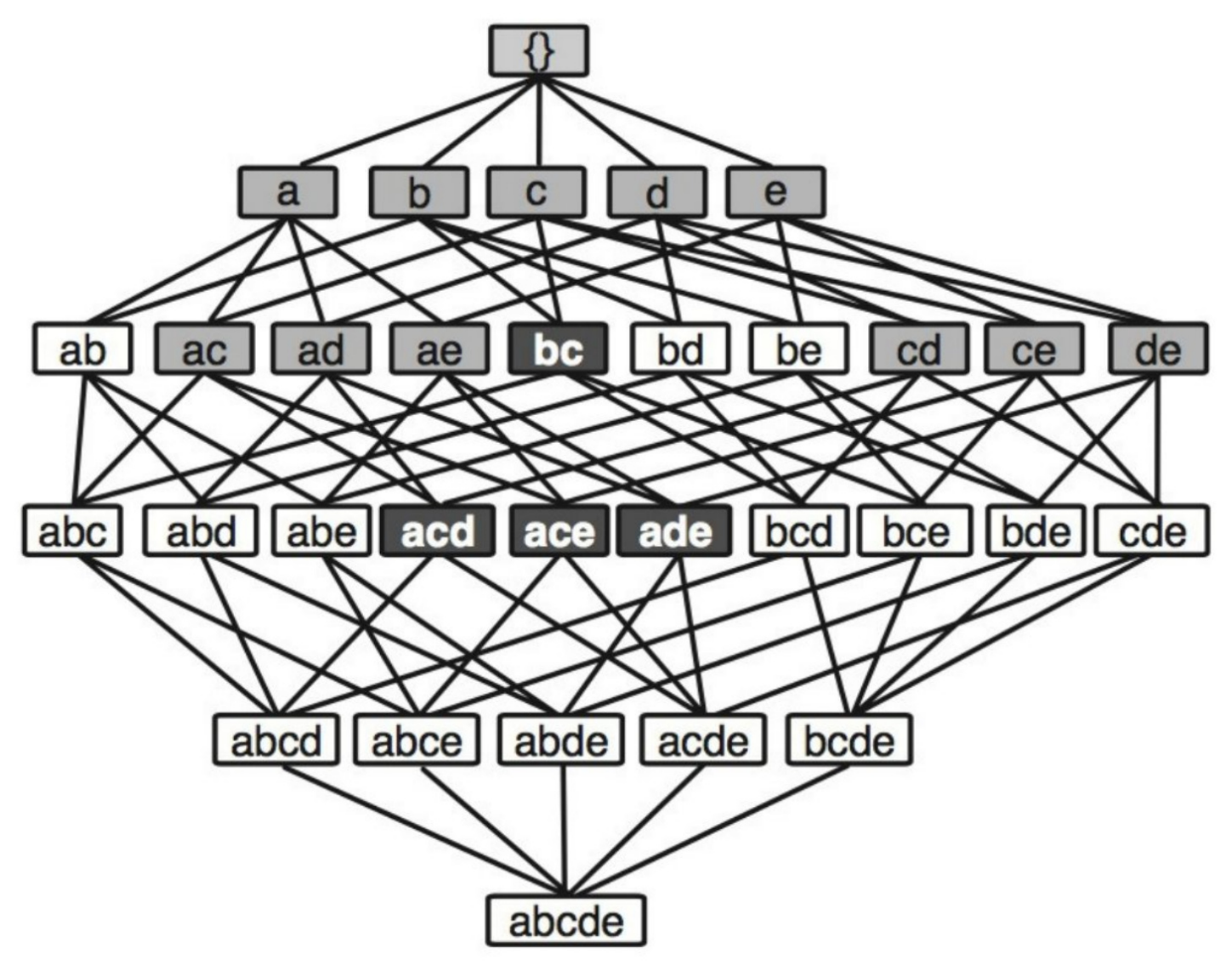
- Se $|A|$ é grande o suficiente, então a geração de todos os *itemsets* e a contagem do suporte dos mesmos pode consumir muito tempo
- A maioria dos algoritmos explora a propriedade monotonicamente decrescente do suporte com relação ao número de itens
- Sejam X e Y dois itensets de um banco de dados de transações T sobre I sendo que $X, Y \subseteq I$
- Se $X \subseteq Y$, então $\mathit{suporte}(Y) \leq \mathit{suporte}(X)$

