Report

Dataset

IBM Request，分別使用兩組資料

**data.ntrans\_0.01.nitems\_0.01.tlen\_5**

這筆資料包含10種items，8個transaction

Number of items =40

Number of transaction=8

**data.ntrans\_0.1.nitems\_0.01.tlen\_5**

**這筆資料包含100種item，10個transaction**

**Number of items =100**

**Number of transaction=10**

**Implement**

**FP-growth**

1. **首先把dataset轉成frozendata set**
2. **建立FP-tree的class，裡面包含tree node 、parent、children**
3. **建立FP-tree，把小於minSup的item刪除**
4. **以sort後的frequent 1 item建立FPtree**
5. **在mineingFPTree，把getPrefixPath函式丟入，建立conditional tree**
6. **最後用getttRules和GenerateRule 兩個函式找出frequent pattern**

**Apriori**

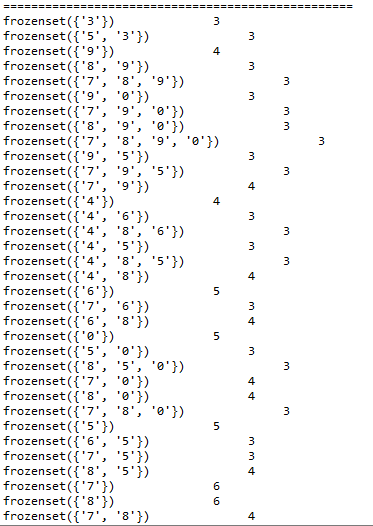
1. **建立一個class存放1 item set**
2. **測試是否滿足apriori property**
3. **產生item set，並作排序，尋找frequent item**
4. **最後丟入generate\_big\_rules 找出frequent pattern**

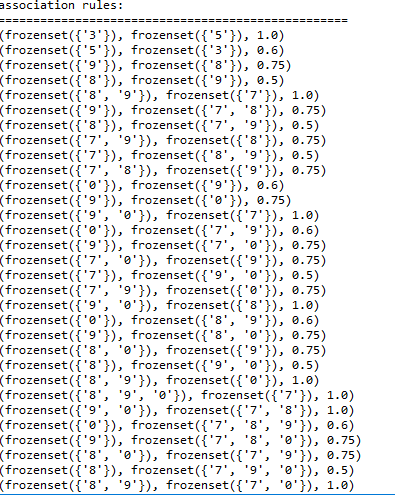
**Data Result**

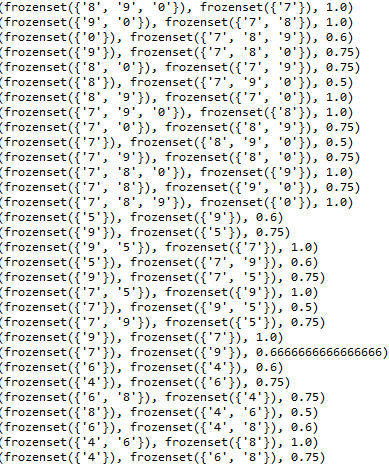
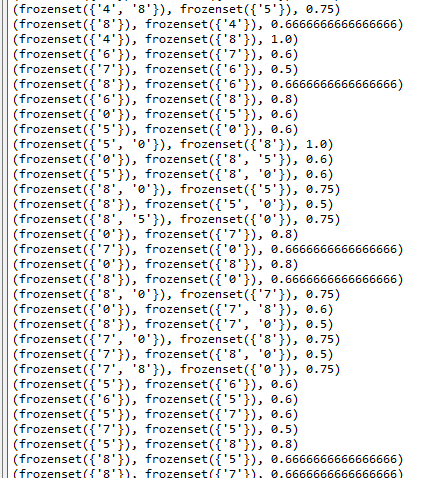
**這裡統一minsupport=3，min confidence=0.5**

**首先用FP-growth 進行第一個data的測試**

**data.ntrans\_0.01.nitems\_0.01.tlen\_5**







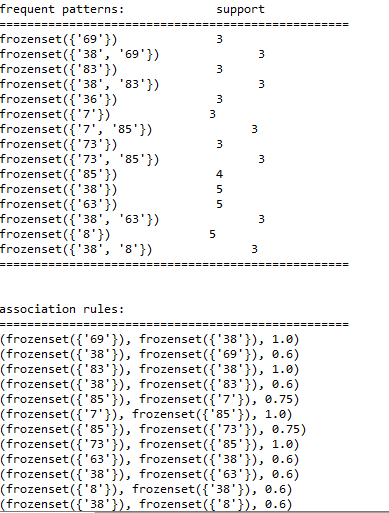
平均花費的時間如下

It takes 6 milliseconds to find the above patterns

It takes 4 milliseconds to find the above patterns

大概時間都在這個區間，接下來是下個data

**data.ntrans\_0.1.nitems\_0.01.tlen\_5**



而平均時間如下

It takes 3 milliseconds to find the above patterns

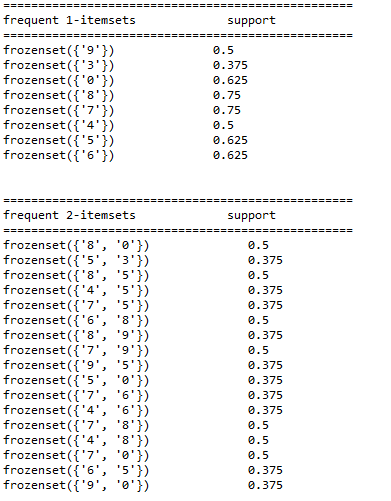
It takes 1 milliseconds to find the above patterns

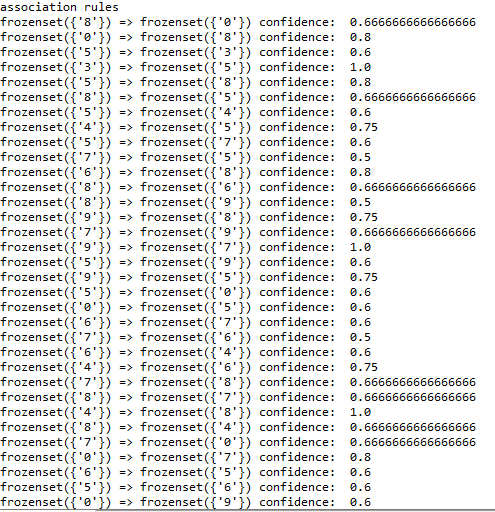
討論

1. 第二筆資料，因為資料分布比較稀疏的關係，所以在尋找frequent pattern較為快速
2. 第一筆資料雖然item數目少，但相對集中，符合min sup 的item較多，找起frequent pattern較花時間

Apriori Data測試結果

只用第一筆資料當作對比





剩餘的部分大都相同

討論

這裡討論一個點，就是速度的問題。

平均按了幾次Apriori，得到的運算時間大概都在

It takes 16 milliseconds to find the above patterns

It takes 14 milliseconds to find the above patterns

範圍內，很顯然的FP-growth的速度明顯快上很多，而且這只是小資料的測試，如果更大筆的資料差距只會更明顯。

結論

這裡比較兩個演算法的一些差異

Apriori

實作容易，但因為需要對database掃描多次，所以時間和空間複雜度較高

FP-Growth

實作困難，但只需要對database進行一次掃描，所以時間和空間複雜度略低

整體而言，FP-Growth在速度的表現上比Apriori優秀許多，變成取代之的關鍵