CS 524

# Data Mining (A-2)
## Frequent Pattern Matching

## Introduction

The aim of this assignment is to learn how to implement algorithms for searching frequent itemsets in a dataset and generating association rules.

## Task-1

For this task, I implemented Apriori algorithm and ECLAT algorithm in python.

**Apriori** is an algorithm for frequent item set mining and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database.

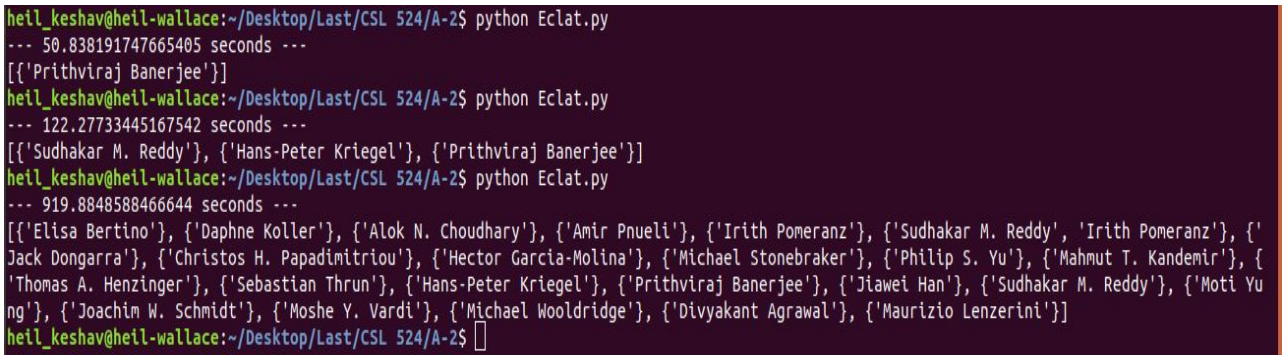
**Eclat** is an algorithm for discovering frequent itemsets in a transaction database. It uses a depth-first search for discovering frequent itemsets instead of a breadth-first search.

## Task-2

Apriori Algorithm output:

```
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$ python Apriori.py
--- 237.86395931243896 seconds ---
[[['Sudhakar M. Reddy'], ['Hans-Peter Kriegel'], ['Prithviraj Banerjee']], []]
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$ python Apriori.py
--- 226.45862412452698 seconds ---
[[['Jiawei Han'], ['Christos H. Papadimitriou'], ['Michael Wooldridge'], ['Elisa
 Bertino'], ['Daphne Koller'], ['Alok N. Choudhary'], ['Philip S. Yu'], ['Thomas
 A. Henzinger'], ['Sudhakar M. Reddy'], ['Irith Pomeranz'], ['Sebastian Thrun'],
 ['Divyakant Agrawal'], ['Jack Dongarra'], ['Moshe Y. Vardi'], ['Hans-Peter Krie
gel'], ['Moti Yung'], ['Amir Pnueli'], ['Joachim W. Schmidt'], ['Maurizio Lenzer
ini'], ['Michael Stonebraker'], ['Hector Garcia-Molina'], ['Prithviraj Banerjee'
], ['Mahmut T. Kandemir']], [['Irith Pomeranz', 'Sudhakar M. Reddy']], []]
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$
```

ECLAT Algorithm Output:




| Min Support | Eclat (Time in sec) | Apriori (Time in sec) |
|---|---|---|
| **20** | **50.838** | **227.967** |
| **16** | **122.277** | **237.863** |
| **12** | **919.884** | **226.45** |

## Machine Configuration:

heil-wallace

description: Computer

width: 64 bits

capabilities: smp vsyscall32

***-core**

description: Motherboard

physical id: 0

***-memory**

description: System memory

physical id: 0

size: 7891MiB

***-cpu**

product: Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz

vendor: Intel Corp.

physical id: 1

bus info: cpu@0

size: 2492MHz

capacity: 2700MHz

width: 64 bits

# Task-3

**Strong Association rule**: An association rule having support and confidence greater than or equal to a user-specified minimum support threshold and respectively a minimum confidence threshold. **Lift** is an objective measure of "interestingness" that has been used in various fields including statistics. For these tasks, I took min support as 12 and min confidence 0 and used lift as a criteria.

Following strong association rules from the frequent co-authorships are obtained:

```
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$ python Task3.py
--- 236.51015758514404 seconds took for frequent item set generation ---
[[['Jiawei Han'], ['Christos H. Papadimitriou'], ['Michael Wooldridge'], ['Elisa
 Bertino'], ['Daphne Koller'], ['Alok N. Choudhary'], ['Philip S. Yu'], ['Thomas
 A. Henzinger'], ['Sudhakar M. Reddy'], ['Irith Pomeranz'], ['Sebastian Thrun'],
 ['Divyakant Agrawal'], ['Jack Dongarra'], ['Moshe Y. Vardi'], ['Hans-Peter Krie
gel'], ['Moti Yung'], ['Amir Pnueli'], ['Joachim W. Schmidt'], ['Maurizio Lenzer
ini'], ['Michael Stonebraker'], ['Hector Garcia-Molina'], ['Prithviraj Banerjee'
], ['Mahmut T. Kandemir']], [['Irith Pomeranz', 'Sudhakar M. Reddy']], []]


['Sudhakar M. Reddy'] ==> ['Irith Pomeranz'] confidence: 0.7058823529411765
['Irith Pomeranz'] ==> ['Sudhakar M. Reddy'] confidence: 0.8571428571428571
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$
```

```
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$ python Task3.py
--- 253.16718935966492 seconds took for frequent item set generation ---
[[['Jiawei Han'], ['Christos H. Papadimitriou'], ['Michael Wooldridge'], ['Elisa
 Bertino'], ['Daphne Koller'], ['Alok N. Choudhary'], ['Philip S. Yu'], ['Thomas
 A. Henzinger'], ['Sudhakar M. Reddy'], ['Irith Pomeranz'], ['Sebastian Thrun'],
 ['Divyakant Agrawal'], ['Jack Dongarra'], ['Moshe Y. Vardi'], ['Hans-Peter Krie
gel'], ['Moti Yung'], ['Amir Pnueli'], ['Joachim W. Schmidt'], ['Maurizio Lenzer
ini'], ['Michael Stonebraker'], ['Hector Garcia-Molina'], ['Prithviraj Banerjee'
], ['Mahmut T. Kandemir']], [['Irith Pomeranz', 'Sudhakar M. Reddy']], []]


['Sudhakar M. Reddy'] ==> ['Irith Pomeranz'] Lift: 0.05042016806722689
['Irith Pomeranz'] ==> ['Sudhakar M. Reddy'] Lift: 0.05042016806722689
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$
```

