Data Mining Project 1 Report

Author Infomation

- Name: 洪裕翔
- Grade: 資訊所碩士一年級
- Student ID: P76124215

1.1 Scenario Analysis

> What do you observe in the below 4 scenarios? What could be the reason?
> (For both support and confidence, you should set 0.05 for low and 0.2 or more for high)
> - High support, high confidence
> - High support, low confidence
> - Low support, high confidence
> - Low support, low confidence

Support	Confidence	#FrequentItemsets	#Rules
High (0.2)	High (0.2)	146	326
High (0.2)	Low (0.05)	146	326
Low (0.05)	High (0.2)	5399	41374
Low (0.05)	Low (0.05)	5399	59472

Table 1: Scenario Statistics

表一之實驗結果分析如下：

1. 將控制變因設為 support，觀察相同 support、不同 confidence 的結果
    - #FrequentItemsets 會相同

> 造成此現象的原因為，兩種演算法（Apriori, FP-Growth）的目標都是為了得到輸入資料的 frequent itemsets，之後再根據 frequent itemsets 找出 rules；並且兩者在計算 frequent itemsets 時，過濾的標準相同（只取 support 大於等於 min support 的 itemsets），因此只要 min support 相同，不論 min confidence 的值為何，皆不影響兩者最後求得之 #FrequentItemsets 結果。

- Min confidence 較小，#Rules 會較大

> 本實驗在 high support 的 #Rules 結果是特例（#Rules 相同），理論上固定 support，#Rules 便會與 min confidence 呈負相關。
> 造成此現象的原因為，confidence 的定義是條件機率（$A \Rightarrow B = \frac{P(B|A)}{P(B)}$），min confidence 較小的值意味著，輸出的 rules 的條件機率門檻不高（大於 min confidence 就能被輸出），因此較小的 min confidence 會使得輸出條件相對寬鬆，#Rules 自然就會比較大，反之則較小。

2. 將控制變因設為 confidence，觀察相同 confidence、不同 support 的結果
   - Min support 越大，#FrequentItemsets 會較小

> 造成此現象的原因為，兩種演算法（Apriori, FP-Growth）計算 frequent itemsets 的標準皆為「itemset 的 support 大於等於 min support，即為 frequent itemset」，min support 可以被理解為門檻，若 min support 的值越大，itemset 要被判定成 frequent itemset 需要的 support 便越大，故 #FrequentItemsets 會越小，反之則越大。

1.2 Runtime Statistics

> Report the run time for both algorithms for the above 4 scenarios in a table.
> Try to provide an explanation for the runtime statistics.

Support	Confidence	Algorithm Runtime (sec)	Mining Time (sec)	Total Time (sec)
High (0.2)	High (0.2)	1.02	0.76	1.78
High (0.2)	Low (0.05)	1.11	0.80	1.91
Low (0.05)	High (0.2)	48.7	109.98	158.68
Low (0.05)	Low (0.05)	47.58	126.23	173.81

Table 2: Apriori Algorithm Runtime Statistics.

Support	Confidence	Algorithm Runtime (sec)	Mining Time (sec)	Total Time (sec)
High (0.2)	High (0.2)	0.45	0.77	1.22
High (0.2)	Low (0.05)	0.44	0.72	1.16
Low (0.05)	High (0.2)	2.59	109.68	112.27
Low (0.05)	Low (0.05)	2.31	123.74	126.05

Table 3: FP-Growth Algorithm Runtime Statistics.

比較表二、表三可以得知：

1. FP-Growth 在四種 scenario 的演算法執行時間（Algorithm Runtime）都遠勝於 Apriori，尤其以 low support 的兩種 scenario 最為明顯。
2. 將 support 設為控制變因
    - 大部分的情況下出現 low confidence scenario 所需的演算法執行時間較短的結果。

> 雖然在兩個表格中出現了三個此現象，但以我的實作方法來說，confidence 理論上並不會影響演算法的執行時間（Algorithm Runtime）。
> 原因為：我實作的 Apriori 和 FP-Growth 都是基於 support 計算 frequent itemsets 的，confidence 則是等到計算出所有 itemsets 後，才會在挖掘 rules 時被使用到。
> 因此，此條件下的 confidence 應只會影響 Mining Time 以及 Total Time，並不會影響演算法執行時間。

- low confidence scenario 可能需要較長的挖掘時間的現象。

> 上一點已有提過，在我的實作方法下，confidence 只會影響 Mining Time 以及 Total Time，且為負相關影響。
> 同樣是前面有提到（1.1 scenario analysis），low confidence 會讓 rules 的生成門檻變低，#Rules 會較多，因此可能造成更久的挖掘時間（Mining Time）。