O Algoritmo Apriori

- O algoritmo Apriori utiliza uma estratégia de busca em largura
- Em cada nível são gerados os possíveis *itemsets*, considerando os *itemsets* frequentes gerados no nível anterior
- Após serem gerados, a frequência desses *itemsets* é testada, percorrendo novamente a base de dados
- O algoritmo Apriori começa com a geração do conjunto F_1 (*itemsets* de tamanho 1)
- Cada item é um possível candidato (é preciso testar o suporte varrendo a base de dados)
- Os *itemsets* de tamanho $k+1$ são obtidos a partir dos *itemsets* de tamanho k em dois passos:
  1. Combinamos os *itemsets* do conjunto F_k (dois *itemsets* podem ser combinados, se eles possui o mesmo $(k-1)$-prefixo
  2. Em seguida é feita a poda, $X \cup Y$ é inserido em F_{k+1} se todos os seus k-subconjuntos ocorrem em F_k
- Por fim, varremos a base de dados e incluímos em F_{k+1} apenas os *itemsets* com suporte válido

Exemplo

ID	Itens
1	$\{a, d, e\}$
2	$\{b, c, d\}$
3	$\{a, c, e\}$
4	$\{a, c, d, e\}$
5	$\{a, e\}$
6	$\{a, c, d\}$
7	$\{b, c\}$
8	$\{a, c, d, e\}$
9	$\{b, c, e\}$
10	$\{a, d, e\}$



AprioriItemsets(T, σ)

1 Varrer T e coletar F_1 (*itemsets* de tamanho 1 frequentes);
2 $k \leftarrow 1$;
3 **while** $F_k \neq \{\}$ **do**
4 $C_{k+1} \leftarrow \{\}$;
5 **foreach** $X, Y \in F_k$ **do**
6 **if** $(X - X[k]) = (Y - Y[k])$ **then** /* Testa se X e Y têm o mesmo prefixo */
7 $Z \leftarrow X \cup Y[k]$; /* Cria *itemset* de tamanho $k+1$ */
8 $C_{k+1} \leftarrow C_{k+1} \cup \{Z\}$;
9 **foreach** $Z' \in Z$ **do** /* Poda */
10 **if** $|Z'| = k$ ***and*** $Z' \notin F_k$ **then** $C_{k+1} \leftarrow C_{k+1} - \{Z\}$;
11 **foreach** $Z \in C_{k+1}$ **do** /* Checagem de suporte */
12 **foreach** $t \in T$ **do**
13 **if** $Z \in t$ **then** $suporte(Z) \leftarrow suporte(Z) + 1$;
14 $F_{k+1} \leftarrow \{Z \in C_{k+1} \mid suporte(Z) \leq \sigma\}$;
15 $k \leftarrow k + 1$

Regras de Associação

AprioriRules(I, c)

1 $S \leftarrow$ subconjuntos não vazios de I;
2 $R \leftarrow \{\}$;
3 **foreach** $s \in S$ **do**
4 **if** $\frac{suporte(I)}{suporte(s)} \leq c$ **then** /* Testa confiança da regra $s \rightarrow I \setminus s$ */
5 $R \leftarrow R \cup \{s \rightarrow I \setminus s\}$;

Observação

- O algoritmo Apriori realiza diversas varreduras sobre o banco de dados para calcular o suporte dos *itemsets* frequentes candidatos
- Como alternativa, diversos algoritmos reduziram essas varreduras por meio da geração de coleções de *itemsets* candidatos em uma estratégia de busca em profundidade

O Algoritmo FP-Growth

- O algoritmo FP-Growth procede em duas fases [Han et al., 2000]:
  1. Constrói uma estrutura de dados a FP-tree percorrendo a base de dados duas vezes
  2. Utiliza a FP-tree para encontrar as regras de associação

Construção da FP-tree

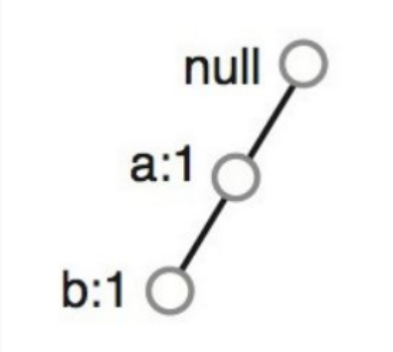
ID	Itens
1	$\{a, b\}$
2	$\{b, c, d\}$
3	$\{a, c, d, e\}$
4	$\{a, d, e\}$
5	$\{a, b, c\}$
6	$\{a, b, c, d\}$
7	$\{a\}$
8	$\{a, b, c\}$
9	$\{a, b, d\}$
10	$\{b, c, e\}$

