Aarjavi Dharaiya

Data mining - CS 634

ard22@njit.edu

Mid-term Project report

Association rules generation by Apriori and Brute Force algorithms.

Table of Contents

1. Screen shots for Database 1, minimum support = 15, minimum confidence = 15

```
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$ python MidTermProj.py
Choose your Store:
1 for Marshals
2 for Deli
3 for Sports Center
4 for OfficeDepo
5 for SpeedCars
1
Enter minimum support (in %)15
Enter minimum confidence (in %)15


Transactions


['BOOK_COVER', 'JOURNAL', 'ATLAS']

['BOOK_COVER', 'PENCIL', 'SHARPNER', 'SOCKS', 'SHOES']

['LAPTOP', 'BAG', 'SCALE', 'NOTEBOOK']

['ERASER', 'ATLAS', 'PENCIL']

['BOOK_COVER', 'JOURNAL', 'SHARPNER', 'SOCKS']

['LAPTOP', 'MONITOR_SCREEN']

['NOTEBOOK', 'SCALE', 'ATLAS']

['BOOK_COVER', 'JOURNAL', 'SOCKS', 'SHOES']

['LAPTOP', 'BAG']

['JOURNAL', 'SHARPNER']

['BOOK_COVER', 'PENCIL', 'SHARPNER']

['LAPTOP', 'BAG', 'LUNCH_BOX']

['LAPTOP', 'BOOK_COVER', 'ATLAS']

['BOOK_COVER', 'LAPTOP']

['ERASER', 'PENCIL']

['LAPTOP', 'SHARPNER']

['BOOK_COVER', 'LAPTOP', 'BAG']

['BOOK_COVER', 'PENCIL']

['BOOK_COVER', 'ERASER']

['BAG', 'SCALE']

User Chose MinSupport : 15.0  and MinConfidence:  15.0

Association rules generated by Apriori Algorithm:

 ['BAG'] -----> ['LAPTOP']    Sup:  20.0%   Conf:  80.0%
```

```
['BAG', 'SCALE']

User Chose MinSupport : 15.0  and MinConfidence:  15.0

Association rules generated by Apriori Algorithm:

 ['BAG'] -----> ['LAPTOP']    Sup:  20.0%   Conf:  80.0%

 ['LAPTOP'] -----> ['BAG']    Sup:  20.0%   Conf:  50.0%

 ['BOOK_COVER'] -----> ['JOURNAL']    Sup:  15.0%   Conf:  30.0%

 ['JOURNAL'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  75.0%

 ['BOOK_COVER'] -----> ['LAPTOP']    Sup:  15.0%   Conf:  30.0%

 ['LAPTOP'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  37.5%

 ['BOOK_COVER'] -----> ['PENCIL']    Sup:  15.0%   Conf:  30.0%

 ['PENCIL'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  60.0%

 ['BOOK_COVER'] -----> ['SHARPNER']    Sup:  15.0%   Conf:  30.0%

 ['SHARPNER'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  60.0%

 ['BOOK_COVER'] -----> ['SOCKS']    Sup:  15.0%   Conf:  30.0%

 ['SOCKS'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  100.0%

Association rules generated by Brute Force Algorithm:

 ['BAG'] -----> ['LAPTOP']    Sup:  20.0%   Conf:  80.0%

 ['LAPTOP'] -----> ['BAG']    Sup:  20.0%   Conf:  50.0%

 ['BOOK_COVER'] -----> ['JOURNAL']    Sup:  15.0%   Conf:  30.0%

 ['JOURNAL'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  75.0%

 ['BOOK_COVER'] -----> ['LAPTOP']    Sup:  15.0%   Conf:  30.0%

 ['LAPTOP'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  37.5%

 ['BOOK_COVER'] -----> ['PENCIL']    Sup:  15.0%   Conf:  30.0%

 ['PENCIL'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  60.0%

 ['BOOK_COVER'] -----> ['SHARPNER']    Sup:  15.0%   Conf:  30.0%

 ['SHARPNER'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  60.0%

 ['BOOK_COVER'] -----> ['SOCKS']    Sup:  15.0%   Conf:  30.0%

 ['SOCKS'] -----> ['BOOK_COVER']    Sup:  15.0%   Conf:  100.0%

Total Association rules generated by Apriori are :  12  and by Brute Force are:  12

Time taken for generating Association rules from Apriori is : 1.03 ms and from Brute Force is : 209.35 ms
Time improvement by Apriori Alg :  208.32 ms
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$
```

2. Screen shots for Database 2, minimum support = 15, minimum confidence = 20

```
