Project 1

Association Analysis

N96094196 張維峻

工程科學系

目錄

Introduction [1]

實做 Apriori 及 FP Growth 兩個演算法並應用在至少兩個資料集上，從 kaggle 找到 Market Basket Optimization 資料集,包含 7501 份資料,並且使用 IBM Quest Data Generator 生成包含 8290 筆的資料集合。此外對不同資料集,不同條件設計下,測量運算時間,進行比較。

1. Kaggle - Market Basket Optimization

2. IBM Quest Data Generator - data.csv

```
Number of transactions in database = 1000000
Average transaction length = 10
Number of items = 10000000
Large Itemsets:
        Number of patterns = 500
        Average length of pattern = 3
        Correlation between consecutive patterns = 0.25
        Average confidence in a rule = 0.75
        Variation in the confidence = 0.1
```

File structure

```
+- algorithms
+- dataset
+- figures
+- ouput
+- script_exp_dataset
+- script_figures
+- Project1_Report_N96094196.pdf
```

- algorithms:讀取資料集格式等等
- dataset:資料集
- figures:產生的圖片
- ouput:產生的關聯法則,存成.csv
- script_exp_dataset:使用範例
- script_figures:產生圖片

Validation^2

表格一為我使用兩種演算法所建立的 association rules，其中 minimum support =2%, minimum Confidence = 20%。它們各自產生的規則及運算數值可驗證我所建立的程式碼之正確性。