- Primeira varredura: $L = [a, b, c, d, e]$ (mais frequentes primeiro)
- Segunda varredura:
  - Processando TID = 1
  - Processando TID = 2
  - Processando TID = 3
  - Processando TID = 10

Construção da FP-tree

ID	Itens
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	$\{b, c, e\}$

- Primeira varredura: $L = [a, b, c, d, e]$ (mais frequentes primeiro)
- Segunda varredura:
  - Processando TID = 1

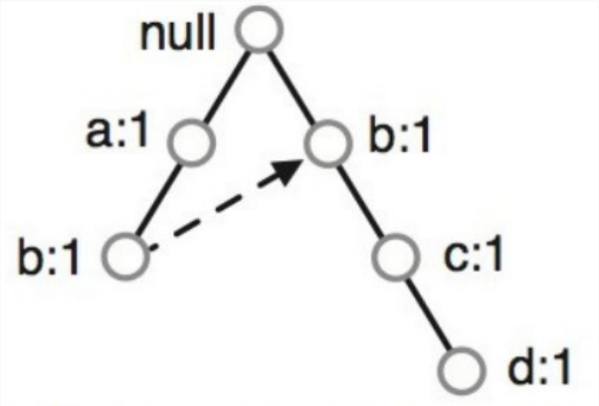


  - Processando TID = 2
  - Processando TID = 3
  - Processando TID = 10

Construção da FP-tree

ID	Itens
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	$\{b, c, e\}$

- Primeira varredura: $L = [a, b, c, d, e]$ (mais frequentes primeiro)
- Segunda varredura:
  - Processando TID = 1
  - Processando TID = 2

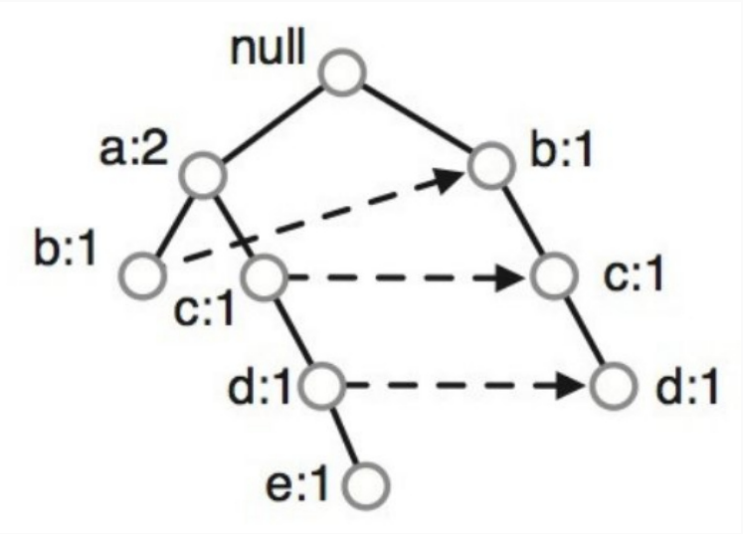


  - Processando TID = 3
  - Processando TID = 10

Construção da FP-tree

ID	Itens
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	$\{b, c, e\}$

- Primeira varredura: $L = [a, b, c, d, e]$ (mais frequentes primeiro)
- Segunda varredura:
  - Processando TID = 1
  - Processando TID = 2
  - Processando TID = 3