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$ python MidTermProj.py
Choose your Store:
1 for Marshals
2 for Deli
3 for Sports Center
4 for OfficeDepo
5 for SpeedCars
2
Enter minimum support (in %)15
Enter minimum confidence (in %)20


Transactions


['SUGAR', 'TEA', 'COFFEE']

['BAGEL', 'FALAFEL', 'FANTA', 'YOGURT']

['TEA', 'GRANOLA']

['BAGEL', 'YOGURT']

['TEA', 'CHEESE', 'BAGEL']

['BAGEL', 'COFFEE', 'YOGURT']

['MILK', 'FANTA', 'TEA']

['TEA']

['BAGEL', 'CHEESE', 'TEA', 'YOGURT', 'COFFEE']

['SUGAR', 'FANTA', 'CHEESE']

['MILK', 'SUGAR']

['MILK']

['TEA', 'GRANOLA', 'FALAFEL']

['FALAFEL', 'FANTA', 'MILK']

['BAGEL', 'YOGURT', 'CHEESE', 'COFFEE']

['TEA', 'FANTA']

['SUGAR', 'FANTA']

['CHEESE', 'GRANOLA', 'FANTA', 'MILK', 'TEA']

['BAGEL', 'TEA', 'COFFEE']

['FALAFEL', 'FANTA']


User Chose MinSupport : 15.0  and MinConfidence:  20.0

Association rules generated by Apriori Algorithm:

 ['BAGEL'] -----> ['CHEESE']    Sup:  15.0%   Conf:  42.86%
```

```
User Chose MinSupport : 15.0  and MinConfidence:  20.0
Association rules generated by Apriori Algorithm:
 ['BAGEL'] -----> ['CHEESE']    Sup:  15.0%   Conf:  42.86%
 ['CHEESE'] -----> ['BAGEL']    Sup:  15.0%   Conf:  60.0%
 ['BAGEL'] -----> ['COFFEE']    Sup:  20.0%   Conf:  57.14%
 ['COFFEE'] -----> ['BAGEL']    Sup:  20.0%   Conf:  80.0%
 ['BAGEL'] -----> ['TEA']    Sup:  15.0%   Conf:  42.86%
 ['TEA'] -----> ['BAGEL']    Sup:  15.0%   Conf:  30.0%
 ['BAGEL'] -----> ['YOGURT']    Sup:  25.0%   Conf:  71.43%
 ['YOGURT'] -----> ['BAGEL']    Sup:  25.0%   Conf:  100.0%
 ['CHEESE'] -----> ['TEA']    Sup:  15.0%   Conf:  60.0%
 ['TEA'] -----> ['CHEESE']    Sup:  15.0%   Conf:  30.0%
 ['COFFEE'] -----> ['TEA']    Sup:  15.0%   Conf:  60.0%
 ['TEA'] -----> ['COFFEE']    Sup:  15.0%   Conf:  30.0%
 ['COFFEE'] -----> ['YOGURT']    Sup:  15.0%   Conf:  60.0%
 ['YOGURT'] -----> ['COFFEE']    Sup:  15.0%   Conf:  60.0%
 ['FALAFEL'] -----> ['FANTA']    Sup:  15.0%   Conf:  75.0%
 ['FANTA'] -----> ['FALAFEL']    Sup:  15.0%   Conf:  37.5%
 ['FANTA'] -----> ['MILK']    Sup:  15.0%   Conf:  37.5%
 ['MILK'] -----> ['FANTA']    Sup:  15.0%   Conf:  60.0%
 ['FANTA'] -----> ['TEA']    Sup:  15.0%   Conf:  37.5%
 ['TEA'] -----> ['FANTA']    Sup:  15.0%   Conf:  30.0%
 ['GRANOLA'] -----> ['TEA']    Sup:  15.0%   Conf:  100.0%
 ['TEA'] -----> ['GRANOLA']    Sup:  15.0%   Conf:  30.0%
 ['BAGEL'] -----> ['COFFEE', 'YOGURT']    Sup:  15.0%   Conf:  42.86%
 ['COFFEE'] -----> ['BAGEL', 'YOGURT']    Sup:  15.0%   Conf:  60.0%
 ['YOGURT'] -----> ['BAGEL', 'COFFEE']    Sup:  15.0%   Conf:  60.0%
 ['BAGEL', 'COFFEE'] -----> ['YOGURT']    Sup:  15.0%   Conf:  75.0%
 ['BAGEL', 'YOGURT'] -----> ['COFFEE']    Sup:  15.0%   Conf:  60.0%
 ['COFFEE', 'YOGURT'] -----> ['BAGEL']    Sup:  15.0%   Conf:  100.0%

Association rules generated by Brute Force Algorithm:
```

```
Association rules generated by Brute Force Algorithm:
 ['BAGEL'] -----> ['CHEESE']    Sup:  15.0%   Conf:  42.86%
 ['CHEESE'] -----> ['BAGEL']    Sup:  15.0%   Conf:  60.0%
 ['BAGEL'] -----> ['COFFEE']    Sup:  20.0%   Conf:  57.14%
 ['COFFEE'] -----> ['BAGEL']    Sup:  20.0%   Conf:  80.0%
 ['BAGEL'] -----> ['TEA']    Sup:  15.0%   Conf:  42.86%
 ['TEA'] -----> ['BAGEL']    Sup:  15.0%   Conf:  30.0%
 ['BAGEL'] -----> ['YOGURT']    Sup:  25.0%   Conf:  71.43%
 ['YOGURT'] -----> ['BAGEL']    Sup:  25.0%   Conf:  100.0%
 ['CHEESE'] -----> ['TEA']    Sup:  15.0%   Conf:  60.0%
 ['TEA'] -----> ['CHEESE']    Sup:  15.0%   Conf:  30.0%
 ['COFFEE'] -----> ['TEA']    Sup:  15.0%   Conf:  60.0%
