

Big Data Mining

Frequent Pattern Mining

Albert Bifet (@abifet)



Paris, 19 October 2016
albert.bifet@telecom-paristech.fr

Association Rules

Dataset Example

Id	Items
1	bread, butter, milk
2	eggs, milk, yogurt
3	bread, cheese, eggs, milk
4	eggs, milk, yogurt
5	cheese, milk, yogurt

Association rule: **eggs, milk** \rightarrow **yogurt**

$$\text{conf}(X \rightarrow Y) = \frac{\text{sup}(X \cup Y)}{\text{sup}(X)}$$

Frequent Patterns

Suppose \mathcal{D} is a dataset of patterns, $t \in \mathcal{D}$, and min_sup is a constant.

Definition

Support (t): number of patterns in \mathcal{D} that are superpatterns of t .

Definition

Pattern t is *frequent* if $Support(t) \geq min_sup$.

Frequent Subpattern Problem

Given \mathcal{D} and min_sup , find all frequent subpatterns of patterns in \mathcal{D} .

Frequent Patterns

Suppose \mathcal{D} is a dataset of patterns, $t \in \mathcal{D}$, and min_sup is a constant.

Definition

Support (t): number of patterns in \mathcal{D} that are superpatterns of t .

Definition

Pattern t is **frequent** if $Support(t) \geq min_sup$.

Frequent Subpattern Problem

Given \mathcal{D} and min_sup , find all frequent subpatterns of patterns in \mathcal{D} .

Frequent Patterns

Suppose \mathcal{D} is a dataset of patterns, $t \in \mathcal{D}$, and min_sup is a constant.

Definition

Support (t): number of patterns in \mathcal{D} that are superpatterns of t .

Definition

Pattern t is **frequent** if $Support(t) \geq min_sup$.

Frequent Subpattern Problem

Given \mathcal{D} and min_sup , find all frequent subpatterns of patterns in \mathcal{D} .

Frequent Patterns

Suppose \mathcal{D} is a dataset of patterns, $t \in \mathcal{D}$, and min_sup is a constant.

Definition

Support (t): number of patterns in \mathcal{D} that are superpatterns of t .

Definition

Pattern t is **frequent** if $Support(t) \geq min_sup$.

Frequent Subpattern Problem

Given \mathcal{D} and min_sup , find all frequent subpatterns of patterns in \mathcal{D} .

Pattern Mining

Dataset Example

Document	Patterns
d1	abce
d2	cde
d3	abce
d4	acde
d5	abcde
d6	bcd

Itemset Mining

d1	abce
d2	cde
d3	abce
d4	acde
d5	abcde
d6	bcd

Support	Frequent
d1,d2,d3,d4,d5,d6	c
d1,d2,d3,d4,d5	e,ce
d1,d3,d4,d5	a,ac,ae,ace
d1,d3,d5,d6	b,bc
d2,d4,d5,d6	d,cd
d1,d3,d5	ab,abc,abe be,bce,abce
d2,d4,d5	de,cde

minimal support = 3

Itemset Mining

d1 abce
d2 cde
d3 abce
d4 acde
d5 abcde
d6 bcd

Support	Frequent
6	c
5	e,ce
4	a,ac,ae,ace
4	b,bc
4	d,cd
3	ab,abc,abe be,bce,abce
3	de,cde

Itemset Mining

d1	abce	Support	Frequent	Gen	Closed
d2	cde	6	c	c	c
d3	abce	5	e,ce	e	ce
d4	acde	4	a,ac,ae,ace	a	ace
d5	abcde	4	b,bc	b	bc
d6	bcd	4	d,cd	d	cd
		3	ab,abc,abe	ab	
			be,bce,abce	be	abce
		3	de,cde	de	cde

Itemset Mining

		Support	Frequent	Gen	Closed	Max
d1	abce	6	c	c	c	
d2	cde	5	e,ce	e	ce	
d3	abce	4	a,ac,ae,ace	a	ace	
d4	acde	4	b,bc	b	bc	
d5	abcde	4	d,cd	d	cd	
d6	bcd	3	ab,abc,abe be,bce,abce	ab be	abce	abce
		3	de,cde	de	cde	cde

Itemset Mining

		Support	Frequent	Gen	Closed	Max
d1	ab c e	6	c	c	c	
d2	c de	5	e,ce	e	ce	
d3	ab c e	4	a,ac,ae,ace	a	ace	
d4	a c de	4	b,bc	b	bc	
d5	ab c de	4	d,cd	d	cd	
d6	b c d	3	ab,abc,abe be,bce,abce	ab be	abce	abce
		3	de,cde	de	cde	cde

Itemset Mining

		Support	Frequent	Gen	Closed	Max
d1	ab c e	6	c	c	c	
d2	c de	5	e,ce	e	ce	
d3	ab c e	4	a,ac,ae,ace	a	ace	
d4	a c de	4	b,bc	b	bc	
d5	ab c de	4	d,cd	d	cd	
d6	bcd	3	ab,abc,abe be,bce,abce	ab be	abce	abce
	e → ce	3	de,cde	de	cde	cde

