資料探勘 作業一

資訊所 P76074509 辜玉雯

- 目的：

  此次作業是實作 Apriori 演算法與 FP-growth 演算法達到找出 frequent itemset 與 association rules。

- Dataset：

  這裡使用兩種 dataset，分別為 IBM Quest Synthetic Data Generator 與 Kaggle 上的資料。

  Kaggle 上的資料大都需要經過前處理，在此找到的是零售紀錄，包含了某次銷售的發票編號、消費者編號、以及所購買的貨品代號。是格式接近於 IBM 產生器所產生的資料。總計約有 4 萬 5 千筆銷售紀錄，3645 種貨品。

- 環境：

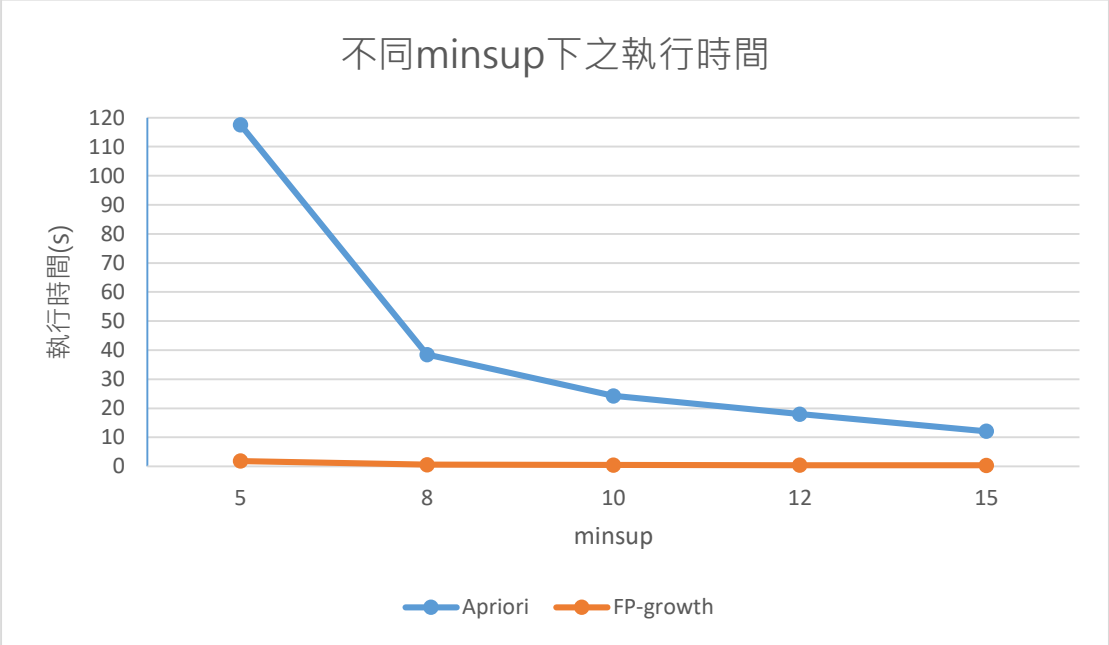
實作語言	**Python**
CPU	intel i5-6400 2.7GHz
Memory	16G
OS	Windows 10 x64

- 實驗：

實驗一、

比較在不同 minimum support 的狀況下，兩種演算法執行時間的差異。在此我們將 minimum confidence 固定為 0.6，用以單獨比較 minimum support 的影響。

minsup	5	8	10	12	15
Frequent itemsets(項)	16060	6097	3958	2763	1803
Rules(條)	4685	545	166	67	24