 ['TEA'] -----> ['COFFEE']    Sup:  15.0%   Conf:  30.0%
 ['COFFEE'] -----> ['YOGURT']    Sup:  15.0%   Conf:  60.0%
 ['YOGURT'] -----> ['COFFEE']    Sup:  15.0%   Conf:  60.0%
 ['FALAFEL'] -----> ['FANTA']    Sup:  15.0%   Conf:  75.0%
 ['FANTA'] -----> ['FALAFEL']    Sup:  15.0%   Conf:  37.5%
 ['FANTA'] -----> ['MILK']    Sup:  15.0%   Conf:  37.5%
 ['MILK'] -----> ['FANTA']    Sup:  15.0%   Conf:  60.0%
 ['FANTA'] -----> ['TEA']    Sup:  15.0%   Conf:  37.5%
 ['TEA'] -----> ['FANTA']    Sup:  15.0%   Conf:  30.0%
 ['GRANOLA'] -----> ['TEA']    Sup:  15.0%   Conf:  100.0%
 ['TEA'] -----> ['GRANOLA']    Sup:  15.0%   Conf:  30.0%
 ['BAGEL'] -----> ['COFFEE', 'YOGURT']    Sup:  15.0%   Conf:  42.86%
 ['COFFEE'] -----> ['BAGEL', 'YOGURT']    Sup:  15.0%   Conf:  60.0%
 ['YOGURT'] -----> ['BAGEL', 'COFFEE']    Sup:  15.0%   Conf:  60.0%
 ['BAGEL', 'COFFEE'] -----> ['YOGURT']    Sup:  15.0%   Conf:  75.0%
 ['BAGEL', 'YOGURT'] -----> ['COFFEE']    Sup:  15.0%   Conf:  60.0%
 ['COFFEE', 'YOGURT'] -----> ['BAGEL']    Sup:  15.0%   Conf:  100.0%
Total Association rules generated by Apriori are :  28  and by Brute Force are:  28

Time taken for generating Association rules from Apriori is : 2.39 ms and from Brute Force is : 22.41 ms
Time improvement by Apriori Alg :  20.02 ms
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$
```

3. Screen shots for Database 3, minimum support = 14, minimum confidence = 8

```
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$ python MidTermPro
Choose your Store:
1 for Marshals
2 for Deli
3 for Sports Center
4 for OfficeDepo
5 for SpeedCars
3
Enter minimum support (in %)14
Enter minimum confidence (in %)8


Transactions


['CARROM', 'CHESS']

['BALL', 'SKATES', 'SOCCER_BALL']

['BAG', 'GOLF_STICK', 'SOCCER_BALL', 'SHOES']

['SKATES']

['SHOES', 'SOCKS']

['SPIKES', 'SKATES', 'SHOES', 'SOCKS', 'BAG']

['GOLF_STICK']

['CHESS', 'CARROM']

['SHOES', 'BALL', 'SOCKS', 'SKATES']

['SOCCER_BALL', 'CARROM', 'GOLF_STICK']

['SPIKES']

['SHOES', 'SOCKS', 'SOCCER_BALL']

['CARROM', 'SKATES', 'CHESS']

['CHESS', 'SKATES', 'BALL', 'CARROM']

['BAG', 'SHOES']

['BALL']

['SOCCER_BALL', 'SHOES', 'SPIKES']

['SHOES']

['CHESS', 'CARROM']

['BALL', 'BAG']


User Chose MinSupport : 14.0  and MinConfidence:  8.0

Association rules generated by Apriori Algorithm:

 ['BAG'] -----> ['SHOES']    Sup:  15.0%   Conf:  75.0%
```

```
['BALL']

['SOCCER_BALL', 'SHOES', 'SPIKES']

['SHOES']

['CHESS', 'CARROM']

['BALL', 'BAG']

User Chose MinSupport : 14.0  and MinConfidence:  8.0

Association rules generated by Apriori Algorithm:

 ['BAG'] -----> ['SHOES']    Sup:  15.0%   Conf:  75.0%

 ['SHOES'] -----> ['BAG']    Sup:  15.0%   Conf:  37.5%

 ['BALL'] -----> ['SKATES']    Sup:  15.0%   Conf:  60.0%

 ['SKATES'] -----> ['BALL']    Sup:  15.0%   Conf:  50.0%

 ['CARROM'] -----> ['CHESS']    Sup:  25.0%   Conf:  83.33%

 ['CHESS'] -----> ['CARROM']    Sup:  25.0%   Conf:  100.0%

 ['SHOES'] -----> ['SOCCER_BALL']    Sup:  15.0%   Conf:  37.5%

 ['SOCCER_BALL'] -----> ['SHOES']    Sup:  15.0%   Conf:  60.0%

 ['SHOES'] -----> ['SOCKS']    Sup:  20.0%   Conf:  50.0%

 ['SOCKS'] -----> ['SHOES']    Sup:  20.0%   Conf:  100.0%

Association rules generated by Brute Force Algorithm:

 ['BAG'] -----> ['SHOES']    Sup:  15.0%   Conf:  75.0%

 ['SHOES'] -----> ['BAG']    Sup:  15.0%   Conf:  37.5%

 ['BALL'] -----> ['SKATES']    Sup:  15.0%   Conf:  60.0%

 ['SKATES'] -----> ['BALL']    Sup:  15.0%   Conf:  50.0%

 ['CARROM'] -----> ['CHESS']    Sup:  25.0%   Conf:  83.33%