Itemset Mining

		Support	Frequent	Gen	Closed	Max
d1	ab ce	6	c	c	c	
d2	cde	5	e,ce	e	ce	
d3	ab ce	4	a,ac,ae,ace	a	ace	
d4	a cde	4	b,bc	b	bc	
d5	ab cde	4	d,cd	d	cd	
d6	bcd	3	ab,abc,abe be,bce,abce	ab be	abce	abce
		3	de,cde	de	cde	cde

Itemset Mining

		Support	Frequent	Gen	Closed	Max
d1	abce	6	c	c	c	
d2	cde	5	e,ce	e	ce	
d3	abce	4	a,ac,ae,ace	a	ace	
d4	acde	4	b,bc	b	bc	
d5	abcde	4	d,cd	d	cd	
d6	bcd	3	ab,abc,abe be,bce,abce	ab be	abce	abce
		3	de,cde	de	cde	cde

Itemset Mining

		Support	Frequent	Gen	Closed	Max
d1	abce	6	c	c	c	
d2	cde	5	e,ce	e	ce	
d3	abce	4	a,ac,ae,ace	a	ace	
d4	acde	4	b,bc	b	bc	
d5	abcde	4	d,cd	d	cd	
d6	bcd	3	ab,abc,abe	ab		
	a → ace		be,bce,abce	be	abce	abce
		3	de,cde	de	cde	cde

Itemset Mining

		Support	Frequent	Gen	Closed	Max
d1	abce	6	c	c	c	
d2	cde	5	e,ce	e	ce	
d3	abce	4	a,ac,ae,ace	a	ace	
d4	acde	4	b,bc	b	bc	
d5	abcde	4	d,cd	d	cd	
d6	bcd	3	ab,abc,abe be,bce,abce	ab be	abce	abce
		3	de,cde	de	cde	cde

Closed Patterns

Usually, there are too many frequent patterns. We can compute a smaller set, while keeping the same information.

Example

A set of 1000 items, has $2^{1000} \approx 10^{301}$ subsets, that is more than the number of atoms in the universe $\approx 10^{79}$

Closed Patterns

A priori property

If t' is a subpattern of t , then $\text{Support}(t') \geq \text{Support}(t)$.

Definition

A frequent pattern t is *closed* if none of its proper superpatterns has the same support as it has.

Frequent subpatterns and their supports can be generated from closed patterns.

Maximal Patterns

Definition

A frequent pattern t is *maximal* if none of its proper superpatterns is frequent.

Frequent subpatterns can be generated from maximal patterns, but not with their support.

All maximal patterns are closed, but not all closed patterns are maximal.

Non streaming frequent itemset miners

Representation:

- Horizontal layout

T1: a, b, c

T2: b, c, e

T3: b, d, e

- Vertical layout

a: 1 0 0

b: 1 1 1

c: 1 1 0

Search:

- Breadth-first (levelwise): Apriori
- Depth-first: Eclat, FP-Growth

The Apriori Algorithm

APRIORI ALGORITHM

- 1 Initialize the item set size $k = 1$
- 2 Start with single element sets
- 3 Prune the non-frequent ones
- 4 **while** there are frequent item sets
- 5 **do** create candidates with one item more
- 6 Prune the non-frequent ones
- 7 Increment the item set size $k = k + 1$
- 8 Output: the frequent item sets

The Eclat Algorithm

Depth-First Search

- divide-and-conquer scheme : the problem is processed by splitting it into smaller subproblems, which are then processed recursively
 - **conditional database for the prefix a**
 - transactions that contain a
 - **conditional database for item sets without a**
 - transactions that not contain a
- Vertical representation
- Support counting is done by intersecting lists of transaction identifiers

The FP-Growth Algorithm

Depth-First Search

- divide-and-conquer scheme : the problem is processed by splitting it into smaller subproblems, which are then processed recursively
 - **conditional database for the prefix a**
 - transactions that contain a
 - **conditional database for item sets without a**
 - transactions that not contain a
- Vertical and Horizontal representation : FP-Tree
 - prefix tree with links between nodes that correspond to the same item
- Support counting is done using FP-Tree

Mining Graph Data

Problem

Given a data set \mathcal{D} of graphs, find frequent graphs.

Transaction Id	Graph
1	$\begin{array}{c} \text{O} \\ \\ \text{C} - \text{C} - \text{S} - \text{N} \\ \\ \text{O} \end{array}$
2	$\begin{array}{c} \text{O} \\ \\ \text{C} - \text{C} - \text{S} - \text{N} \\ \\ \text{C} \end{array}$
3	$\begin{array}{c} \text{N} \\ \\ \text{C} - \text{C} - \text{S} - \text{N} \end{array}$

The gSpan Algorithm

$\text{GSPAN}(g, D, \text{min_sup}, S)$

Input: A graph g , a graph dataset D , min_sup .

Output: The frequent graph set S .

```
1  if  $g \neq \text{min}(g)$ 
2    then return  $S$ 
3  insert  $g$  into  $S$ 
4  update support counter structure
5   $C \leftarrow \emptyset$ 
6  for each  $g'$  that can be right-most
    extended from  $g$  in one step
7    do if  $\text{support}(g) \geq \text{min\_sup}$ 
8      then insert  $g'$  into  $C$ 
9  for each  $g'$  in  $C$ 
10    do  $S \leftarrow \text{GSPAN}(g', D, \text{min\_sup}, S)$ 
11  return  $S$ 
```