Comp7103B: First Assignment

**Question** 1 Frequent Itemsets

**a)** Table 1 shows a list of baskets as well as the items they contain. For example, this could be the set of products bought by each customer during a single trip to a grocery store.  
Using the **A-priori** algorithm, find all frequent itemsets with support threshold **0.4** (i.e. in this example they occur at least 40% of 7 times, i.e. at least three times.) In particular you should specify for each pass of the algorithm, the frequent itemsets, as well as the counters (and their values) kept in main memory by the A-priori algorithm.

|  |  |
| --- | --- |
| ID | Baskets |
| 1 | a,b,c,e |
| 2 | a,d,b |
| 3 | c,b |
| 4 | a,b,d,e |
| 5 | b,d |
| 6 | a,b |
| 7 | a |

Since Threshold = 0.4 = 40%

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Items | Transaction ID | | | | | | | Support |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| a | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 5 / 7 = 71% |
| b | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 6 / 7 = 85% |
| c | 1 | 0 | 1 | 0 | 0 | 0 | 0 | ~~2 / 7 = 28%~~ |
| d | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 3 / 7 = 42% |
| e | 1 | 0 | 0 | 1 | 0 | 0 | 0 | ~~2 / 7 = 28%~~ |

We will keep **{a, b, d}**

Itemset after 1st compute = {a,b}, {a,d}, {b,d}

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Items | Transaction ID | | | | | | | Support |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| {a, b} | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 4 / 7 = 57% |
| {a, d} | 0 | 1 | 0 | 1 | 0 | 0 | 0 | ~~2 / 7 = 28%~~ |
| {b, d} | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 3 / 7 = 42% |

We will keep **{a, b} , {b, d}**

**All frequent itemset**

|  |  |  |
| --- | --- | --- |
| **Frequent itemset** | Support | **Size** |
| {a} | 5 / 7 = 71% | 1 |
| {b} | 6 / 7 = 85% | 1 |
| {d} | 3 / 7 = 42% | 1 |
| {a, b} | 4 / 7 = 57% | 2 |
| {b, d} | 3 / 7 = 42% | 2 |

**Main memory**

Frequent items

{a}, {b},{d}, {a,b}, {b,d}

Counts of pairs of frequent items

= 2

{a, b}, {b, d}

Item counts

a,b,c,d,e = 5

Unused

**Pass 1 Pass 2**

**b)** use the results from the previous step so as to compute the confidence and support of the rule b -> d. In order to answer this question, are the results from the previous step sufficient or some information is missing?

**Confidence** (b -> d) = **Support** (b, d) / **Support** (b)

= (3 / 7) / (6 / 7)

= (3 / 7) \* (7 / 6)

= 1 / 2

= 50**%**

**Lift** (b -> d)= **Support** (b, d) / (**Support** (b) \* **Support** (d))

= (3 / 7) / ( (6 / 7) \* (3 / 7) )

= 7 / 6

= 1.166

Result from previous step is already sufficient.

Since be got all **Support** (b, d), **Support** (b) and **Support** (d)

**Table 2**

|  |  |
| --- | --- |
| ID | Baskets |
| 1 | 1,3,4 |
| 2 | 4,5 |
| 3 | 2,7 |
| 4 | 1,6 |
| 5 | 2,7 |
| 6 | 3 |

**c)** Table 2 shows a new list of baskets as well as the items they contain. Using the PCY algorithm, find all frequent itemsets with support threshold 0.33 (i.e. in this example they occur at least 33% of 6, i.e. at least two times.) To this end, we are going to use the hash function

f (i, j) = (i + j) % 3 (% and mod are equivalent). In particular you should specify for each pass of the algorithm, the frequent itemsets, as well as all the counters (and their values) kept in main

memory by the PCY algorithm.

threshold = **0.33** = **33% (**at least **2** times**)**

**Pass 1**

|  |  |
| --- | --- |
| Item set | Support (times) |
| {1} | 2 |
| {2} | 2 |
| {3} | 2 |
| {4} | 2 |
| {5} | 1 |
| {6} | 1 |
| {7} | 2 |

Transaction 1 : 1, 3, 4

Itemset 1 : {1, 3}, {1, 4}, {3,4}

~~Counter 1 : 1 , 1 1~~

All pairs in Transaction 1 appear **less** than threshold so we are not going to use it.

~~Transaction 2 : 4, 5 (5 is not a frequent item)~~

Transaction 3 : 2, 7

Itemset 3 : {2, 7}

Counter 3 : 2

Hash 3 : 0

~~Transaction 4 : 1, 6 (6 is not a frequent item)~~

Transaction 5 : 2, 7

Itemset 5 : {2, 7}

Counter 5 : 2

Hash 5 : 0

~~Transaction 6 : 3 no pair~~

Hash Table

|  |  |
| --- | --- |
| Bucket | 0 |
| Hash Counter | 2 |

**Pass 2**

|  |  |  |  |
| --- | --- | --- | --- |
| Bit Vector | Bucket | Hash Counter | Pairs |
| 1 | 0 | 2 | {2, 7} |

Frequent Bucket: 0

**Memory**

Count of candidate pairs = 1

Frequent items

{2, 7}

Item counts

1,2,3,4,7 = 5

Hash Table

Bitmap

1

**Pass 1 Pass 2**