資料探勘

專案作業四

組員:黃凱祥、吳俊園、賀鈞嗣、吳鈞達

2023 年 1 月

摘要

此次實驗是針對大量的交易資料進行關聯規則的分析，前處理中預先剔除退貨以及註銷的交易(數量為零或負值)，對於不同的資料分別設定信心度及支持度。實驗中使用了 Apriori 演算法以及 FP-Growth 演算法，通過修改不同的參數設定，紀錄規則數量和執行時間並進行比對。

關鍵字:python、Apriori、FP-Growth

第一章、緒論

1.1 動機

透過對課堂上所教授的 Apriori 和 FP-Growth 演算法，使用作業提供的交易資料集，以了解學習的成果和累積實作之經驗。

1.2 目的

在交易資料集中進行挖掘資料關聯性和頻繁項，主要觀看使用 Apriori 和 FP-Growth 演算法兩個執行的速度，以及支持度、信心度和最小支持度多寡對資料關聯規則分析影響。

第二章、資料集

2.1 資料集

2.1.1 交易資料集

ITEM_ID	ITEM_NO	PRODUCT_T	CUST_ID	TRX_DATE	INVOICE_NO	QUANTITY
14605751	X79A-GD45-PLU	OTHERS	999999999	2016/07/31	5153	1
15141363	Z170A GAMING	OTHERS	999999999	2016/07/31	5153	1
15280990	BX80662I76700S	CPU / MPU	999999999	2016/07/31	5153	2
15288976	BX80662I56500S	CPU / MPU	999999999	2016/07/31	5153	4
14782210	BX80646I54460S	CPU / MPU	999999999	2016/07/31	5153	5
15288975	BX80662I56400S	CPU / MPU	999999999	2016/07/31	5153	14
15189014	BX80662I36100S	CPU / MPU	999999999	2016/07/31	5153	16
134855	PBSS4350T,215	DISCRETE	30786	2016/07/01	216070000	9000
70448	PMBT2222A,215	DISCRETE	30786	2016/07/01	216070000	9000
70448	PMBT2222A,215	DISCRETE	30786	2016/07/01	216070000	54000
134855	PBSS4350T,215	DISCRETE	30786	2016/07/01	216070000	60000
3250360	PTVS5V0S1UR,1	DISCRETE	2065	2016/07/01	216070001	9000

Figure 1 交易部分資料

(a) 資料筆數：128838。

(b) 資料屬性欄位數：7。

(c) 資料型態：顧客交易資料。

(d) 缺失值：無。

其屬性的資訊為

(a) ITEM_ID : 商品編號。

(b) ITEM_NO：商品條碼。

(c) PRODUCT_TYPE：產品類型。

(d) CUST_ID：顧客編號。

(e) TRX_DATE：交易日期。

(f) INVOICE_NO: 發票編號

(g) QUANTITY : 購買數量

第三章、方法

3.1 前置處理

在交易資料集中，將購買數量為零或是負值的交易剔除。

3.2 實驗設計

由於 scikit-learn 沒有包含內建交易資料集，必須讀檔，且設定欄位，將交易資料集以發票順序做排序 invoice-sorted，排序完成後再從程式碼中相同發票編號放在同一個陣列作為一比交易，分別使用 Apriori、FP-Growth 演算法，使用 Apriori 會得到每個 item 或多個 item 組合的支持度找出 frequentitemsets，調整 min_support 參數，也就是最小支持度，即在所有交易中支持度大於最小支持度，才可符合 frequenset 的要求，並且使用 time 函式個別得出執行速度，其中 FP-Growth 跟上述 Apriori 一樣的方法。

交易資料集中，商品種類太多種，min_support 值要設到低於 0.008 才會有頻繁項集。

3.3 實驗結果

首先將交易資料集中使 apriori 演算法，調整 min_support 至 0.001，產生大於 min_support 的 frequent itemset 和支持度和執行速度。