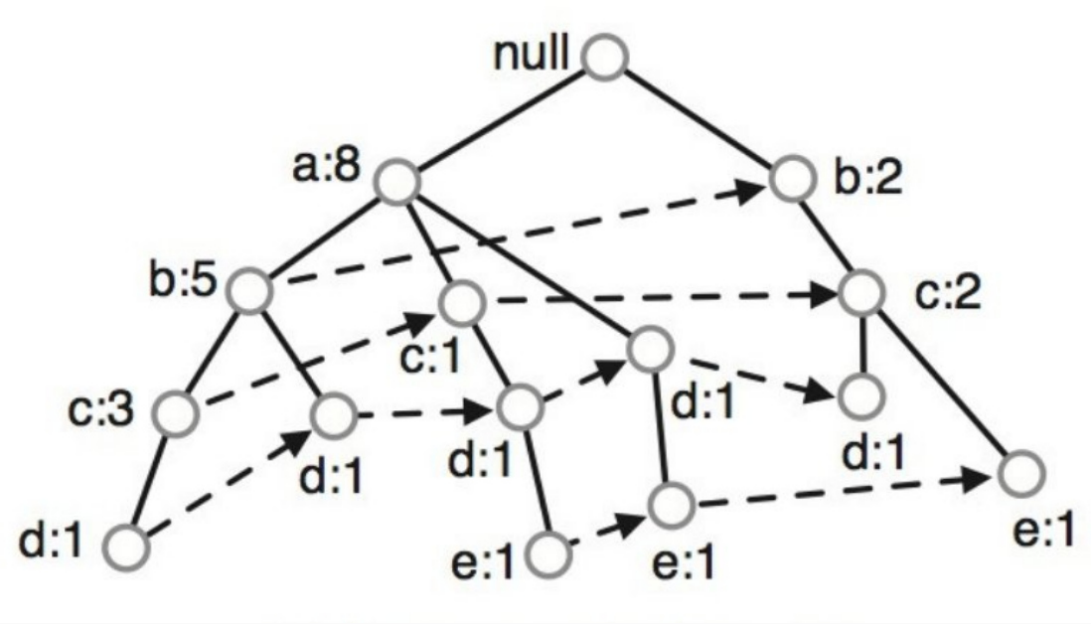


- Processando TID = 10

Construção da FP-tree

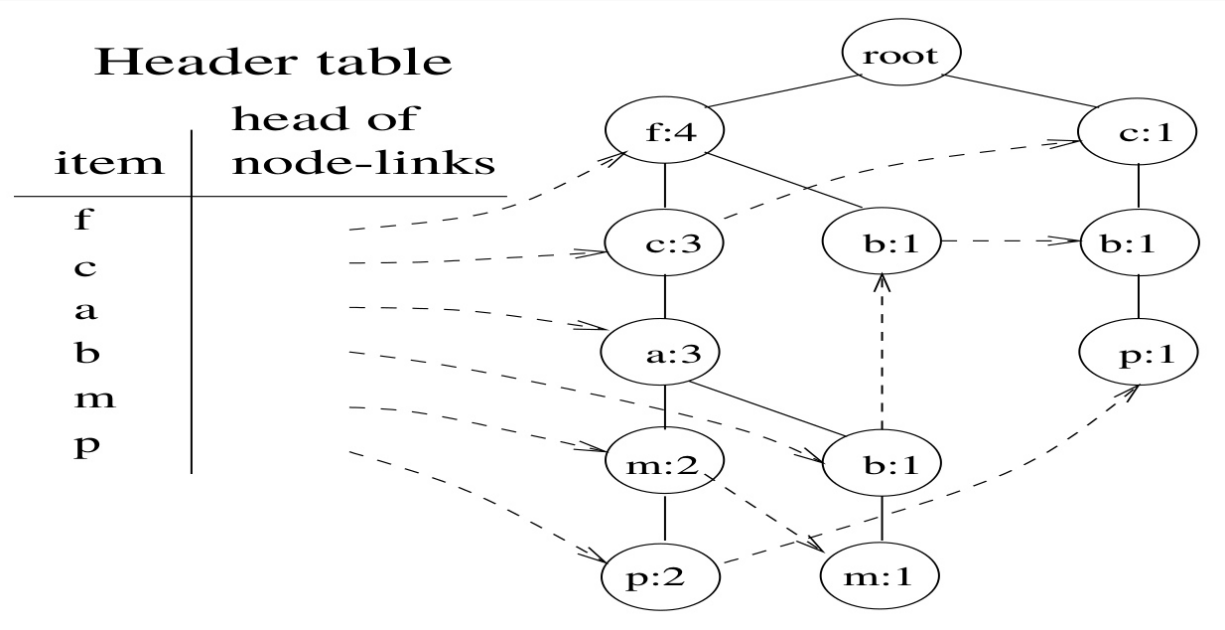
ID	Itens
1	$\{a, b\}$
2	$\{b, c, d\}$
3	$\{a, c, d, e\}$
4	$\{a, d, e\}$
5	$\{a, b, c\}$
6	$\{a, b, c, d\}$
7	$\{a\}$
8	$\{a, b, c\}$
9	$\{a, b, d\}$
10	$\{b, c, e\}$

- Primeira varredura: $L = [a, b, c, d, e]$ (mais frequentes primeiro)
- Segunda varredura:
  - Processando TID = 1
  - Processando TID = 2
  - Processando TID = 3
  - Processando TID = 10



FP-Tree Completa ($\sigma = 0.6$

TID	Itens
100	f, a, c, d, g, i, m,p
200	a, b, c, f, l, m, o
300	b, f, h, j, o
400	b, c, k, s, p
500	a, f, c, e, l, p, m, n