 ['CHESS'] -----> ['CARROM']    Sup:  25.0%   Conf:  100.0%

 ['SHOES'] -----> ['SOCCER_BALL']    Sup:  15.0%   Conf:  37.5%

 ['SOCCER_BALL'] -----> ['SHOES']    Sup:  15.0%   Conf:  60.0%

 ['SHOES'] -----> ['SOCKS']    Sup:  20.0%   Conf:  50.0%

 ['SOCKS'] -----> ['SHOES']    Sup:  20.0%   Conf:  100.0%

Total Association rules generated by Apriori are :  10  and by Brute Force are:  10

Time taken for generating Association rules from Apriori is : 1.6 ms and from Brute Force is : 15.78 ms
Time improvement by Apriori Alg :  14.18 ms
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$
```

4. Screen shots for Database 4, minimum support = 16, minimum confidence = 18

```
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$ python MidTermProj.py
Choose your Store:
1 for Marshals
2 for Deli
3 for Sports Center
4 for OfficeDepo
5 for SpeedCars
4
Enter minimum support (in %)16
Enter minimum confidence (in %)18


Transactions

['SPEAKERS', 'MONITORS', 'LAPTOPS']

['SPEAKERS', 'MONITORS']

['PENS', 'STICKY_NOTES']

['PENS', 'SHARPIES']

['STICKY_NOTES', 'PENS', 'SHARPIES']

['MOUSE', 'DOCK_STATION', 'KEYBOARD']

['SHARPIES', 'DONGLES']

['SPEAKERS', 'MONITORS', 'LAPTOPS', 'DOCK_STATION']

['SHARPIES', 'MOUSE']

['LAPTOPS', 'MOUSE', 'SPEAKERS', 'KEYBOARD']

['SPEAKERS', 'DOCK_STATION']

['SPEAKERS', 'DOCK_STATION', 'KEYBOARD']

['DONGLES', 'PENS', 'STICKY_NOTES']

['MOUSE', 'DOCK_STATION']

['STICKY_NOTES', 'SHARPIES']

['SPEAKERS', 'MONITORS', 'LAPTOPS', 'MOUSE', 'KEYBOARD']

['DONGLES', '']

['MONITORS', 'LAPTOPS']

['DOCK_STATION', 'KEYBOARD']

['STICKY_NOTES', 'PENS', 'SHARPIES', 'MOUSE', 'DOCK_STATION']

User Chose MinSupport : 16.0  and MinConfidence:  18.0

Association rules generated by Apriori Algorithm:

 ['LAPTOPS'] -----> ['MONITORS']    Sup:  20.0%   Conf:  80.0%
```

```
['SPEAKERS', 'DOCK_STATION', 'KEYBOARD']

['DONGLES', 'PENS', 'STICKY_NOTES']

['MOUSE', 'DOCK_STATION']

['STICKY_NOTES', 'SHARPIES']

['SPEAKERS', 'MONITORS', 'LAPTOPS', 'MOUSE', 'KEYBOARD']

['DONGLES', '']

['MONITORS', 'LAPTOPS']

['DOCK_STATION', 'KEYBOARD']

['STICKY_NOTES', 'PENS', 'SHARPIES', 'MOUSE', 'DOCK_STATION']


User Chose MinSupport : 16.0  and MinConfidence:  18.0

Association rules generated by Apriori Algorithm:

 ['LAPTOPS'] -----> ['MONITORS']    Sup:  20.0%   Conf:  80.0%

 ['MONITORS'] -----> ['LAPTOPS']    Sup:  20.0%   Conf:  80.0%

 ['LAPTOPS'] -----> ['SPEAKERS']    Sup:  20.0%   Conf:  80.0%

 ['SPEAKERS'] -----> ['LAPTOPS']    Sup:  20.0%   Conf:  57.14%

 ['MONITORS'] -----> ['SPEAKERS']    Sup:  20.0%   Conf:  80.0%

 ['SPEAKERS'] -----> ['MONITORS']    Sup:  20.0%   Conf:  57.14%

 ['PENS'] -----> ['STICKY_NOTES']    Sup:  20.0%   Conf:  80.0%

 ['STICKY_NOTES'] -----> ['PENS']    Sup:  20.0%   Conf:  80.0%


Association rules generated by Brute Force Algorithm:

 ['LAPTOPS'] -----> ['MONITORS']    Sup:  20.0%   Conf:  80.0%

 ['MONITORS'] -----> ['LAPTOPS']    Sup:  20.0%   Conf:  80.0%

 ['LAPTOPS'] -----> ['SPEAKERS']    Sup:  20.0%   Conf:  80.0%

 ['SPEAKERS'] -----> ['LAPTOPS']    Sup:  20.0%   Conf:  57.14%

 ['MONITORS'] -----> ['SPEAKERS']    Sup:  20.0%   Conf:  80.0%

 ['SPEAKERS'] -----> ['MONITORS']    Sup:  20.0%   Conf:  57.14%

 ['PENS'] -----> ['STICKY_NOTES']    Sup:  20.0%   Conf:  80.0%

 ['STICKY_NOTES'] -----> ['PENS']    Sup:  20.0%   Conf:  80.0%

Total Association rules generated by Apriori are :  8  and by Brute Force are:  8

Time taken for generating Association rules from Apriori is : 1.34 ms and from Brute Force is : 30.23 ms
