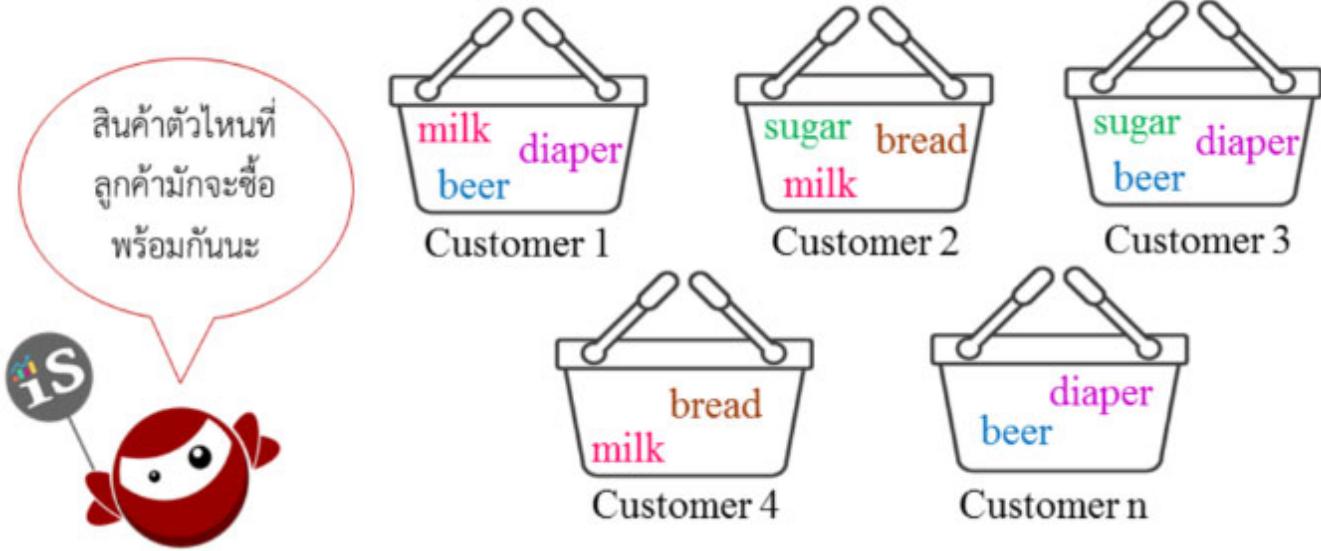


Basket Analysis



- by  MLxtend

unasso, rvl e ajo w j

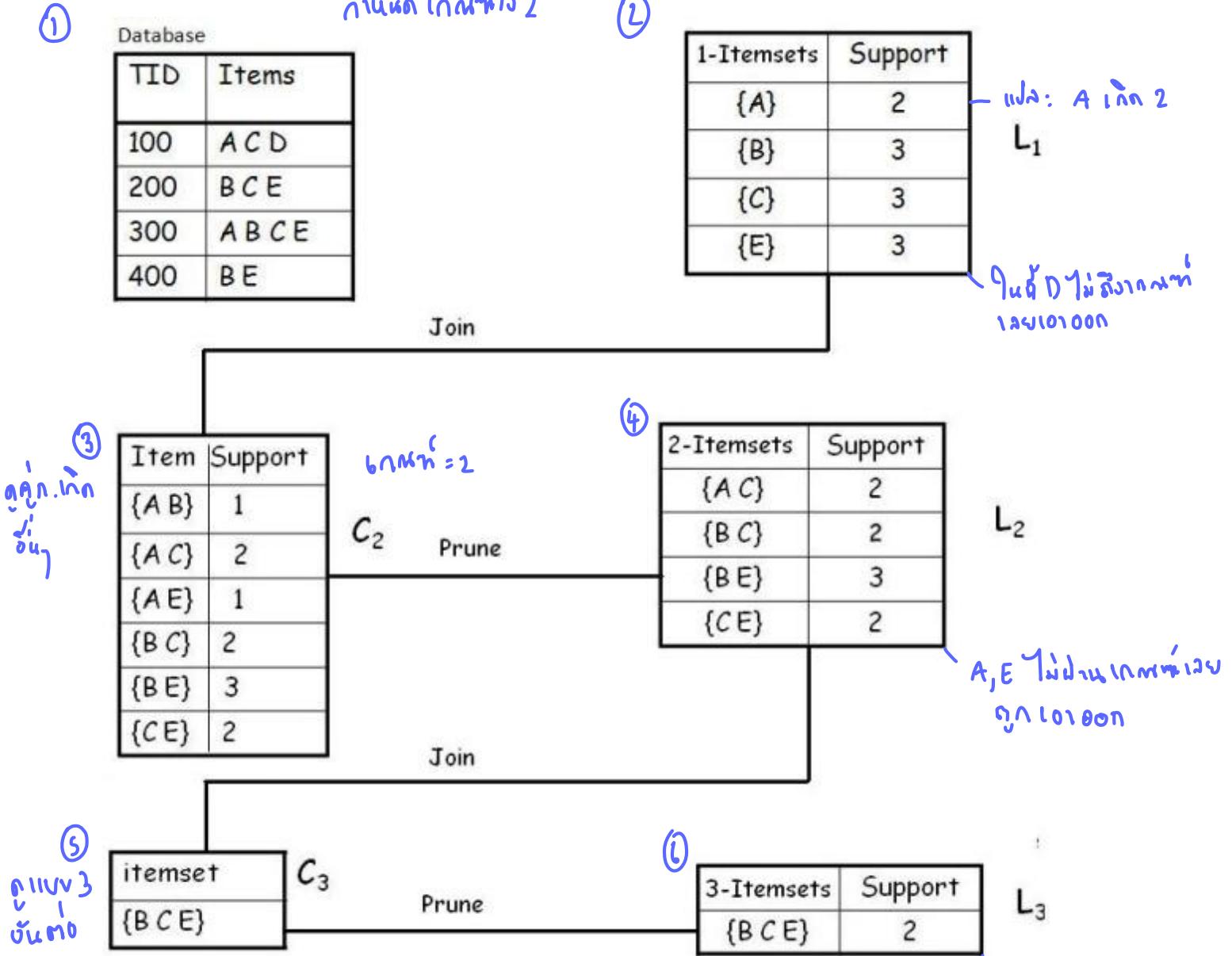
Apriori Algorithm

อัลกอริทึม Apriori เป็นเทคนิคที่ถูกนำมาใช้ในการขุดหา association rule ในฐานข้อมูลที่มีการทำธุรกรรม มีประสิทธิภาพในการระบุชุดรายการที่เป็นที่นิยมและสร้าง association rule อัลกอริทึมนี้ใช้วิธีการ "bottom-up" และมีพื้นฐานบนหลัก Apriori

Key Steps: "ກໍາຕົວວິທະຍາກົດລົງທະບຽນ"

- Generate Frequent Itemsets:** อัลกอริทึมเริ่มด้วยการค้นหา frequent itemsets ซึ่งเป็นชุดของสินค้าที่เกิดขึ้นพร้อมกันในการทำธุรกรรมมากกว่า **minimum support threshold** = $\frac{\text{จำนวนที่ผ่าน muster}}{\text{จำนวน transaction}} \times 100\%$ (แล้วแต่เรียกตามความถี่ของ transaction) ตัวอย่าง $\frac{10}{100} \times 100\% = 10\%$