Pseudo-Código do Algoritmo FP-Growth

FP-Growth(T, σ)

```
   /* itemsets de tamanho 1 frequentes                    */
1  Varrer T e coletar F_1 ;
2  Ordenar decrescente F_1 em L ;
3  T ← null ; /* FP-Tree                                  */
4  foreach t ∈ T do
5  |   i ← itens frequentes de t ordenados
   |     por L ;
6  |   InsereArvore(T, i) ;
```

InsereArvore(T, i)

```
   /* i[1] é o primeiro item de i                         */
1  if i[1] é filho de T then
2  |   f ← T.Filho(i[1]) ;
3  |   Incremente contador de f ;
4  else
5  |   Insere filho f = i[1] com contador 1
   |     em T ;
6  |   Ligue último nó i[1] a f ;
7  i' ← i − i[1] ;
8  if i ≠ [] then
9  |   InsereArvore(f, i') ;
```

Exercício

- Desenhar a FP-Tree para a seguinte base de dados:

ID	Itens
1	$\{d, a, e, f\}$
2	$\{b, c, d, g\}$
3	$\{a, e, c\}$
4	$\{c, a, g, d, e\}$
5	$\{b, a, e, g\}$
6	$\{a, f, c, d\}$
7	$\{j, b, h, c\}$
8	$\{l, a, c, d, e\}$
9	$\{p, b, c, e\}$
10	$\{a, q, d, e\}$

Heurísticas para Seleção de Regras de Associação

- Os algoritmos de regras de associação costumam gerar grandes volumes de regras
- Devemos considerar as regras que são mais *interessantes* ou *relevantes*

Exemplo

- Dados de 1000 pessoas:

	Café	Não Café	Total
Chá	150	50	200
Não chá	650	150	800
Total	800	200	1000

- Analisando a regra $\{\text{Chá}\} \rightarrow \{\text{Café}\}$, temos:
  - Suporte $= 150/1000 = 0.15$
  - Confiança $= 0.15/0.2 = 0.75$
- A confiança da regra é elevada, no entanto a probabilidade de uma pessoa beber café, independentemente de beber chá, é de 80%
- Sabendo que se uma pessoa bebe chá, diminui a probabilidade dela beber também café, a $\{\text{Chá}\} \rightarrow \{\text{Café}\}$ é enganadora.

Coeficiente de Interesse ou *Lift*

- O coeficiente de interesse, ou *Lift*, reflete a noção estatística de independência entre duas variáveis aleatórias

$$I(A, B) = \frac{P(A, B)}{P(A) \times P(B)}$$

- Também podemos calcular com base no suporte e confiança da regra

$$lift(A \rightarrow B) = \frac{confiança(A \rightarrow B)}{suporte(B)} = \frac{suporte(A \cup B)}{suporte(A) \times suporte(B)}$$

- O *lift* menor que 1 indica que as variáveis A e B não são correlacionadas
- O *lift* da regra $\{\text{Chá}\} \rightarrow \{\text{Café}\}$ é $0.15/(0.2 \times 0.8) = 0.9375$ (mostrando pouca correlação entre chá e café)

Convicção

- A convicção de uma regra permite medir a frequência de erro da mesma

$$convicção(A \rightarrow B) = \frac{1 - suporte(B)}{1 - confiança(A -> B)}$$

- A convicção da regra {Chá} $\rightarrow$ {Café} é: $(1 - 0.8)/(1 - O.75) = 0.8$
- Na convicção, tal como o *lift*, valores inferiores a 1 indicam que a associação entre A e B é aleatória

Referências I


Agrawal, R., Imieliński, T., and Swami, A. (1993).
Mining association rules between sets of items in large databases.
In *ACM SIGMOD International Conference on Management of Data*, volume 22, pages 207–216, Washington, D.C., USA. ACM.


Faceli, K., Lorena, A. C., Gama, J., and de Carvalho, A. E. P. L. F. (2011).
Inteligência artificial: uma abordagem de aprendizagem de máquina.
LTC, Rio de Janeiro.


Han, J., Pei, J., and Yin, Y. (2000).
Mining frequent patterns without candidate generation.
In *ACM SIGMOD International Conference on Management of Data*, pages 1–12, Dallas, Texas, USA. ACM, ACM.

Referências II


TAN, P.-N., STEINBACH, M., and KUMAR, V. (2009).
***Introdução ao data mining: mineração de dados*.**
Ciência Moderna, Rio de Janeiro.