Apriori				FP Growth		
rule	confidence	lift		rule	confidence	lift
{'burgers'} --> {'eggs'}	0.330275229	1.8375847		{'soup'} --> {'mineral water'}	0.471389646	2.288299
{'burgers'} --> {'french fries'}	0.252293578	1.4759765		{'cooking oil'} --> {'mineral water'}	0.403225806	1.957407
{'burgers'} --> {'mineral water'}	0.279816514	1.1743838		{'chicken'} --> {'mineral water'}	0.387990762	1.88345
{'burgers'} --> {'spaghetti'}	0.24617737	1.4137292		{'olive oil'} --> {'spaghetti'}	0.370860927	2.345242
{'cake'} --> {'mineral water'}	0.338815789	1.4220025		{'olive oil'} --> {'mineral water'}	0.439293598	2.132493
{'chicken'} --> {'mineral water'}	0.38	1.5948517		{'tomatoes'} --> {'spaghetti'}	0.325630252	2.059213
{'chocolate'} --> {'eggs'}	0.202603743	1.1272463		{'tomatoes'} --> {'mineral water'}	0.37605042	1.825487
{'french fries'} --> {'chocolate'}	0.20124805	1.2281207		{'low fat yogurt'} --> {'mineral water'}	0.342158859	1.660965
{'chocolate'} --> {'french fries'}	0.20992677	1.2281207		{'shrimp'} --> {'spaghetti'}	0.307240705	1.942922
{'frozen vegetables'} --> {'chocolate'}	0.240559441	1.4680194		{'shrimp'} --> {'mineral water'}	0.338551859	1.643456
{'ground beef'} --> {'chocolate'}	0.234735414	1.4324781		{'cake'} --> {'mineral water'}	0.36329588	1.763572
{'milk'} --> {'chocolate'}	0.247942387	1.513074		{'burgers'} --> {'french fries'}	0.258064516	1.947167
{'mineral water'} --> {'chocolate'}	0.221040851	1.3489067		{'burgers'} --> {'spaghetti'}	0.26655348	1.685625
{'chocolate'} --> {'mineral water'}	0.321399512	1.3489067		{'burgers'} --> {'mineral water'}	0.303904924	1.475267
{'spaghetti'} --> {'chocolate'}	0.225114855	1.3737684		{'burgers'} --> {'eggs'}	0.317487267	2.221226
{'chocolate'} --> {'spaghetti'}	0.239218877	1.3737684		{'pancakes'} --> {'eggs'}	0.243034056	1.700332
{'cooking oil'} --> {'mineral water'}	0.394255875	1.6546833		{'pancakes'} --> {'spaghetti'}	0.289473684	1.830567
{'eggs'} --> {'french fries'}	0.202522255	1.1848026		{'pancakes'} --> {'mineral water'}	0.390092879	1.893655
{'french fries'} --> {'eggs'}	0.212948518	1.1848026		{'frozen vegetables'} --> {'eggs'}	0.238095238	1.665778
{'frozen vegetables'} --> {'eggs'}	0.227972028	1.2683904		{'frozen vegetables'} --> {'chocolate'}	0.254464286	1.817602
{'ground beef'} --> {'eggs'}	0.203527815	1.1323877		{'frozen vegetables'} --> {'milk'}	0.261904762	2.237227
{'milk'} --> {'eggs'}	0.237654321	1.3222607		{'milk'} --> {'frozen vegetables'}	0.200455581	2.237227
{'eggs'} --> {'mineral water'}	0.283382789	1.1893514		{'frozen vegetables'} --> {'spaghetti'}	0.300595238	1.900897
{'mineral water'} --> {'eggs'}	0.213766088	1.1893514		{'frozen vegetables'} --> {'mineral water'}	0.388392857	1.885402
{'pancakes'} --> {'eggs'}	0.228611501	1.2719483		{'ground beef'} --> {'milk'}	0.233285917	1.992761
{'spaghetti'} --> {'eggs'}	0.209800919	1.16729		{'ground beef'} --> {'chocolate'}	0.244665718	1.747612
{'eggs'} --> {'spaghetti'}	0.203264095	1.16729		{'spaghetti'} --> {'ground beef'}	0.243676223	2.599675
{'green tea'} --> {'french fries'}	0.216161616	1.264596		{'ground beef'} --> {'spaghetti'}	0.411095306	2.599675
{'pancakes'} --> {'french fries'}	0.211781206	1.2389696		{'ground beef'} --> {'mineral water'}	0.432432432	2.099187
{'frozen smoothie'} --> {'mineral water'}	0.318565401	1.337012		{'green tea'} --> {'chocolate'}	0.201149425	1.436782
{'frozen vegetables'} --> {'milk'}	0.247552448	1.9101269		{'green tea'} --> {'eggs'}	0.208045977	1.455546
{'frozen vegetables'} --> {'mineral water'}	0.374825175	1.5731331		{'green tea'} --> {'spaghetti'}	0.228735632	1.446473
{'frozen vegetables'} --> {'spaghetti'}	0.292307692	1.6786429		{'green tea'} --> {'french fries'}	0.240229885	1.8126
{'green tea'} --> {'spaghetti'}	0.201010101	1.1543459		{'french fries'} --> {'green tea'}	0.210261569	1.8126
{'ground beef'} --> {'milk'}	0.223880597	1.7274737		{'green tea'} --> {'mineral water'}	0.256321839	1.244281

表一,兩種演算法對' Kaggle'資料產生的 association rules

Kaggle Dataset Compare[3]

Use Apriori Algorithm

Low support, Low confidence

MinSupport	MinConf
15	0.6

1-itemset = 115
2-itemset = 1225
3-itemset= 1154
4-itemset=170
5-itemset=2
Time taken= 203.25ms

Low support, High confidence

MinSupport	MinConf
15	0.9

1-itemset = 115
2-itemset = 1225
3-itemset= 1154
4-itemset=170
5-itemset=2
6-itemset=1
Time taken= 215.62ms

