CSE 40647/60647: Data Science	Fall 2019
Homework 4: Written Assignments	
Handed Out: November 14, 2019	*Due: November 29, 2019 11:55pm*

Save your homework submission as *NETID-hw4-written.pdf*.

1 FP-Growth (30 points)

A database has 10 transactions. Let $min\_sup = 2$. Items are a, b, c, d, and e.

Trans. ID	Itemset
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}

Draw the first FP-tree that the FP-Growth algorithm creates when given this transaction database. By saying the "first", this FP-tree should not be a conditional FP-tree. Use FP-Growth to find all the frequent patterns and their support. Attach the FP-tree (either typed or hand-written+scanned) and write down the patterns and support in your PDF.

Solution:

The FP-Tree is shown below in Fig. 1 (Dashed links between nodes of same labels are not required) (10 pts)

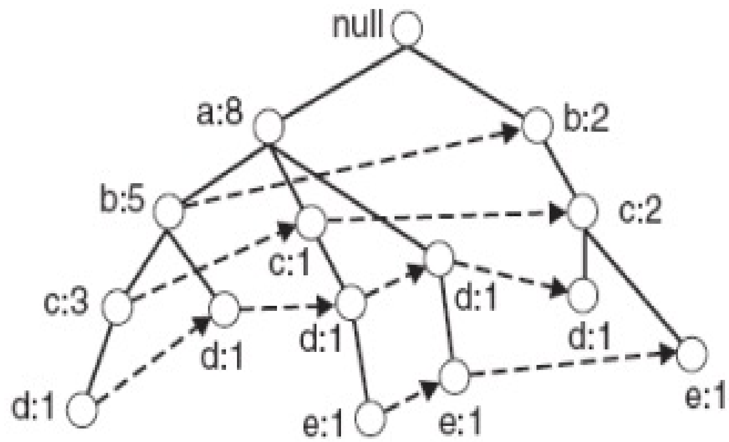

Figure 1: FP tree.

Apriori algorithm need to generate a huge number of candidate sets and may need repeatly scan the whole database and check a large set of candicates by pattern matching. It is costly to go over each transaction in the databse to determine the support of the candidate itemsets. While FP-growth, adopts a divide-and-conquer strategy which may substantially reduce the size of the data sets to be searched.

Frequent itemsets found (descending order by frequency of each item) (4 pts each):

Item	Frequent Patterns
a	**a**
b	**b**, **ba** (from b-conditional FP-tree)
c	**c**, **cb** and **ca** (from c-conditional FP-tree), **cba** (from cb-conditional FP-tree)
d	**d**, **dc** and **db** and **da** (from d-conditional FP-tree),
	dcb and **dca** (from dc-conditional FP-tree), **dba** (from db-conditional FP-tree)
e	**e**, **ed** and **ec** and **ea** (from e-conditional FP-tree), **eda** (from ed-conditional FP-tree)

Conditional FP-trees can easily be derived from Fig. 1 step by step.

2 Pattern Evaluation Measures (10 points)

The definitions of two measures, *lift* and *cosine*, look rather similar as shown below,

$$\text{lift}(A, B) = \frac{s(A \cup B)}{s(A) \times s(B)}, \tag{1}$$

and

$$\text{cosine}(A, B) = \frac{s(A \cup B)}{\sqrt{s(A) \times s(B)}}, \tag{2}$$

where $s(X)$ is the *relative* support of itemset X. Which measure is *null-invariant*, and which is not, and why? Can you prove it? You must formally define what is null-invariant using the symbols and give your proof.

Solution:
A measure is null-invariant if the value of the measure does not change with the number of null-transactions.
cosine is null-invariant while *lift* is not.
Let n be the total number of transactions, and $count(\neg(A \cup B))$ be the number of null-transactions.

$$
\begin{aligned}
\mathit{lift}(A, B) &= \frac{s(A \cup B)}{s(A) \times s(B)} \\
&= \frac{count(A \cup B)}{n} / \left(\frac{count(A)}{n} \times \frac{count(B)}{n}\right) \\
&= \frac{count(A \cup B) \times n}{count(A) \times count(B)} \\
&= \frac{count(A \cup B) \times (count(A \cup B) + count(\neg(A \cup B))}{count(A) \times count(B)}
\end{aligned}
$$

$$
\begin{aligned}
\mathit{cosine}(A, B) &= \frac{s(A \cup B)}{\sqrt{s(A) \times s(B)}} \\
&= \frac{\frac{count(A \cup B)}{n}}{\sqrt{\frac{count(A)}{n} \times \frac{count(B)}{n}}} \\
&= \frac{count(A \cup B)}{\sqrt{count(A) \times count(B)}}
\end{aligned}
$$

We can clearly see that *cosine* is invariant with the number of null-transactions, while *lift* is not.

