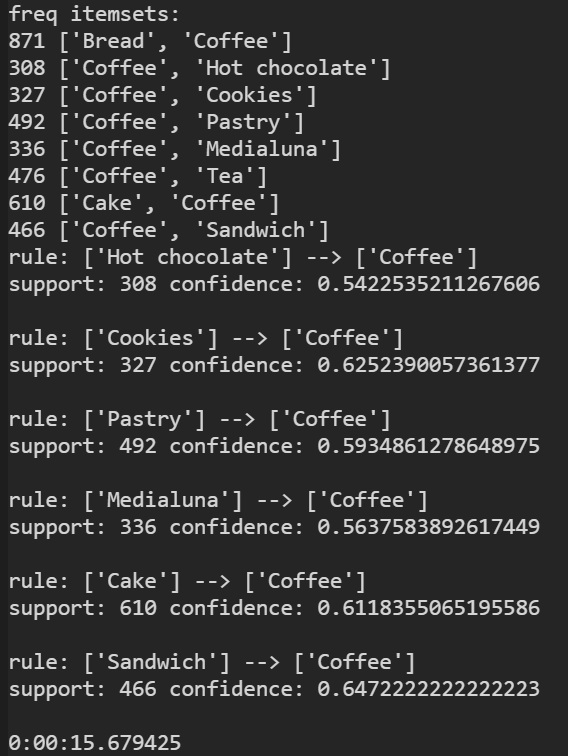
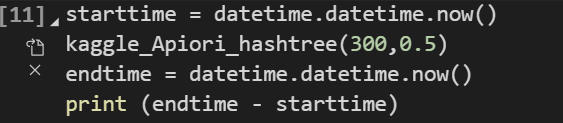
使用python實作Apiori暴力法、Apiori使用hash tree搜尋support值以及使用FP-Growth方法直接找出frequent itemset，並且由frequent itemset生成關聯規則。

其中暴力法為逐筆transaction循序搜尋下去，如果比對candidate成功便會將其support值加一，這是裡面最緩慢的方法

Hash tree部分，其實作的hash function為h(x) = x % 5，其中x為itemset當中所有數字相加，若item為英文字串，則將字串中所有字母轉為ascii代碼後加總。每個節點都可以有5棵子樹，因此搜尋candidate時便可以透過hash function的配對排除其他不必要的搜尋。

FP-Growth部分，為三種方法裡面最為迅速的方法，不論minimum support設定多低，都有不錯的表現，迅速地找出Frequent itemset。

程式執行 : (以kaggle資料集，使用hashtree實作apiori方法為例，並設定minimum support為300 instance、minimum confidence為0.5)