High support, Low confidence

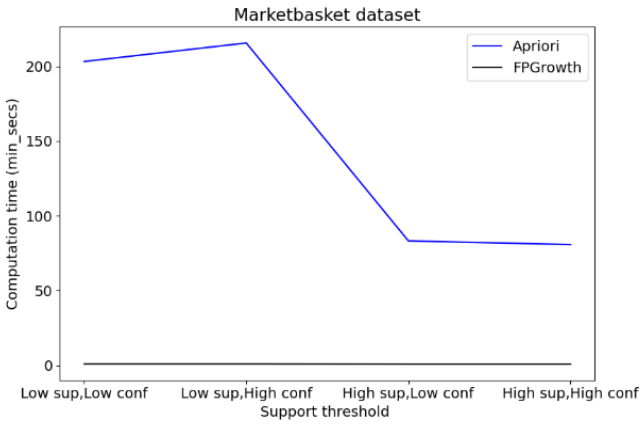
MinSupport	MinConf
30	0.6

1-itemset = 114
2-itemset = 595
3-itemset= 284
4-itemset=11
Time taken= 83.26ms

High support, High confidence

MinSupport	MinConf
30	0.9

1-itemset = 114
2-itemset = 595
3-itemset= 284
4-itemset= 11
Time taken= 80.79ms



Use FP Growth

Low support, Low confidence	
MinSupport	MinConf
15	0.6
1-itemset = 115	
2-itemset = 1219	
3-itemset= 1147	
4-itemset= 170	
5-itemset=2	
Time taken= 0.95ms	

Low support, High confidence	
MinSupport	MinConf
15	0.9
1-itemset = 115	
2-itemset = 1219	
3-itemset= 1147	
4-itemset= 170	
5-itemset=2	
Time taken= 0.95ms	

High support, Low confidence	
MinSupport	MinConf
30	0.6
1-itemset = 113	
2-itemset = 587	
3-itemset= 281	
4-itemset= 11	
Time taken= 0.84ms	

High support, High confidence	
MinSupport	MinConf
30	0.9
1-itemset = 113	
2-itemset = 587	
3-itemset= 281	
4-itemset= 11	
Time taken= 0.84ms	

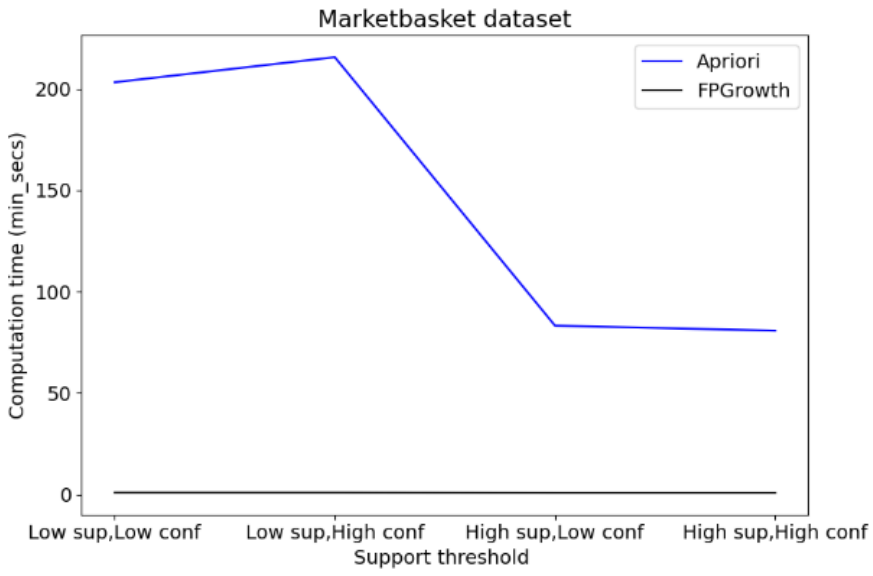


圖 一、使用 Kaggle dataset 進行條件差別實驗之運算時間結果。

IBM Dataset Compare[4]