The first part shows the frequent item list, while lower half shows strong association rules. As value of lift is near to 0, it shows strong negative correlation.

Note: Here, A and B both have 1 element (A->B). The computational power of this machine was not efficient to work with lesser min support, so I worked with 12 (min support)

## Task-4

We cannot judge the "interestingness" of strong association rules obtained in task 3 because of the lift measure that we used earlier. It includes the null count which affects the overall calculation. Null content refers to the transactions in which neither A nor B is present (A==>B) and yet it still affects the lift value. So, to measure interestingness, I used Max confidence which is a Null-Invariant Correlation measure.

| Confidence Measure | Definition |
|---|---|
| All Confidence | $\frac{s(A \cap B)}{\max\{s(A), s(B)\}}$ |
| Kulczynski | $\frac{P(A\|B)+P(B\|A)}{2}$ |
| Max Confidence | $\max\{P(A\|B), P(B\|A)\}$ |

```
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$ python Task3.py
--- 250.91481471061707 seconds took for frequent item set generation ---
[[['Jiawei Han'], ['Christos H. Papadimitriou'], ['Michael Wooldridge'], ['Elisa
 Bertino'], ['Daphne Koller'], ['Alok N. Choudhary'], ['Philip S. Yu'], ['Thomas
 A. Henzinger'], ['Sudhakar M. Reddy'], ['Irith Pomeranz'], ['Sebastian Thrun'],
 ['Divyakant Agrawal'], ['Jack Dongarra'], ['Moshe Y. Vardi'], ['Hans-Peter Krie
gel'], ['Moti Yung'], ['Amir Pnueli'], ['Joachim W. Schmidt'], ['Maurizio Lenzer
ini'], ['Michael Stonebraker'], ['Hector Garcia-Molina'], ['Prithviraj Banerjee'
], ['Mahmut T. Kandemir']], [['Sudhakar M. Reddy', 'Irith Pomeranz']], []]


['Irith Pomeranz'] ==> ['Sudhakar M. Reddy'] Max Confidence: 0.8571428571428571
['Sudhakar M. Reddy'] ==> ['Irith Pomeranz'] Max Confidence: 0.8571428571428571
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$
```

So, after calculation of max confidence for these association rules, we can see that both of them are interesting as their value is greater than 0.5

## Task-5

For this task, I used Kulczynski measure. If the value is less than 1, then it means that there is a negative correlation. If it is 1, then that would mean that there is no correlation and if it is greater than 1, then there is a positive correlation. So, for this task, we need to find strongly negatively correlated frequent sets.

```
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$ python Task3.py
--- 254.0788984298706 seconds took for frequent item set generation ---
[[['Jiawei Han'], ['Christos H. Papadimitriou'], ['Michael Wooldridge'], ['Elisa Berti
no'], ['Daphne Koller'], ['Alok N. Choudhary'], ['Philip S. Yu'], ['Thomas A. Henzinge
r'], ['Sudhakar M. Reddy'], ['Irith Pomeranz'], ['Sebastian Thrun'], ['Divyakant Agraw
al'], ['Jack Dongarra'], ['Moshe Y. Vardi'], ['Hans-Peter Kriegel'], ['Moti Yung'], ['
Amir Pnueli'], ['Joachim W. Schmidt'], ['Maurizio Lenzerini'], ['Michael Stonebraker']
, ['Hector Garcia-Molina'], ['Prithviraj Banerjee'], ['Mahmut T. Kandemir']], [['Sudha
kar M. Reddy', 'Irith Pomeranz']], []]


['Irith Pomeranz'] ==> ['Sudhakar M. Reddy'] Kulczynski Confidence: 0.7815126050420168
['Sudhakar M. Reddy'] ==> ['Irith Pomeranz'] Kulczynski Confidence: 0.7815126050420168
heil_keshav@heil-wallace:~/Desktop/Last/CSL 524/A-2$
```

Both the association rules are negatively correlated frequent sets as for both the Kulczynski confidence measure is less than 1.

# References:


1. https://adataanalyst.com/machine-learning/apriori-algorithm-python-3-0/
2. http://simpledatamining.blogspot.com/2015/02/lift.html
3. https://www.philippe-fournier-viger.com/spmf/Eclat_dEclat.php
4. http://simpledatamining.blogspot.com/2015/03/null-invariant-measures-of.html