P76074648 張媤婷 Project1

檔案說明：

Homework1\_Apriori.py : 實作Apriori演算法的程式碼

Homework2\_FP-tree.py : 實作FP-Growth的程式碼

Book\_data : 課程講義上Apriori的範例資料

Book\_data2.txt : 課程講義上FP-tree的範例資料

Data.data : 由IBM Quest Data Generator產生的小型資料

IBM.data : 由IBM Quest Data Generator產生的大型資料

Output\_Testing\_Data.csv.csv : 由Data.data轉換成WEKA所能讀取的資料

實作方法：

Apriori： 用list來存每次產生出來的C(n)與L(n)，並用getWinner(k)來取出特定L(k)，取出的資料可以用來組成新的L(k+1)；另一getLoser()函式，用來找出被pruning掉的element，用這些elements來掃描L(k+1)，不斷重複，直到找不到新的一個Level。

FP-Growth：新增class Tree()，Tree有parent、child、sibling、next、data、num這些屬性，分別用來記錄物件的父親、第一個孩子、父親的下一個孩子，下一個相同值的物件、這個物件出現的次數。建立樹的方向是由上往下，經歷相同prefix的物件時不新增物件，只增加次數(num)；找尋Conditional Patterns由鎖定的item開始往上找，找完就前往next再往上找。最後用找到的pattern去找出frequent patterns。

效能分析：

Apriori versus FP-Growth Support=0.01

1. Book\_data : 課程講義上Apriori的範例資料，4筆交易

Apriori ：It takes 2 milliseconds to find the above patterns

FP-Growth ：It takes 1 milliseconds to find the above patterns

1. Book\_data2.txt : 課程講義上FP-tree的範例資料，5筆交易

Apriori ：It takes 25 milliseconds to find the above patterns

FP-Growth ：It takes 6.4 milliseconds to find the above patterns

1. Data.data : 由IBM Quest Data Generator產生的小型資料，37筆交易

Apriori ：It takes 25 milliseconds to find the above patterns

FP-Growth ：It takes 26 milliseconds to find the above patterns

1. IBM.data : 由IBM Quest Data Generator產生的大型資料，24154筆交易

Apriori ：It takes 1374288 milliseconds to find the above patterns

FP-Growth ：It takes 320436 milliseconds to find the above patterns

第一、二、三種Dataset的運算時間為十次下來的平均值，可以觀察出運算時間的差異度不大，但第四種dataset

Aprior\_WEKA

It takes 1374288 milliseconds to find the above patterns

=== Run information ===

Scheme: weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

Relation: Apriori\_output-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last

Instances: 37

Attributes: 20 (Transation &1~19)

=== Associator model (full training set) ===

Apriori

=======

Minimum support: 0.95 (35 instances)

Minimum metric <confidence>: 0.9

Number of cycles performed: 1

Generated sets of large itemsets:

Size of set of large itemsets L(1): 12

Size of set of large itemsets L(2): 66

Size of set of large itemsets L(3): 220

Size of set of large itemsets L(4): 495

Size of set of large itemsets L(5): 792

Size of set of large itemsets L(6): 924

Size of set of large itemsets L(7): 792

Size of set of large itemsets L(8): 495

Size of set of large itemsets L(9): 220

Size of set of large itemsets L(10): 66

Size of set of large itemsets L(11): 12

Size of set of large itemsets L(12): 1

Best rules found:

1. 2=0 37 ==> 1=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

2. 1=0 37 ==> 2=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

3. 10=0 37 ==> 1=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

4. 1=0 37 ==> 10=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

5. 11=0 37 ==> 1=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

6. 1=0 37 ==> 11=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

7. 12=0 37 ==> 1=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

8. 1=0 37 ==> 12=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

9. 13=0 37 ==> 1=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

10. 1=0 37 ==> 13=0 37 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)

FP-Growth

It takes 320436 milliseconds to find the above patterns

=== Run information ===

Scheme: weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.01

Relation: Apriori\_output-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last-weka.filters.unsupervised.attribute.NumericToBinary-Rfirst-last

Instances: 37

Attributes: 20 (Transation & 1~19)

=== Associator model (full training set) ===

FPGrowth found 523250 rules (displaying top 10)

1. [2=0]: 37 ==> [19=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

2. [19=0]: 37 ==> [2=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

3. [2=0]: 37 ==> [18=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

4. [18=0]: 37 ==> [2=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

5. [2=0]: 37 ==> [17=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

6. [17=0]: 37 ==> [2=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

7. [2=0]: 37 ==> [16=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

8. [16=0]: 37 ==> [2=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

9. [2=0]: 37 ==> [15=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)

10. [15=0]: 37 ==> [2=0]: 37 <conf:(1)> lift:(1) lev:(0) conv:(0)