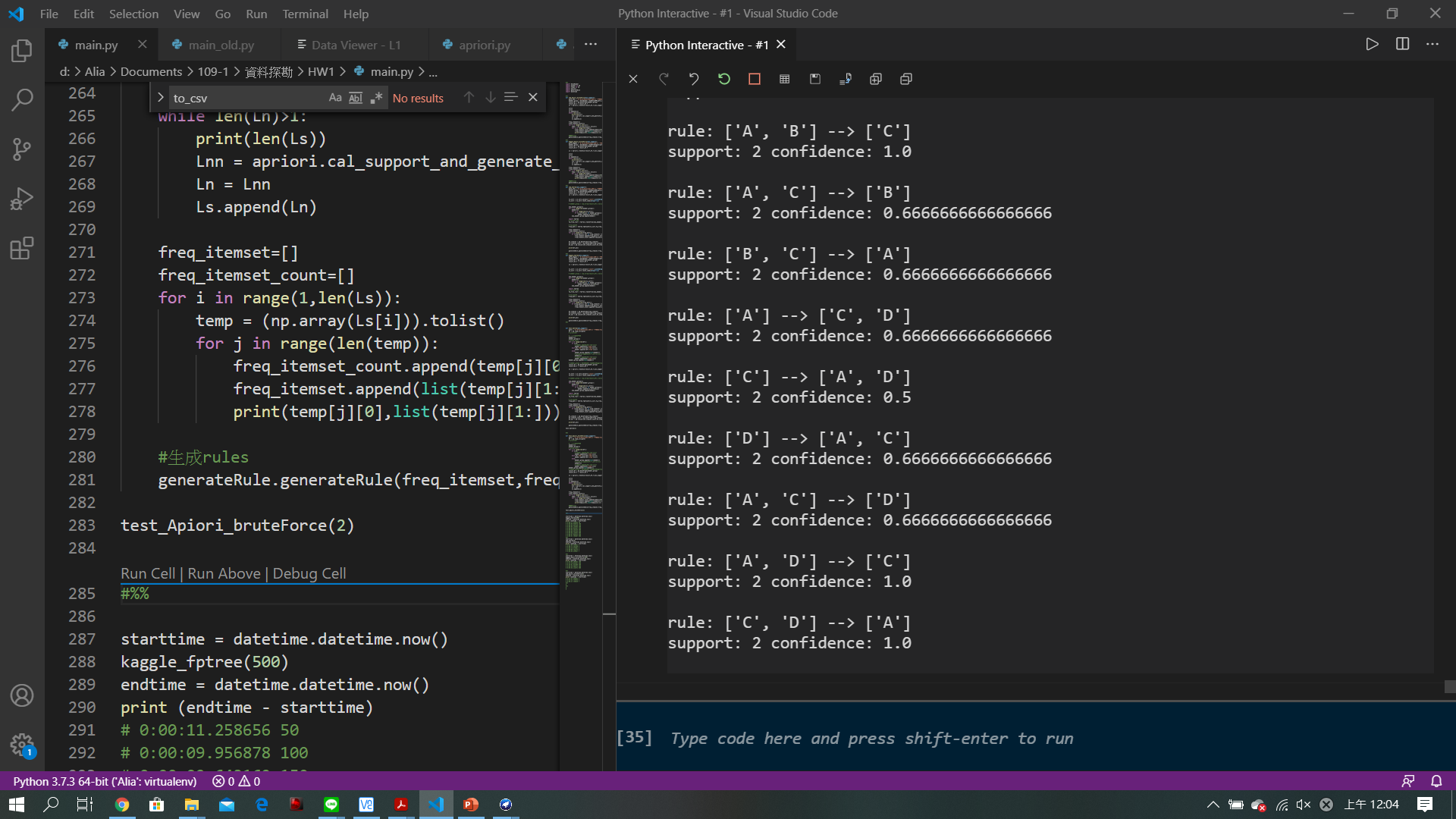
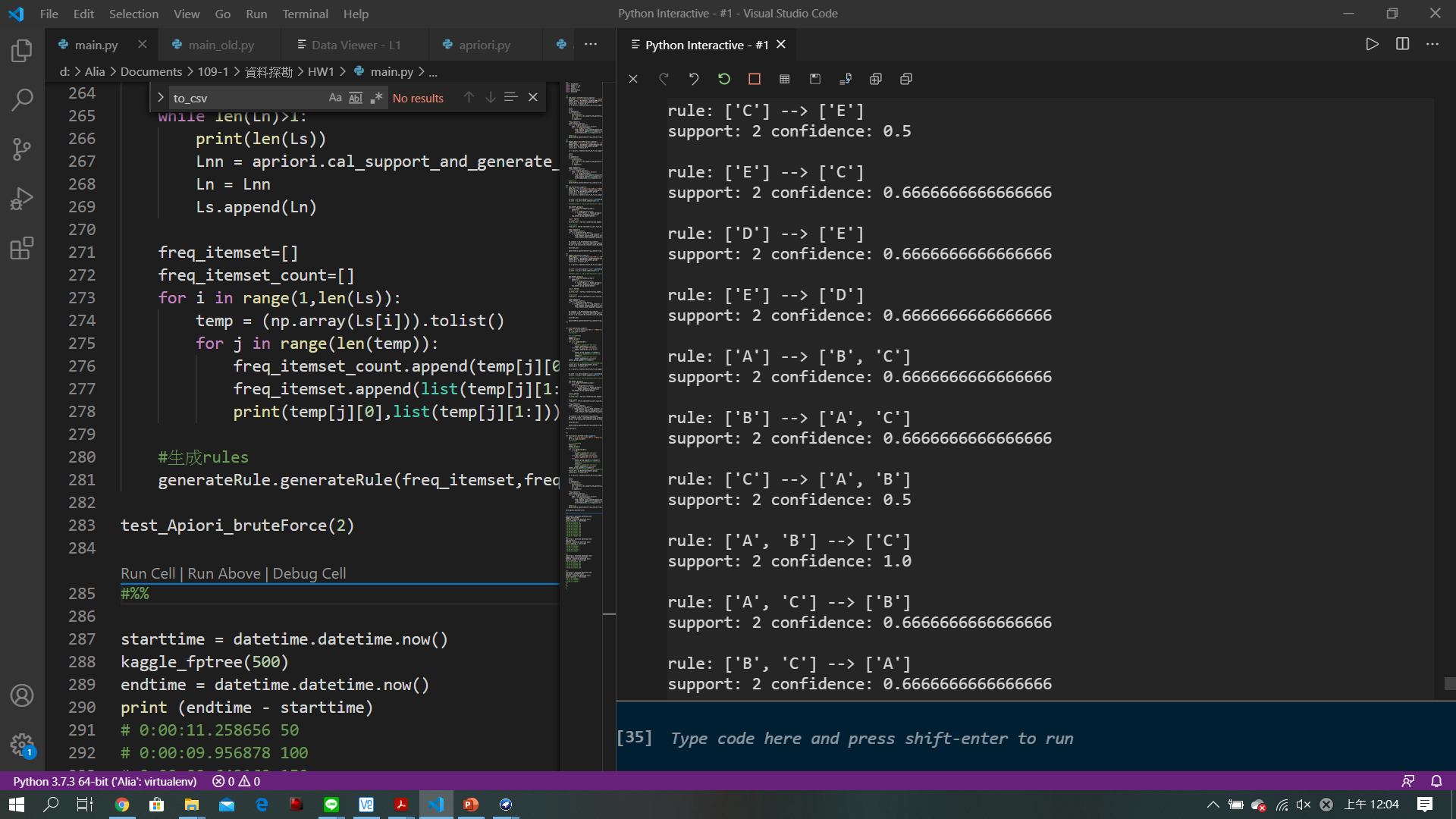