 - Join Step:** ในขั้นตอนนี้อัลกอริทึมสร้าง candidate itemsets โดยการรวม frequent itemsets จากการทำซ้ำก่อนหน้า \rightarrow ถ้า **ทั้งห้า** ตัว "ห้ามมีกันหมด" ตัว **ก็** ว่า **เป็นcandidate** 
 - Prune Step:** candidate itemsets ที่ไม่ตรงตามเกณฑ์ minimum support threshold จะถูกตัดทิ้งเพื่อลดพื้นที่ในการค้นหา \rightarrow ในที่นี่ **ไม่ต้องเก็บห้าก็ได้** 
 - Repeat the Join and Prune Steps:** อัลกอริทึมทำซ้ำในกระบวนการรวมและตัดจนกว่าจะไม่สามารถสร้าง frequent itemsets ใหม่ได้ **แล้วถูกห้ามมีกันหมด...** ท่อปัสสาวะ 
 - Generate Association Rules:** เมื่อได้รับ frequent itemsets และ association rules จะถูกสร้างขึ้นจากชุดรายการเหล่านี้ กฎประกอบด้วย antecedent (ด้านซ้าย) และส่วน consequent (ด้านขวา) 
 - Calculate Metrics:** ตัวชี้วัดเช่น support, confidence, และ lift ถูกคำนวณสำหรับแต่ละ association rule เพื่อวัดความแข็งแกร่งและความสำคัญของความสัมพันธ์

