Examples on Finding Association Rules

Mauro Sozio

March 20, 2017

Naive Algorithm: Pass 1

Support treshold: 2.

Counters in Main Memory

d1	acde
d2	abc
d3	acd
d4	ab

Naive Algorithm: Pass 1

Counters in Main Memory
a:1,c:1,d:1,e:1

d1	**acde**
d2	abc
d3	acd
d4	ab

Naive Algorithm: Pass 1

Counters in Main Memory
a:2,c:2,d:1,e:1,b:1

d1	**acde**
d2	**abc**
d3	acd
d4	ab

Naive Algorithm: Pass 1

Counters in Main Memory
a:3,c:3,d:2,e:1,b:1

d1	**acde**
d2	**abc**
d3	**acd**
d4	ab

Naive Algorithm: Pass 1

Counters in Main Memory
a:4,c:3,d:2,e:1,b:2

d1	**acde**
d2	**abc**
d3	**acd**
d4	**ab**

Frequent items: a,c,d,b

Naive Algorithm: Pass 2

d1	**acde**
d2	abc
d3	acd
d4	ab

Counters in Main Memory
ac:1,ad:1,ae:1,cd:1,ce:1,de:1

Naive Algorithm: Pass 2

d1	**acde**
d2	**abc**
d3	acd
d4	ab

Counters in Main Memory
ac:2,ad:1,ae:1,cd:1,ce:1,de:1,ab:1,bc:1

Naive Algorithm: Pass 2

d1	**acde**
d2	**abc**
d3	**acd**
d4	ab

Counters in Main Memory

ac:3,ad:2,ae:1,cd:2,ce:1,de:1,ab:1,bc:1

Naive Algorithm: Pass 2

Counters in Main Memory
ac:3,ad:2,ae:1,cd:2,ce:1,de:1,ab:2,bc:1

d1	**acde**
d2	**abc**
d3	**acd**
d4	**ab**

Naive Algorithm: Pass 2

Counters in Main Memory
ac:3,ad:2,ae:1,cd:2,ce:1,de:1,ab:2,bc:1

d1	**acde**
d2	**abc**
d3	**acd**
d4	**ab**

Frequent 2-itemsets: ac, ad, cd, ab

Naive Algorithm: Pass 3

Counters in Main Memory
acd:1,ace:1,ade:1,cde:1

d1	**acde**
d2	abc
d3	acd
d4	ab

Naive Algorithm: Pass 3

Counters in Main Memory
acd:1,ace:1,ade:1,cde:1,abc:1

d1	**acde**
d2	**abc**
d3	acd
d4	ab

Naive Algorithm: Pass 3

Counters in Main Memory
acd:2,ace:1,ade:1,cde:1,abc:1

d1	**acde**
d2	**abc**
d3	**acd**
d4	ab

Naive Algorithm: Pass 3

Counters in Main Memory
acd:2,ace:1,ade:1,cde:1,abc:1

d1	**acde**
d2	**abc**
d3	**acd**
d4	ab

Frequent 3-Itemsets: acd

A-priori algorithm: Pass 1 (same as Naive)

Counters in Main Memory
a:4,c:3,d:2,e:1,b:2

d1	**acde**
d2	**abc**
d3	**acd**
d4	**ab**

Frequent items: a,c,d,b

A-priori algorithm: Pass 2

d1	acde
d2	abc
d3	acd
d4	ab

Frequent items

a,c,d,b

Counters in Main Memory

A-priori algorithm: Pass 2

d1	**acde**
d2	abc
d3	acd
d4	ab

Frequent items
a,c,d,b
Counters in Main Memory
ac:1,ad:1,cd:1

Note: We don't keep any counter for ae,ce,de as 'e' is not frequent so none of them can be frequent!

A-priori algorithm: Pass 2

d1	**acde**
d2	**abc**
d3	acd
d4	ab

Frequent items
a,c,d,b
Counters in Main Memory
ac:2,ad:1,cd:1,ab:1,bc:1

A-priori algorithm: Pass 2

d1	**acde**
d2	**abc**
d3	**acd**
d4	ab

Frequent items
a,c,d,b
Counters in Main Memory
ac:3,ad:2,cd:2,ab:1,bc:1

A-priori algorithm: Pass 2

d1	**acde**
d2	**abc**
d3	**acd**
d4	**ab**

Frequent items
a,c,d,b
Counters in Main Memory
ac:3,ad:2,cd:2,ab:2,bc:1

Frequent 2-items: ac, ad,cd,ab

A-priori algorithm: Pass 3

d1	acde
d2	abc
d3	acd
d4	ab

Frequent 2-items

ac, ad,cd,ab

Counters in Main Memory

A-priori algorithm: Pass 3

d1	**acde**
d2	abc
d3	acd
d4	ab

Frequent 2-items
ac, ad,cd,ab
Counters in Main Memory
acd:1

Note: We don't maintain a counter for 'ace','ade','cde' as 'ce','de' are not frequent.