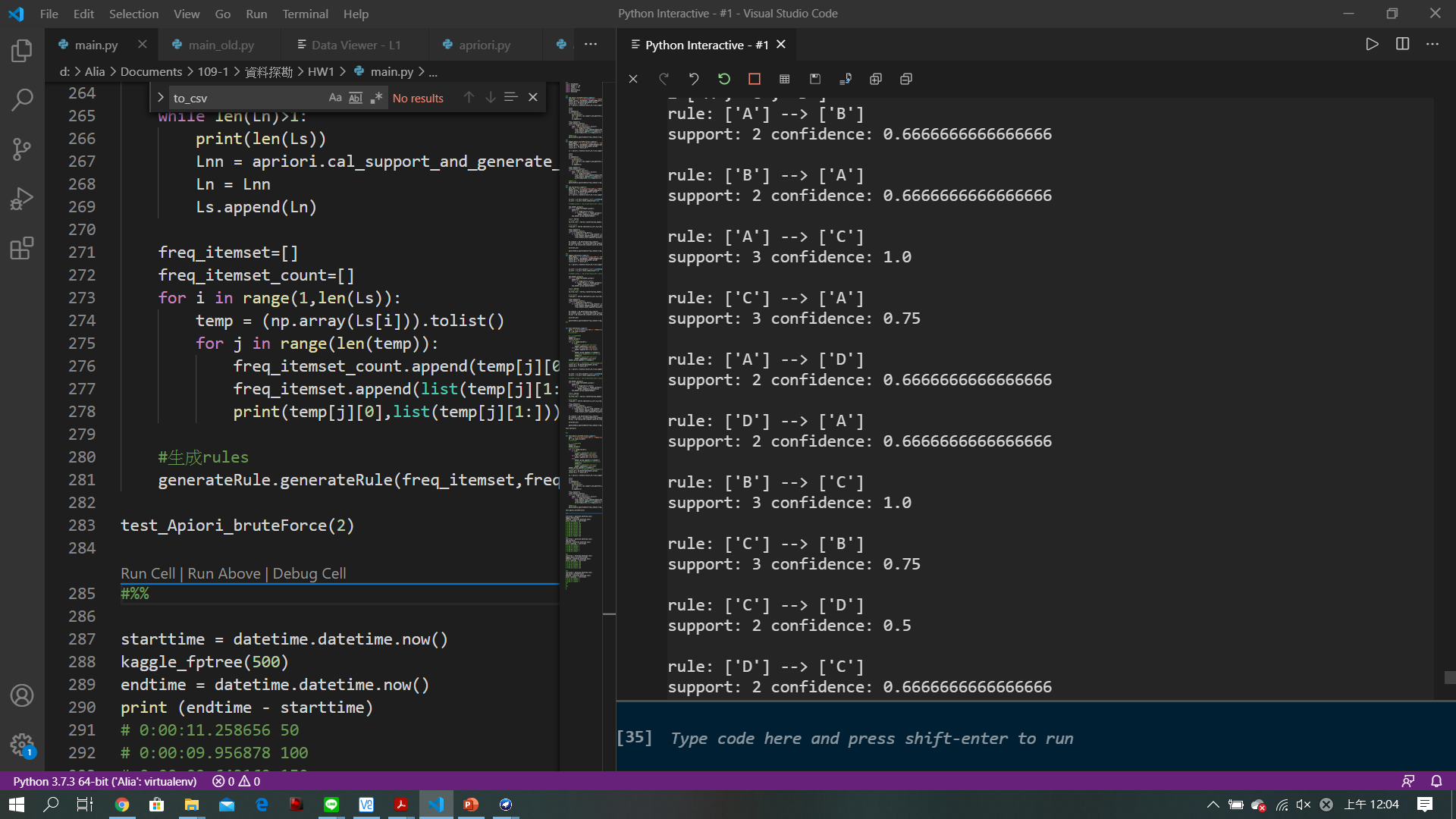
輸出 : (包含frequent itemset , rules, 執行時間)