3 Closed Patterns (20 points)

A database has 4 transactions as shown below. Let $min\_sup = 2$. Items are A, B, C, D, E, F, and G.

Trans. ID	Itemset
1	{A, C, F, G}
2	{A, B, C, F}
3	{A, B, C, D, F}
4	{B, D, E}

Which patterns from the following are **closed patterns**? Please briefly describe your idea for each pattern on why it is closed or not.

- Pattern 1: {D}
- Pattern 2: {A, B, C, F}
- Pattern 3: {B, F}
- Pattern 4: {B, D}
- Pattern 5: {A, C, F}

Solution: (4 pts each)
Idea: according to the definition of closed pattern, which is: A pattern X is closed if X is frequent and there exists no super pattern Y such that $X \subset Y$, and X and Y has the same support.
Pattern 1 {**D**}: not closed pattern because $\{D\} \subset \{B, D\}$ and they have the same support 2.
Pattern 2 {**A, B, C, F**}: closed pattern.
Pattern 3 {**B, F**}: not closed pattern because $\{B, F\} \subset \{A, B, C, F\}$ and they have the same support 2.
Pattern 4 {**B, D**}: closed pattern.
Pattern 5 {**A, C, F**}: closed pattern.

4 Sequential Patterns (20 points)

A sequence database has 3 sequences as shown below. Items in the same parenthesis means they were got together in one event. Let $min\_sup = 2$. Items are A, B, C, D, F, and G. Which patterns from the following are **sequential patterns**? Please briefly describe your idea for each pattern on why it is a good sequential pattern or not.

Seq. ID	Sequence
1	(AB)C(FG)G
2	(AD)CB(ABF)
3	AB(FG)

- Pattern 1: ACF
- Pattern 2: (FG)B
- Pattern 3: (FG)
- Pattern 4: B(FG)
- Pattern 5: GF

Solution: (4 pts each)
Idea: sequential pattern should has $min\_sup = 2$.
Pattern 1 ACF: Sequential pattern, has $min\_sup = 2$ and can be found in Seq1 and Seq2.
Pattern 2 (FG)B: Not sequential pattern because its support is 0.
Pattern 3 (FG): Sequential pattern, has $min\_sup = 2$ and can be found in Seq1 and Seq3.
Pattern 4 B(FG): Sequential pattern, has $min\_sup = 2$ and can be found in Seq1 and Seq3.
Pattern 5 GF: Not sequential pattern because its support is 0.

CSE 40647/60647: Data Science	Fall 2019

Homework 4: Programming Assignments

Handed Out: November 12, 2019 | *Due: November 25, 2019 11:55pm*

Save your homework submission as *NETID-hw4-programming.zip*. The zip file has one pdf file *NETID-hw4-programming.pdf*, one code file, the Dataset-apriori.txt file, and one README file.
In the README file, please specify the python version you used and how to run your code in command line.

Apriori (50 points)

Please use **Python** to solve the problem. You are NOT allowed to directly call any frequent pattern mining functions (like the Apriori functions in Scikit).

A database has 10 transactions. Let $min\_sup = 2$. Items are a, b, c, d, and e.

Trans. ID	Itemset
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}

Use Python to implement Apriori to find all frequent patterns (i.e., frequent itemsets) and their counts from the transaction database.

Output: Write down the patterns and their support in the pdf. Save your code as NETID-hw4.py.

Solutions:
1-itemset candidates C1:
a: 8; b: 7; c: 6; d: 5; e: 3.
Compare C1 with minimum support. The frequent 1-itemsets F1:
a: 8; b: 7; c: 6; d: 5; e: 3.
2-itemset candidates C2: ab: 5; ac: 4; ad: 4; ae: 2; bc: 5; bd: 3; be: 0; cd: 3; ce: 2; de: 2.
Compare C2 with minimum support. The frequent 2-itemsets F2:

ab: 5; ac: 4; ad: 4; ae: 2; bc: 5; bd: 3; cd: 3; ce: 2; de: 2.
3-itemset candidates C3: abc: 3; abd: 2; bcd: 2; cde: 1; acd: 2; ace: 1; ade: 2.
Compare C3 with minimum support. The frequent 3-itemsets F3:
abc: 3; abd: 2; bcd: 2; acd: 2; ade: 2.

There is no frequent 4-itemsets.