3. 將 confidence 設為控制變因
   - Support 對於各種時間都影響巨大
     - support 會直接影響 #FrequentItemsets，因此 support 對於演算法執行時間（Algorithm Runtime）影響很大，因為 #FrequentItemsets 多，就需要比較多的演算法執行時間，反之則較少。
     - #FrequentItemsets 會影響 #Rules，也就是說 support 會間接影響 #Rules。若資料的特徵分佈平均（假設 confidence 的資料數量分佈大致雷同），#FrequentItemsets 越多便代表 #Rules 也會越多，這會導致 support 對於挖掘時間（Mining Time）的影響也很大。

推測：

1. 雖然 FP-Growth 在本實驗中效能表現良好，但由於其建樹的成本高，因此在特定情況下，Apriori 反而有可能因為其運算方法簡單，而取得較好的演算法執行時間表現。

1.3 Interested Topics

> Any topics you are interested in?

1. Python 的語言特性
    - 內建的資料結構（Set, Frozenset）
    - 不像 C/Cpp 一樣有明確指針，但也能做到類似的功能
    - 小數點誤差問題

Bonus

- Apply your algorithms to another dataset from Kaggle or UCI.
- Do some experiments (eg. observe the 4 scenarios as requested for other datasets).
- Make sure to specify the name of self-selected dataset(s), and include your discoveries in the report.

Basic Information

- Dataset Name: Groceries
- Dataset Source: Kaggle
- Dataset Link: **https://www.kaggle.com/datasets/heeraldedhia/groceries-dataset/** (https://www.kaggle.com/datasets/heeraldedhia/groceries-dataset/)

Preprocess Method

Kaggle 上該資料集的檔案為一個 csv 檔案。
我移除檔案中的購買日期資訊，只保留顧客編號、商品描述，將其依照顧客編號升冪排列，並格式化成和 `ibm-2023-released.txt` 相同格式的文件，便能不更動程式碼，挖掘此資料集。

Experiments

Support	Confidence	#FrequentItemsets	#Rules
0.5	0.5	0	0
0.25	0.25	0	0
0.1	0.1	4	0
0.01	0.01	64	0
0.01	0.001	64	0
0.001	0.001	719	1148
0.001	0.01	719	1122

Table 4: Bonus Scenario Statistics

Support	Confidence	Algorithm Runtime (sec)	Mining Time (sec)	Total Time (sec)
0.50	0.50	0.27	0.00	0.27
0.25	0.25	0.25	0.00	0.25
0.10	0.10	0.33	0.01	0.34
0.01	0.01	3.46	0.13	3.59
0.01	0.001	3.29	0.14	3.43
0.001	0.001	32.78	4.77	37.55
0.001	0.01	32.94	4.77	37.71

Table 5: Bonus Apriori Algorithm Runtime Statistics.

Support	Confidence	Algorithm Runtime (sec)	Mining Time (sec)	Total Time (sec)
0.50	0.50	0.01	0.00	0.01
0.25	0.25	0.01	0.00	0.01
0.10	0.10	0.02	0.01	0.03
0.01	0.01	0.07	0.10	0.17
0.01	0.001	0.08	0.10	0.18
0.001	0.001	0.12	4.55	4.67
0.001	0.01	0.13	4.54	4.67

Table 6: Bonus FP-Growth Algorithm Runtime Statistics.

觀察表四至六可以發現：

1. 相同 support
    - #FrequentItemsets 會相同
        - Support = 0.01, #FrequentItemsets = 64
        - Support = 0.001, #FrequentItemsets = 719
    - Min confidence 較小，#Rules 會較大
        - Support = 0.001, confidence = 0.01 , #Rules = 1122
        - Support = 0.001, confidence = 0.001 , #Rules = 1148
    - low confidence scenario 可能需要較長的挖掘時間的現象

        - Support = 0.001, confidence = 0.01
            - Apriori mining time = 4.77
            - Fp-growth mining time = 4.54
        - Support = 0.001, confidence = 0.001
            - Apriori mining time = 4.77
            - Fp-growth mining time = 4.55

2. 相同 confidence
    - Min support 越大，#FrequentItemsets 會較小
        - Confidence = 0.001, support = 0.001 , #FrequentItemsets = 719
        - Confidence = 0.001, support = 0.01 , #FrequentItemsets = 64
    - Support 對於各種時間都影響巨大
        - Confidence = 0.01, support = 0.01
            - Apriori runtime = 3.46
            - Fp-growth runtime = 0.07
            - Apriori mining time = 0.13
            - Fp-growth mining time = 0.10
        - Confidence = 0.01, support = 0.001
            - Apriori runtime = 32.94
            - Fp-growth runtime = 0.13
            - Apriori mining time = 4.77
            - Fp-growth mining time = 4.54

3. 演算法效能表現
    - FP-Growth 效能遠比 Apriori 好很多

Conclusion

可以看出，以上結果完全符合我於 1.1 Scenario Analysis、1.2 Runtime Statistics 所提出之分析。