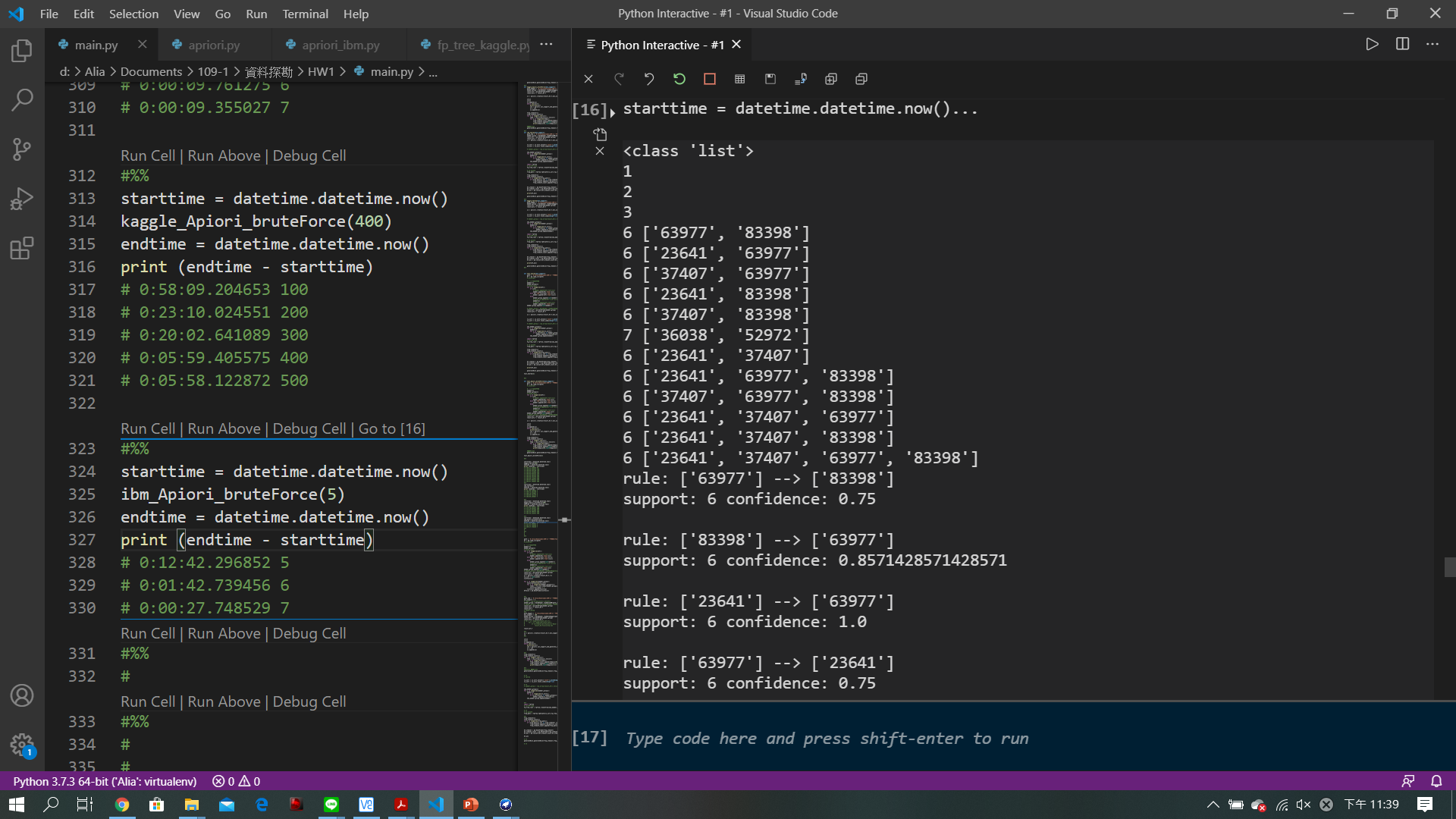
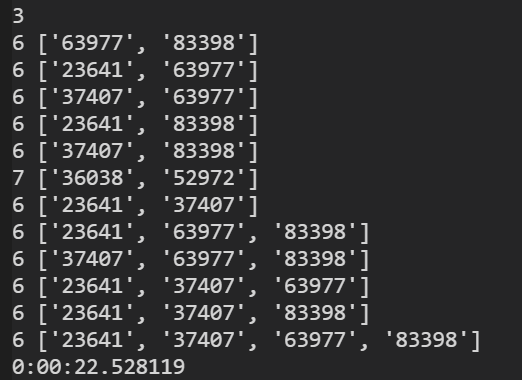
驗證