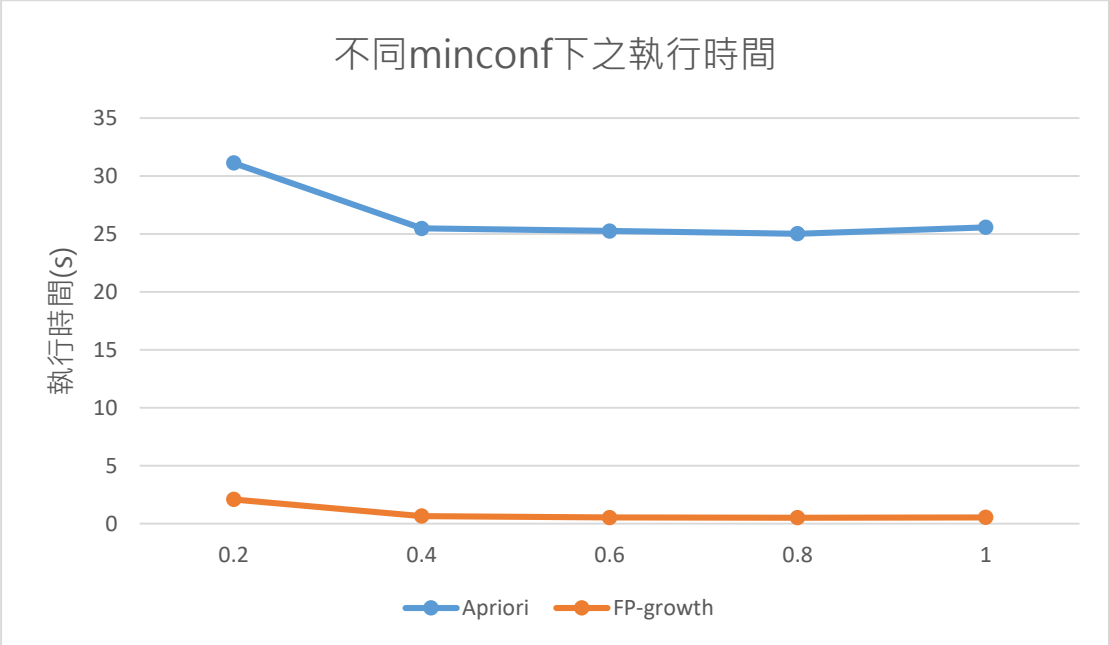
實驗結果：

可以發現 FP-growth 的效能明顯優於 Apriori 演算法，FP-growth 都在 1 秒左右完成計算，而 Apriori 演算法則是隨著 minsup 的增加，itemset 減少，執行時間才逐漸下降。

實驗二、

比較在不同 minimum confidence 的狀況下，兩種演算法執行時間的差異。在此我們將 minimum support 固定為 10，用以單獨比較 minimum confidence 的影響。

minconf	0.2	0.4	0.6	0.8	1.0
Rules(條)	6308	1868	166	13	1



實驗結果：

在不同 minimum confidence 的情形下，可以發現 FP-growth 的效能同樣優於 Apriori 演算法。

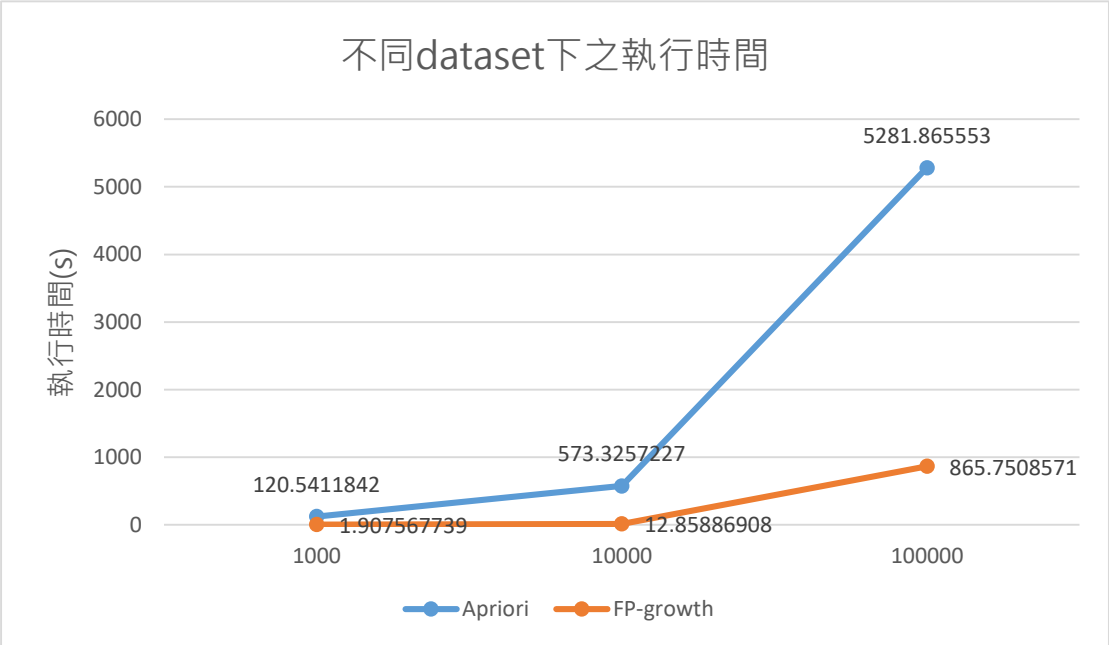
而 Apriori 演算法在 minconf 大於等於 0.4 後，執行時間大抵維持在 25 秒左右，可以推測在演算法中耗費較多時間的部分是在產生 frequent itemset 的階段，所以在 minsup 固定的情況下，minconf 對演算法效能相較之下影響較小。

實驗三、

比較在不同 dataset 大小的狀況下，兩種演算法執行時間的差異。在此我們將 minimum support 固定為 0.005*(transaction 數量)、minimum confidence 固定為 0.6，用以比較 dataset 大小的影響。