ค้นหา itemset ที่บ่อย



(image from: <https://www.codeproject.com/Articles/70371/Apriori-Algorithm>)

Extract frequent itemsets

เริ่มจากการหา frequent itemsets

- ใช้ dataset จากไฟล์ Groceries_dataset.csv
- อัปโหลดขึ้น sample_data

ต้องคำนึงถึง
ไม่เกิน 1 บิลลี่ แล้ว
ต้องคำนึงถึง
เบอร์ เท่าที่ต้องการ
และ rule ต้องคำนึงถึง
support ที่ต้องการ

สร้าง DataFrame

- เรียกใช้ pandas

```
import pandas as pd
```

- อ่านไฟล์ Groceries_dataset.csv จาก sample_data โดยคลิกขวาที่ชื่อไฟล์ และเลือก Copy path

```
data = pd.read_csv('/content/sample_data/Groceries_dataset.csv')
data.head()
```

	Member_number	Date	itemDescription
0	1808	21-07-2015	tropical fruit
1	2552	05-01-2015	whole milk
2	2300	19-09-2015	pip fruit
3	1187	12-12-2015	other vegetables
4	3037	01-02-2015	whole milk

```
data.shape
```

row col
(38765, 3)

```
data.sort_values(['Member_number', 'Date'])
```

	Member_number	Date	itemDescription
4843	1000	15-03-2015	sausage
8395	1000	15-03-2015	whole milk
20992	1000	15-03-2015	semi-finished bread
24544	1000	15-03-2015	yogurt
13331	1000	24-06-2014	whole milk
...
3578	5000	10-02-2015	soda
19727	5000	10-02-2015	root vegetables
34885	5000	10-02-2015	semi-finished bread
9340	5000	16-11-2014	bottled beer
25489	5000	16-11-2014	other vegetables

38765 rows × 3 columns

```
len(data.groupby(['Member_number', 'Date']))
```

14963

- ดูรายการสินค้าและความถี่ *

```
data['itemDescription'].value_counts()
```

whole milk	2502
other vegetables	1898
rolls/buns	1716
soda	1514
yogurt	1334
...	
rubbing alcohol	5
bags	4
baby cosmetics	3
kitchen utensil	1
preservation products	1

Name: itemDescription, Length: 167, dtype: int64

၁၆၇

၁၆၇

သို့ မိမိ၏ အား ၁၆၇ ရှုံး၏

- ทำให้อยู่ในรูปของ list of transactions

Group အား ၃ မျှ၏ ၁၆၇

list(data.groupby(['Member_number', 'Date']))[:3] — ၃ မျှ၏ (0,1,2)

၁၆၇ အား ၃ မျှ၏ ၁၆၇

```

→ [((1000, '15-03-2015'),
      Member_number           Date      itemDescription
      4843                  1000  15-03-2015      sausage
      8395                  1000  15-03-2015  whole milk
      20992                 1000  15-03-2015 semi-finished bread
      24544                 1000  15-03-2015      yogurt),
→ ((1000, '24-06-2014'),
      Member_number           Date      itemDescription
      13331                 1000  24-06-2014  whole milk
      29480                 1000  24-06-2014      pastry
      32851                 1000  24-06-2014 salty snack),
→ ((1000, '24-07-2015'),
      Member_number           Date      itemDescription
      2047                  1000  24-07-2015  canned beer
      18196                 1000  24-07-2015 misc. beverages)]