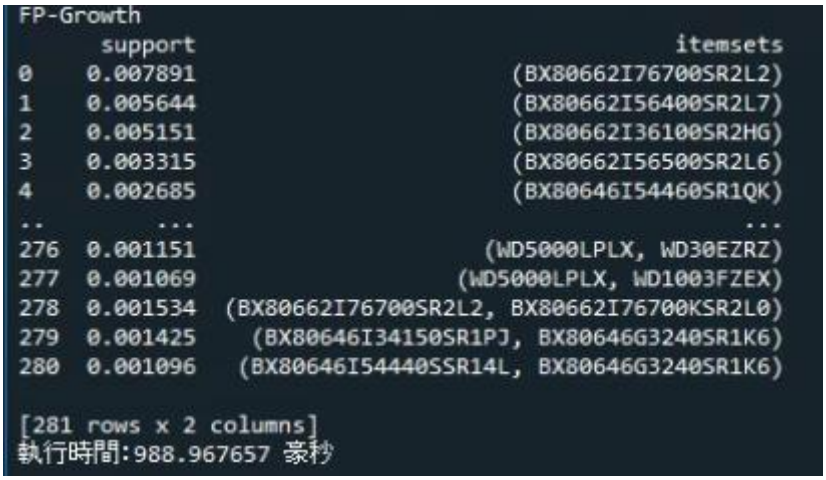
```
Apriori:
      support                                itemsets
0    0.001151                            (1N4148,133)
1    0.001151                          (1PS79SB30,115)
2    0.003151                             (2N7002,215)
3    0.002274                           (2N7002BK,215)
4    0.001014                           (2N7002CK,215)
..        ...                                     ...
276  0.001315   (BAV99,215, BAS316,115, BAV70,215)
277  0.001233   (BAV99,215, BAW56,215, BAS316,115)
278  0.001096  (BAV99,215, BAS316,115, PDZ15B,115)
279  0.001041  (BAV99,215, BAT54A,215, BAT54C,215)
280  0.001041   (BAT54S,215, BAV99,215, BAV70,215)

[281 rows x 2 columns]
執行時間:17097.979307 毫秒
```

再產生商品的關聯規則

```
                 antecedents      consequents  antecedent support  \
0                  (SC6531DA)        (HS8292U)            0.001562
1             (PBSS4240T,215)  (PBSS5240T,215)            0.001397
2                  (SR2351C)        (SR2319A)            0.001096
3                  (SR2319A)        (SR2351C)            0.001123
4  (2N7002BK,215, BAS316,115)       (BAV99,215)            0.001151
5     (BAW56,215, BAS316,115)       (BAV99,215)            0.001315
```

最後使用 FP-Growth 演算法，調整 min_support 至 0.001，產生大於 min_support 的 frequent itemset 和支持度和執行速度。



FP-Growth

	support	itemsets
0	0.007891	(BX80662I76700SR2L2)
1	0.005644	(BX80662I56400SR2L7)
2	0.005151	(BX80662I36100SR2HG)
3	0.003315	(BX80662I56500SR2L6)
4	0.002685	(BX80646I54460SR1QK)
..	...	...
276	0.001151	(WD5000LPLX, WD30EZRZ)
277	0.001069	(WD5000LPLX, WD1003FZEX)
278	0.001534	(BX80662I76700SR2L2, BX80662I76700KSR2L0)
279	0.001425	(BX80646I34150SR1PJ, BX80646G3240SR1K6)
280	0.001096	(BX80646I54440SSR14L, BX80646G3240SR1K6)

[281 rows x 2 columns]
執行時間:988.967657 毫秒

第四章、結論

在交易資料集中，需要預處理資料，將數量為零或負值的剔除，且要把發票順序排序，將相同的放在同一個陣列，才能開始關聯規則分析，並且商品的種類太多必須將 min_support 值低於 0.008 才會產生 frequent item，最小支持度越大，發現產生的 frequent item 越少，各 item 的支持度也會降低，最後在使用 time 函數，發現 FP-Growth 比 Apriori 執行速度更快，因為 FP-Growth 指掃描兩次。

第五章、參考文獻

[1] *Apriori Algorithm(Mlxtend library)*
https://artsdatascience.wordpress.com/2019/12/10/apriori-algorithm/