1. 比較三種 1000、10000、100000 筆 transaction 的執行效能

dataset	1000trans	10000trans	100000trans
Frequent itemsets(項)	16060	10925	10534



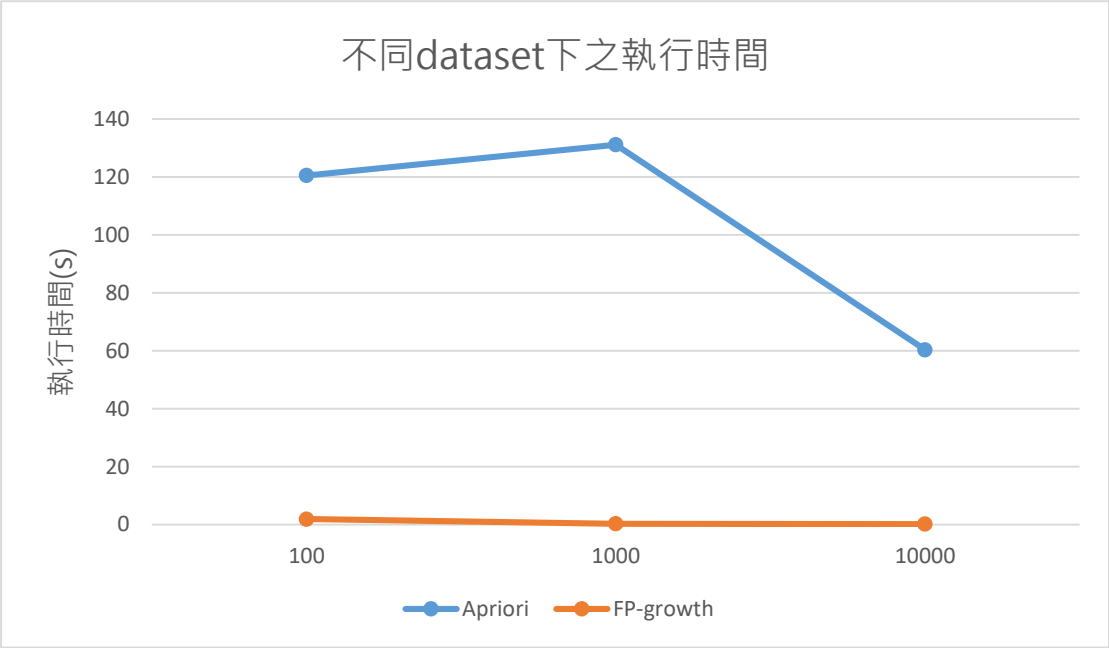
實驗結果：

在不同 dataset 的情形下，可以發現 FP-growth 的效能同樣優於 Apriori 演算法。

透過 transaction 數量增加，更可以明顯看出在資料量大的狀況下，Apriori 花費的時間增長太快，因此使用 FP-growth 相較於 Apriori 更加適合。

2. 比較三種 100、1000、10000 種 item 的執行效能

dataset	100items	1000items	10000items
Frequent itemsets(項)	16060	779	430



實驗結果：

在不同 dataset 的情形下，可以發現 FP-growth 的效能同樣優於 Apriori 演算法。

但在這裡比較特別的一點是，隨著 item 數量增加，Apriori 演算法執行時間先上升而後下降。理論上應該隨著 item 數量增加，而時間增加。在這裡之所以會有這樣的現象是因為，IBM 產生器的產生的資料，每筆交易平均只有 10 樣商品，因此在商品種類增加，而交易商品數沒有增加的情況下，會導致很多商品的 support 無法超越 minimum support，因此在商品數量增加時，執行時間可能像 FP-growth 保持穩定，也可能像 Apriori 演算法呈現起伏不定的情況。

實驗四、

比較使用產生器資料與 Kaggle 資料狀況下，對於演算法執行時間是否有所差異。在此我們將 minimum confidence 固定為 0.7，minimum support 設為 0.005*transaction 數量。

dataset	Kaggle (45000trans)	IBM (1000trans)	IBM (10000trans)	IBM (100000trans)
Apriori	22628.2223	120.5412	573.3257	5281.8656
FP-growth	76.4793	1.9076	12.8589	865.7509

實驗結果：

比較兩種資料集，我認為 FP-growth 在執行效能上符合預期，隨著 transaction 數量增加，執行時間也倍數成長。但是 Apriori 演算法甚至比 100000trans 的狀況下多出了約 4 倍的時間，會有這樣的情況應該是每次增加 frequent itemset 大小時，都需要重新掃描整個資料集，雖然他的 transaction 數量小於十萬筆，但 item 數量多達四千種，因此整體算下來資料量比起 100000trans 多出許多；FP-growth 之所以影響不大，是因為他只需掃描兩次資料集，所以整體時間不會增長太多。

- 結論：

透過各種數據的比較，FP-growth 確實是優於 Apriori 的一種演算法，雖然在實作過程比較困難，但由於不必產生 candidate itemset，因此在時間效能上能改善許多。

關於 Kaggle 的資料，由於這次抓到的是比較類似於產生器產生出來的資料集，因此觀察 FP-growth 演算法在效能上差異不大；但若是未來需要處理較多欄位的真實資料時，可能甚至需要先將欄位根據數值範圍拆成更多欄位的資料，那在處理上勢必會需要更多時間，這是在未來希望能改善之處。