```

list(data.groupby(['Member_number', 'Date']))[0]

```

((1000, '15-03-2015'),
      Member_number           Date      itemDescription
      4843                  1000  15-03-2015      sausage
      8395                  1000  15-03-2015  whole milk
      20992                 1000  15-03-2015 semi-finished bread
      24544                 1000  15-03-2015      yogurt)

```

transaction: key, value
value အား value အား (dataframe)

list(data.groupby(['Member_number', 'Date']))[0][1]

၁၆၇ အား ၁၆၇

dataframe	Member_number	Date	itemDescription
4843	1000	15-03-2015	sausage
8395	1000	15-03-2015	whole milk
20992	1000	15-03-2015	semi-finished bread
24544	1000	15-03-2015	yogurt

```
list(data.groupby(['Member_number', 'Date']))[0][1]['itemDescription']
```

ກົດໃຫຍ້
trnsac.

4843	sausage
8395	whole milk
20992	semi-finished bread
24544	yogurt

Name: itemDescription, dtype: object

ດີດ້ວຍກຳນົດ
col.

```
list(data.groupby(
```

['Member_number', 'Date']))[0][1]['itemDescription'].tolist() ໄຟລ່າຍິນດູລັດ

ດີຕັ້ງ.. ['sausage', 'whole milk', 'semi-finished bread', 'yogurt']

```
transactions = [
    a[1]['itemDescription'].tolist()
    for a in list(data.groupby(['Member_number', 'Date']))]
transactions[:10] ເປັນ trnsac.
```

ດີຕັ້ງສັນກັບ
ໂອງໄລ່ຕົວ trnsac.

```
[['sausage', 'whole milk', 'semi-finished bread', 'yogurt'],
 ['whole milk', 'pastry', 'salty snack'],
 ['canned beer', 'misc. beverages'],
 ['sausage', 'hygiene articles'],
 ['soda', 'pickled vegetables'],
 ['frankfurter', 'curd'],
 ['sausage', 'whole milk', 'rolls/buns'],
 ['whole milk', 'soda'],
 ['beef', 'white bread'],
 ['frankfurter', 'soda', 'whipped/sour cream']]
```

▼ ປັບຮູບແບບ transactions

transactions ຈະຕ້ອງອີ່ມໃນຮູບແບບຂອງຕາரາງ ທີ່ແຕ່ລະແກ່ແສດງການມີ (True) ອີ່ມີ (False) ສິນຄ້າໃນ 1
ຮາຍການຂອງ transaction

ຮູບແບບ
matrix

- ໃຊ້ TransactionEncoder

∴ ອົກການຂໍມູນຕາරາງ 15,000 ໄດ້

ໄລຍະ 175 col.

```
from mlxtend.preprocessing import TransactionEncoder
```

ແກ່ລະແກວ: ບອກວ່າດັກນີ້ຕັ້ງກັນບໍ່
ກົດເປັນ true ດ້ວຍບໍ່ກົດ false

```
te = TransactionEncoder()
```

```
→ te.fit(transactions)
```

```
t_encode = te.transform(transactions)
```

```
t_encode
```

```
array([[False, False, False, ..., True, True, False],
       [False, False, False, ..., True, False, False],
       [False, False, False, ..., False, False, False],
       ...,
       [False, False, False, ..., False, False, False],
       [False, False, False, ..., False, False, False],
       [False, False, False, ..., False, False, False]])
```

ດີ

colonization. fit

```
df = pd.DataFrame(t_encode, columns=te.columns_)
```

	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics
0		False	False	False	False
1		False	False	False	False
2		False	False	False	False
3		False	False	False	False
4		False	False	False	False
...
14958		False	False	False	False
14959		False	False	False	False
14960		False	False	False	False
14961		False	False	False	False
14962		False	False	False	False

▼ Apriori

- մայ,

หา frequent itemsets ด้วย Apriori

- เรียกใช้ apriori library

```
from mlxtend.frequent_patterns import apriori
```

```
import warnings
warnings.filterwarnings('ignore', category=DeprecationWarning)
```