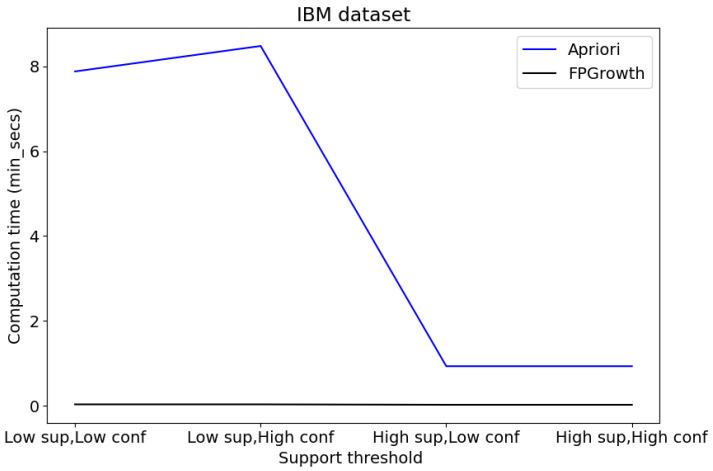
Use Apriori Algorithm

Low support, Low confidence	
MinSupport	MinConf
15	0.6
1-itemset = 144	
2-itemset = 143	
3-itemset= 87	
4-itemset=34	
5-itemset=8	
6-itemset=1	
Time taken= 7.88ms	

Low support, High confidence	
MinSupport	MinConf
15	0.9
1-itemset = 144	
2-itemset = 143	
3-itemset= 87	
4-itemset=34	
5-itemset=8	
6-itemset=1	
Time taken= 8.48ms	

High support, Low confidence	
MinSupport	MinConf
30	0.6
1-itemset = 17	
2-itemset = 9	
3-itemset= 3	
Time taken= 0.93ms	

High support, Low confidence	
MinSupport	MinConf
30	0.9
1-itemset = 17	
2-itemset = 9	
3-itemset= 3	
Time taken= 0.93ms	



Use FP Growth

Low support, Low confidence

MinSupport	MinConf
15	0.6

1-itemset = 143
2-itemset = 123
3-itemset= 50
4-itemset= 16
5-itemset=5
6-itemset=1
Time taken= 0.03m

low support, High confidence

MinSupport	MinConf
15	0.9

1-itemset = 143
2-itemset = 123
3-itemset= 50
4-itemset= 16
5-itemset=5
6-itemset=1
Time taken= 0.03ms

High support, Low confidence

MinSupport	MinConf
30	0.6

1-itemset = 16
2-itemset = 7
3-itemset= 2
Time taken= 0.02ms

High support, High confidence

MinSupport	MinConf
30	0.9

1-itemset = 16
2-itemset = 7
3-itemset= 2
Time taken= 0.02ms

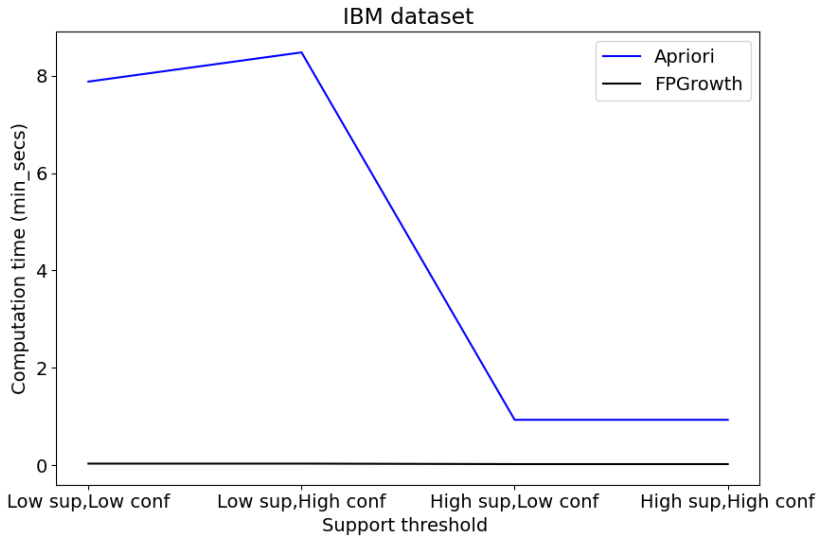


圖 一、使用 IBM dataset 進行條件差別實驗之運算時間結果。

Apriori hashtree[5]

DataSet	MinSupport	max_leaf_count	max_child_count
IBM	15	3	5

1-itemset = 144
2-itemset = 131
3-itemset=76
4-itemset=29
5-itemset=7
6-itemset=1
Time taken= 0.225ms

Summary[6]