Time improvement by Apriori Alg :  28.89 ms
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$
```

5. Screen shots for Database 5, minimum support = 12, minimum confidence = 16

```
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$ python MidTermProj.py
Choose your Store:
1 for Marshals
2 for Deli
3 for Sports Center
4 for OfficeDepo
5 for SpeedCars
5
Enter minimum support (in %)12
Enter minimum confidence (in %)16


Transactions

['USB_CABLE', 'CHARGER', 'KEY_FOB']

['USB_CABLE', 'KEY_FOB', 'STEREO']

['JACK_LIFT', 'AIR_PUMP', 'CAR_VACCUM']

['USB_CABLE', 'CHARGER', 'CAR_VACCUM', 'WIPERS']

['USB_CABLE', 'CHARGER', 'KEY_FOB', 'BLIND_SPOT_MIRROR']

['JACK_LIFT', 'BREAK_LIGHTS', '']

['USB_CABLE', 'CHARGER']

['STEREO', 'BLIND_SPOT_MIRROR']

['AIR_PUMP', 'JACK_LIFT', 'BREAK_LIGHTS']

['USB_CABLE', 'CHARGER', 'BREAK_LIGHTS', 'WIPERS']

['STEREO', 'WIPERS', '']

['USB_CABLE', 'CHARGER', 'KEY_FOB', 'JACK_LIFT', 'WIPERS']

['KEY_FOB', 'AIR_PUMP', 'INDICATORS']

['BREAK_LIGHTS', 'CAR_VACCUM']

['USB_CABLE', 'CHARGER', 'BLIND_SPOT_MIRROR']

['WIPERS', 'AIR_PUMP']

['BLIND_SPOT_MIRROR', 'KEY_FOB']

['STEREO', 'BREAK_LIGHTS', '']

['CHARGER', 'JACK_LIFT', '']

User Chose MinSupport : 12.0  and MinConfidence:  16.0

Association rules generated by Apriori Algorithm:

 ['CHARGER'] -----> ['KEY_FOB']    Sup:  15.79%   Conf:  37.5%

 ['KEY_FOB'] -----> ['CHARGER']    Sup:  15.79%   Conf:  50.0%
```

```
['CHARGER'] -----> ['KEY_FOB']    Sup:  15.79%   Conf:  37.5%
['KEY_FOB'] -----> ['CHARGER']    Sup:  15.79%   Conf:  50.0%
['CHARGER'] -----> ['USB_CABLE']    Sup:  36.84%   Conf:  87.5%
['USB_CABLE'] -----> ['CHARGER']    Sup:  36.84%   Conf:  87.5%
['CHARGER'] -----> ['WIPERS']    Sup:  15.79%   Conf:  37.5%
['WIPERS'] -----> ['CHARGER']    Sup:  15.79%   Conf:  60.0%
['KEY_FOB'] -----> ['USB_CABLE']    Sup:  21.05%   Conf:  66.67%
['USB_CABLE'] -----> ['KEY_FOB']    Sup:  21.05%   Conf:  50.0%
['USB_CABLE'] -----> ['WIPERS']    Sup:  15.79%   Conf:  37.5%
['WIPERS'] -----> ['USB_CABLE']    Sup:  15.79%   Conf:  60.0%
['CHARGER'] -----> ['KEY_FOB', 'USB_CABLE']    Sup:  15.79%   Conf:  37.5%
['KEY_FOB'] -----> ['CHARGER', 'USB_CABLE']    Sup:  15.79%   Conf:  50.0%
['USB_CABLE'] -----> ['CHARGER', 'KEY_FOB']    Sup:  15.79%   Conf:  37.5%
['CHARGER', 'KEY_FOB'] -----> ['USB_CABLE']    Sup:  15.79%   Conf:  100.0%
['CHARGER', 'USB_CABLE'] -----> ['KEY_FOB']    Sup:  15.79%   Conf:  42.86%
['KEY_FOB', 'USB_CABLE'] -----> ['CHARGER']    Sup:  15.79%   Conf:  75.0%
['CHARGER'] -----> ['USB_CABLE', 'WIPERS']    Sup:  15.79%   Conf:  37.5%
['USB_CABLE'] -----> ['CHARGER', 'WIPERS']    Sup:  15.79%   Conf:  37.5%
['WIPERS'] -----> ['CHARGER', 'USB_CABLE']    Sup:  15.79%   Conf:  60.0%
['CHARGER', 'USB_CABLE'] -----> ['WIPERS']    Sup:  15.79%   Conf:  42.86%
['CHARGER', 'WIPERS'] -----> ['USB_CABLE']    Sup:  15.79%   Conf:  100.0%
['USB_CABLE', 'WIPERS'] -----> ['CHARGER']    Sup:  15.79%   Conf:  100.0%

Association rules generated by Brute Force Algorithm:
['CHARGER'] -----> ['KEY_FOB']    Sup:  15.79%   Conf:  37.5%
['KEY_FOB'] -----> ['CHARGER']    Sup:  15.79%   Conf:  50.0%
['CHARGER'] -----> ['USB_CABLE']    Sup:  36.84%   Conf:  87.5%
['USB_CABLE'] -----> ['CHARGER']    Sup:  36.84%   Conf:  87.5%
['CHARGER'] -----> ['WIPERS']    Sup:  15.79%   Conf:  37.5%
['WIPERS'] -----> ['CHARGER']    Sup:  15.79%   Conf:  60.0%
['KEY_FOB'] -----> ['USB_CABLE']    Sup:  21.05%   Conf:  66.67%
['USB_CABLE'] -----> ['KEY_FOB']    Sup:  21.05%   Conf:  50.0%
```

```