由於在大量數據的association rules並不會依照固定順序排列，再比對上有些難度，所以就採用上課投影片比較簡單的transaction，限制minimum support為0.4（2 instance）、minimum confidence為0.5，使用Weka產生association rules，再比對這次作業程式輸出的association rules，確認結果一致。

一張含有 文字 的圖片

自動產生的描述



一張含有 螢幕擷取畫面, 電子用品, 電腦 的圖片

自動產生的描述驗證2 frequent itemset by ibm(暴力、hash、fp)

由於從frequent itemset產出關連規則是使用相同函式(於驗證1已驗證過)，且規則數量較多在比對上有困難，因此只比對frequent itemset是否相同。

使用ibm資料集，限制minimum support的instance為5，確認3種方法所產出之frequent itemset相同。

比較

Ibm

Number of items 10877

Number of transactions 1069

Max item number 29

暴力

# 0:12:42.296852 5

# 0:01:42.739456 6

# 0:00:27.748529 7

Fptree

# 0:01:00.330683 3

# 0:00:15.282360 4

# 0:00:10.475901 5

# 0:00:09.761275 6

# 0:00:09.355027 7

kaggle

Number of items 21293

Number of transactions 9684(9531)

Max item number 12

暴力

# 0:58:09.204653 100

# 0:23:10.024551 200

# 0:20:02.641089 300

# 0:05:59.405575 400

# 0:05:58.122872 500

Fptree

# 0:00:11.258656 50

# 0:00:09.956878 100

# 0:00:09.643169 150

# 0:00:09.152948 200

# 0:00:08.925191 250

# 0:00:08.382345 300

# 0:00:08.332754 350

# 0:00:08.134953 400

# 0:00:07.955423 450

# 0:00:07.796567 500