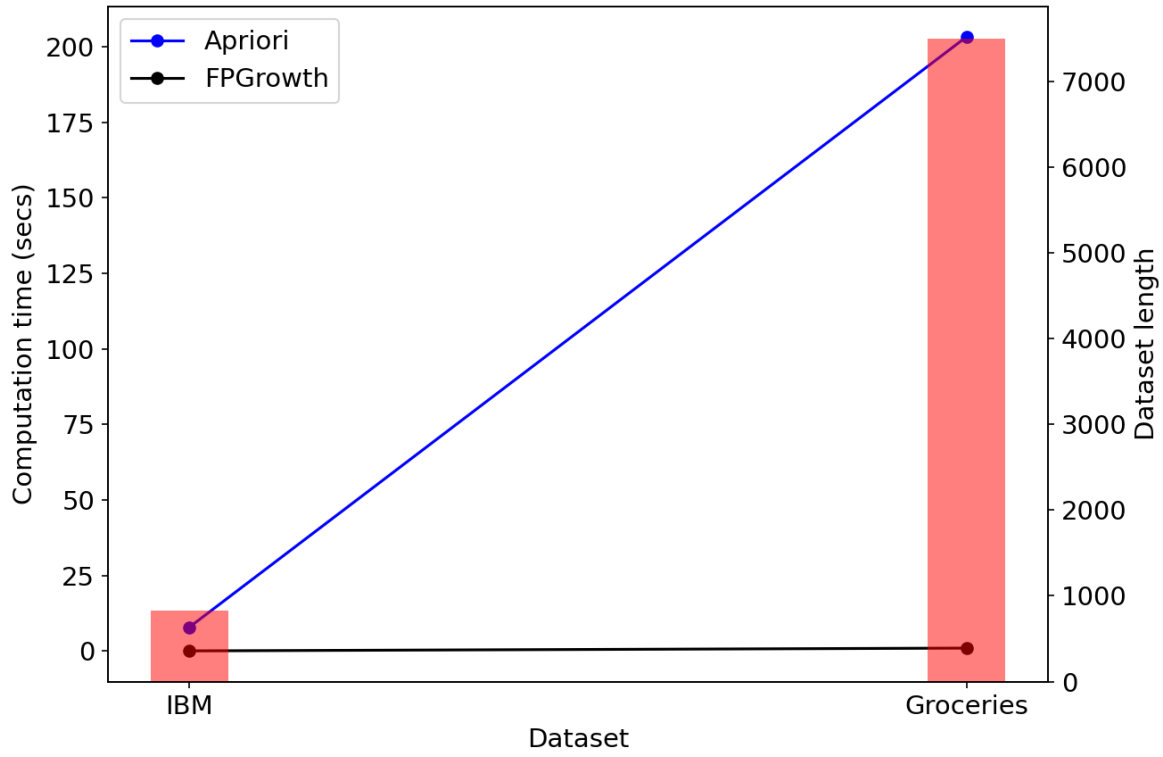
Apriori 是使用 candidate_set 搜尋 frequent patterns,進行到每一層時都必須重新產生 candidate_set,並且花費大量資源在掃描資料庫;雖然 apriori 的缺點很明顯,但 apriori 的出現,改良了傳統完整掃描的方法,加快運算速度,也為 datamining 提供了一種新的想法。

FP-Growth 是由 apriori 改良而成,採用樹狀結構,先構建出 FPtree,再從 FPtree 中挖掘 frequent itemset,減少掃描次數並且,fp-growth 只需要做兩次掃描,而非 apriori 在每一層都生成 candidate 來進行掃描。

	掃描資料次數	是否產生候選集
Apriori	N+1	有
FP Growth	2	無

	Ls, Lc	Ls, Hc	Hs, Lc	Hs, Hc
Support	15	15	30	30
Confidence	60%	90%	60%	90%
apriori_kaggle(sec)	203.25ms	215.62ms	83.26ms	80.79ms
FP-growth_kaggle(sec)	0.95ms	0.95ms	0.84ms	0.84ms

表二、求不同 min_sup，min_conf 下，花費運算時間。



圖三、不同資料集長度、不同演算法的花費時間