['USB_CABLE'] -----> ['CHARGER', 'WIPERS']    Sup:  15.79%   Conf:  37.5%
['WIPERS'] -----> ['CHARGER', 'USB_CABLE']    Sup:  15.79%   Conf:  60.0%
['CHARGER', 'USB_CABLE'] -----> ['WIPERS']    Sup:  15.79%   Conf:  42.86%
['CHARGER', 'WIPERS'] -----> ['USB_CABLE']    Sup:  15.79%   Conf:  100.0%
['USB_CABLE', 'WIPERS'] -----> ['CHARGER']    Sup:  15.79%   Conf:  100.0%

Association rules generated by Brute Force Algorithm:
['CHARGER'] -----> ['KEY_FOB']    Sup:  15.79%   Conf:  37.5%
['KEY_FOB'] -----> ['CHARGER']    Sup:  15.79%   Conf:  50.0%
['CHARGER'] -----> ['USB_CABLE']    Sup:  36.84%   Conf:  87.5%
['USB_CABLE'] -----> ['CHARGER']    Sup:  36.84%   Conf:  87.5%
['CHARGER'] -----> ['WIPERS']    Sup:  15.79%   Conf:  37.5%
['WIPERS'] -----> ['CHARGER']    Sup:  15.79%   Conf:  60.0%
['KEY_FOB'] -----> ['USB_CABLE']    Sup:  21.05%   Conf:  66.67%
['USB_CABLE'] -----> ['KEY_FOB']    Sup:  21.05%   Conf:  50.0%
['USB_CABLE'] -----> ['WIPERS']    Sup:  15.79%   Conf:  37.5%
['WIPERS'] -----> ['USB_CABLE']    Sup:  15.79%   Conf:  60.0%
['CHARGER'] -----> ['KEY_FOB', 'USB_CABLE']    Sup:  15.79%   Conf:  37.5%
['KEY_FOB'] -----> ['CHARGER', 'USB_CABLE']    Sup:  15.79%   Conf:  50.0%
['USB_CABLE'] -----> ['CHARGER', 'KEY_FOB']    Sup:  15.79%   Conf:  37.5%
['CHARGER', 'KEY_FOB'] -----> ['USB_CABLE']    Sup:  15.79%   Conf:  100.0%
['CHARGER', 'USB_CABLE'] -----> ['KEY_FOB']    Sup:  15.79%   Conf:  42.86%
['KEY_FOB', 'USB_CABLE'] -----> ['CHARGER']    Sup:  15.79%   Conf:  75.0%
['CHARGER'] -----> ['USB_CABLE', 'WIPERS']    Sup:  15.79%   Conf:  37.5%
['USB_CABLE'] -----> ['CHARGER', 'WIPERS']    Sup:  15.79%   Conf:  37.5%
['WIPERS'] -----> ['CHARGER', 'USB_CABLE']    Sup:  15.79%   Conf:  60.0%
['CHARGER', 'USB_CABLE'] -----> ['WIPERS']    Sup:  15.79%   Conf:  42.86%
['CHARGER', 'WIPERS'] -----> ['USB_CABLE']    Sup:  15.79%   Conf:  100.0%
['USB_CABLE', 'WIPERS'] -----> ['CHARGER']    Sup:  15.79%   Conf:  100.0%
Total Association rules generated by Apriori are :  22  and by Brute Force are:  22

Time taken for generating Association rules from Apriori is : 2.52 ms and from Brute Force is : 55.09 ms
Time improvement by Apriori Alg :  52.57 ms
(base) Aarjavis-MacBook-Pro:MidTermProj aarjavi$
```

6. Conclusion

We generated 5 databases of different item sets and found the results to be in the favour of Apriori algorithm. We ran the datasets for different values of min support and min confidence, and for every run, the association rules generated by the Apriori and Brute Force are the same. But, Apriori is much faster as shown by the screenshots of the results attached

7. Program Code

```
import time

def isSuperSetofNFST(intersection,disjoint) :

    if(len(intersection) == 0):
        return 0
    this_hashcode = getUniqueHashCode(disjoint)
    if this_hashcode in NFST_hash:
        return 1
    for elem in intersection:
        temp = []
        temp = [elem] + disjoint
        temp.sort()
        this_hashcode = getUniqueHashCode(temp)
        if this_hashcode in NFST_hash:
            return 1
    return 0

def getIntersectingElements(list1,list2) :
    intersectList = []
    for elemOfList1 in list1:
        if elemOfList1 in list2:
            intersectList.append(elemOfList1)
    intersectList.sort()
    return intersectList


def getDisjointElements(list1,list2) :
    disjointList = []
    for elemOfList1 in list1:
        if elemOfList1 not in list2:
            disjointList.append(elemOfList1)
    for elemOfList2 in list2:
        if elemOfList2 not in list1:
            disjointList.append(elemOfList2)
    disjointList.sort()
    return disjointList


def getUniqueHashCode(list) :
    NewHashCode = 0
    iter1 = 0
    for i in list:
        this_single_item_code = HashCode[i]
```

```
        NewHashCode += this_single_item_code * pow(1000,iter1)
        iter1 += 1
    return NewHashCode

def getEventOccurence(event,alg=1):
    Occur = 0
    for this_trans in transactions:
        flag = 1
        for elem in event:
            if elem not in this_trans:
                flag = 0
        if flag:
            Occur += 1

    if(Occur > 0):
        event.sort()
        this_hashcode = getUniqueHashCode(event)
        if(alg==1):
            OccurenceCount[this_hashcode] = Occur
        if(alg==2):
            OccurenceCountBF[this_hashcode] = Occur
    return Occur

def issubset(list1,list2):
    return set(list1).issubset(list2)

#Below function to generate FST by brute force alg for a set of K elements
def getFSTfromAllComb(SampleList):
    FSTbf = []
    for event in SampleList:
        eventCount = getEventOccurence(event,2)
        if(eventCount >= minoccur):
            FSTbf.append(event)
    return FSTbf

#Generate all possible combinations of length k
def GenCombin(lst, lst_tmp, start,end, index,k) :

    if (index == k) :
        lst1_tmp = lst_tmp[0:k]

        CombinationList.append(lst1_tmp)
        return

    i = start
    while(i<=end and end-i+1 >= k-index):
        lst_tmp[index] = lst[i]
        GenCombin(lst, lst_tmp, i+1, end, index+1, k)
```

```
        i += 1


#Geneate FST by Apriori Alg
def getFrequentItemSets_Apr(FSTk,newK,minoccur):

    N = len(FSTk)
    listOfListsTemp = []
    listOfLists = []
    intersection = []
    disjoint = []
    temp = []
    for elem1 in range (0,N-1):
        for elem2 in range (elem1+1, N ):
            intersection = getIntersectingElements(FSTk[elem1],FSTk[elem2])
            disjoint = getDisjointElements(FSTk[elem1],FSTk[elem2])
            temp = intersection + disjoint
            temp.sort()
            if(len(temp) == newK):
                if temp not in listOfLists:
                    this_hashcode = getUniqueHashCode(temp)
                    isNFST = isSuperSetofNFST(intersection,disjoint)
                    if isNFST == 1 :
                        NFST_hash[this_hashcode] = 1
                    elif(getEventOccurence(temp)  < minoccur):
                        NFST_hash[this_hashcode] = 1
                    else:
                        listOfLists.append(temp)
    return listOfLists



def genAssrules(FST,alg=1):
    AllAssRules = []
    for thislist in FST:
        for smthislist in FST:
            if (issubset(smthislist,thislist)) :
                a1 = getIntersectingElements(smthislist,thislist)
                a2 = getDisjointElements(smthislist,thislist)
                if (len(a1)>0 and len(a2)>0):
                    a1.sort()
                    a2.sort()
                    comb = a1 + a2
                    sortedcomb = sorted(comb)
                    if(alg==1):
                        Sup_list = OccurenceCount[getUniqueHashCode(sortedcomb)]
                        Sup_list1 = OccurenceCount[getUniqueHashCode(a1)]
```

```
                if(alg==2):
                    Sup_list = OccurenceCountBF[getUniqueHashCode(sortedcomb)]
                    Sup_list1 = OccurenceCountBF[getUniqueHashCode(a1)]

                sup = str(round((Sup_list/TotalTransact)*100,2)) + "%"
                conf12 = Sup_list/Sup_list1