A-priori algorithm: Pass 3

d1	**acde**
d2	**abc**
d3	acd
d4	ab

Frequent 2-items
ac, ad,cd,ab
Counters in Main Memory
acd:1

Note: We don't maintain a counter for 'abc',as 'bc' is not frequent.

A-priori algorithm: Pass 3

d1	**acde**
d2	**abc**
d3	**acd**
d4	ab

Frequent 2-items
ac, ad,cd,ab
Counters in Main Memory
acd:2

A-priori algorithm: Pass 3

d1	**acde**
d2	**abc**
d3	**acd**
d4	ab

Frequent 2-items
ac, ad,cd,ab
Counters in Main Memory
acd:2

Frequent 3-itemsets: acd

PCY algorithm

Input data:

d1	2,3,4
d2	1,4
d3	2,3

Hash function $f(i,j) = i + j \mod 3$, where mod denotes the remainder of the integer division. E.g. $f(2,4) = 6 \mod 3 = 0$, $f(1,4) = 5 \mod 3 = 2$. Therefore we have three buckets, one for each value of the hash function.

Support threshold $= 2$.

PCY algorithm: Pass 1

d1	2,3,4
d2	1,4
d3	2,3

Counters for the itemsets

Counters for the buckets

PCY algorithm: Pass 1

d1	**2,3,4**
d2	1,4
d3	2,3

Counters for the itemsets
2:1,3:1,4:1
Counters for the buckets
B0:1,B1:1,B2:1

Note: in bucket 0 we have the pair (2,4), in 1 the pair (3,4) in 2 the pair (2,3). We don't keep in main memory all the elements in the buckets but only a counter.

PCY algorithm: Pass 1

d1	**2,3,4**
d2	**1,4**
d3	2,3

Counters for the itemsets
2:1,3:1,4:2,1:1
Counters for the buckets
B0:1,B1:1,B2:2

PCY algorithm: Pass 1

d1	**2,3,4**
d2	**1,4**
d3	**2,3**

Counters for the itemsets

2:2,3:2,4:2,1:1

Counters for the buckets

B0:1,B1:1,B2:3

PCY algorithm: Pass 1

d1	**2,3,4**
d2	**1,4**
d3	**2,3**

Counters for the itemsets

2:2,3:2,4:2,1:1

Counters for the buckets

B0:1,B1:1,B2:3

Frequent items: 2,3,4

PCY algorithm: Pass 2

d1	2,3,4
d2	1,4
d3	2,3

Frequent items
2,3,4
Counters for the buckets
B0:1,B1:1,B2:3
Counters for the itemsets

PCY algorithm: Pass 2

d1	**2,3,4**
d2	1,4
d3	2,3

Frequent items
2,3,4
Counters for the buckets
B0:1,B1:1,B2:3
Counters for the itemsets
(2,3):1

Note: we don't maintain a counter for (2,4) and (3,4) as they hash to non-frequent buckets.

PCY algorithm: Pass 2

d1	**2,3,4**
d2	**1,4**
d3	2,3

Frequent items
2,3,4
Counters for the buckets
B0:1,B1:1,B2:3
Counters for the itemsets
(2,3):1

Note: we don't maintain a counter for (1,4) as 1 is not frequent.

PCY algorithm: Pass 2

d1	**2,3,4**
d2	**1,4**
d3	**2,3**

Frequent items
2,3,4
Counters for the buckets
B0:1,B1:1,B2:3
Counters for the itemsets
(2,3):2

PCY algorithm: Pass 2

d1	**2,3,4**
d2	**1,4**
d3	**2,3**

Frequent items
2,3,4
Counters for the buckets
B0:1,B1:1,B2:3
Counters for the itemsets
(2,3):2

Frequent 2-itemsets:(2,3)

PCY algorithm: Pass 2

d1	**2,3,4**
d2	**1,4**
d3	**2,3**

Frequent items
2,3,4
Counters for the buckets
B0:1,B1:1,B2:3
Counters for the itemsets
(2,3):2

Frequent 2-itemsets:(2,3)

The remaining passes are the same of A-priori.