                if(conf12 >= minConf):
                    lista = []
                    lista.append(a1)
                    lista.append(a2)
                    lista.append(sup)
                    conf = str(round(conf12*100,2)) + "%"
                    lista.append(conf)
                    if lista not in AllAssRules:
                        AllAssRules.append(lista)

    return AllAssRules


#Program Starting
import time

store = 1
print('Choose your Store:')
print('1 for Marshals\n2 for Deli\n3 for Sports Center\n4 for OfficeDepo\n5 for SpeedCars')
store = input()
minSupt = float(input('Enter minimum support (in %)'))
minConf = float(input('Enter minimum confidence (in %)'))
fileObject = open('./StoresDB/store'+str(store)+'.txt', 'r')

transactions = []
Singleitems = []
OccurenceCountTemp = {}
OccurenceCount = {}
HashCode = {}
TotalTransact = 0
OccurenceCountBF = {}

start_time = time.time()

#Scan all the trasactions to generate list of SingleItems
for row in fileObject:
    tokenList = row.split(' ')
    ThisTrans = tokenList[1].rstrip('\n').split(',')
    transactions.append(ThisTrans)
    TotalTransact += 1
    for item in ThisTrans:
        if item not in Singleitems:
```

```
            Singleitems.append(item)
            OccurenceCountTemp[item] = 1
        else:
            OccurenceCountTemp[item] += 1
Singleitems.sort()

FST1 = []
FST = []
NFST_hash = {}
NFST = []
k = 2
minoccur = (minSupt * TotalTransact )/100
minConf = minConf/100

for item in Singleitems :
    index = Singleitems.index(item)
    HashCode[item] = index
    if OccurenceCountTemp[item] >= minoccur :
        OccurenceCount[HashCode[item]] = OccurenceCountTemp[item]
        OccurenceCountBF[HashCode[item]] = OccurenceCountTemp[item]
        temp = [item]
        FST1.append(temp)
this_FST = []
prev_FST = []
FST = []
k = 2
time_pause_Bf_start = time.time()

prev_FST = FST1
FST = FST1

#Generate all FSTs by Apriori Alg
while True :
    this_FST = getFrequentItemSets_Apr(prev_FST,k,minoccur)
    FST += this_FST
    prev_FST = this_FST
    k += 1
    if len(this_FST) < 2 :
        break

asrAP = genAssrules(FST)
end_time_apr = time.time()
time_pause_Bf_stop = time.time()

#Generate all FSTs by Brute Force Alg
FST_bf = []
n = len(Singleitems)
for CombLent in range(1,n):
```

```
    CombinationList = []
    this_FST_bf = []
    temp_list = [None]*CombLent
    GenCombin(Singleitems,temp_list,0,n-1,0,CombLent)
    this_FST_bf = getFSTfromAllComb(CombinationList)
    FST_bf += this_FST_bf

asrBF = genAssrules(FST_bf,2)
end_time_bf = time.time()

apr_time = round((end_time_apr - start_time) *1000 ,2)
bf_time = round((end_time_bf - start_time - (time_pause_Bf_stop -
time_pause_Bf_start))*1000,2)
time_impr = bf_time - apr_time
print("\n\n\nTransactions\n\n")
for trans in transactions:
    print(trans,"\n")

print("\nUser Chose MinSupport :",minSupt," and MinConfidence: ",minConf*100)
print("\nAssociation rules generated by Apriori Algorithm: ")
for elem in asrAP:
    print("\n",elem[0],"----->",elem[1],"    Sup: ",elem[2],"  Conf: ",elem[3])
print("\n\nAssociation rules generated by Brute Force Algorithm: ")
for elem in asrBF:
    print("\n",elem[0],"----->",elem[1],"    Sup: ",elem[2],"  Conf: ",elem[3])
print("\nTotal Association rules generated by Apriori are : ",len(asrAP)," and by Brute Force
are: ",len(asrBF))
print("\nTime taken for generating Association rules from Apriori is :",apr_time,"ms and from
Brute Force is :",bf_time,"ms" )
print("Time improvement by Apriori Alg : ",time_impr,"ms")
```