- ໜ frequent itemsets

15000*.001

15.0

10 मिनी

minimum
(minimum support)

```
fia = apriori(df, min_support=0.001, use_colnames=True)
```

9. 0.067 → 1000 / 15000
กม. 9. 0.067 → 1000 / 15000

	support	itemsets
0	0.004010	(Instant food products)
1	<u>0.021386</u> ^{2% มาก} ₁₅₀₀₀	(UHT-milk)
2	0.001470	(abrasive cleaner)
3	0.001938	(artif. sweetener)
4	0.008087	(baking powder)
...
745	0.001136	(rolls/buns, sausage, whole milk)
746	0.001002	(rolls/buns, soda, whole milk)
747	0.001337	(rolls/buns, yogurt, whole milk)
748	0.001069	(whole milk, sausage, soda)
749	0.001470	(whole milk, sausage, yogurt)

- ผลลัพธ์ที่ได้จะเป็น DataFrame ^{750 รายการหนึ่ง}
- สามารถปรับแต่งได้ตามต้องการ เช่น เพิ่ม Series ที่แสดงจำนวนรายการใน itemset

↙ e.g.

```
fia['length'] = fia['itemsets'].apply(lambda x: len(x))
fia
```

	support	itemsets	length
0	0.004010	(Instant food products)	1
1	0.021386	(UHT-milk)	1
2	0.001470	(abrasive cleaner)	1
3	0.001938	(artif. sweetener)	1
4	0.008087	(baking powder)	1
...
745	0.001136	(rolls/buns, sausage, whole milk)	3
746	0.001002	(rolls/buns, soda, whole milk)	3
747	0.001337	(rolls/buns, yogurt, whole milk)	3
748	0.001069	(whole milk, sausage, soda)	3
749	0.001470	(whole milk, sausage, yogurt)	3

750 rows × 3 columns

- สามารถค้นหา itemsets ที่ต้องการได้

```
fia[fia['length']==2]
```

	support	itemsets	length
149	0.001069	(bottled water, UHT-milk)	2
150	0.002139	(other vegetables, UHT-milk)	2
151	0.001804	(rolls/buns, UHT-milk)	2
152	0.001002	(root vegetables, UHT-milk)	2
153	0.001136	(sausage, UHT-milk)	2
...
736	0.002941	(yogurt, whipped/sour cream)	2
737	0.003141	(whole milk, white bread)	2
738	0.001069	(yogurt, white bread)	2
739	0.001270	(whole milk, white wine)	2
740	0.011161	(whole milk, yogurt)	2

592 rows × 3 columns

1% ≈ 150 รายการ.

```
fia[(fia['length']==2) & (fia['support']>=0.01)]
```

	support	itemsets	length
609	0.010559	(rolls/buns, other vegetables)	2
625	0.014837	(whole milk, other vegetables)	2
677	0.013968	(rolls/buns, whole milk)	2
717	0.011629	(soda, whole milk)	2
740	0.011161	(whole milk, yogurt)	2

```
fia[fia['itemsets'] == {'yogurt', 'whole milk'}]
```

= รายการ set ที่เราต้องการ ที่มีความถี่
ว่า 10% ของกันและกัน

	support	itemsets	length
740	0.011161	(whole milk, yogurt)	2



FP-Growth Algorithm

= Algor. นี้

ดูคลิป รีวิว กับ Apri

อัลกอริทึม FP-Growth (Frequent Pattern-Growth) เป็นทางเลือกในการคัดกรอง frequent itemsets แทนอัลกอริทึม Apriori ในชุดข้อมูลขนาดใหญ่

Key Steps:

1. Build FP-Tree: อัลกอริทึมสร้าง FP-Tree → เอามาใช้ ทั้งในเก็บข้อมูลในหน้าจอ

(มีขั้นตอนทั้ง 1, 2, 3 itemset สร้าง bottom-up ที่เรียกว่า Apriori)

2. Construct Conditional Pattern Bases: สร้าง conditional pattern base จาก FP-Tree

3. Mine Frequent Itemsets: หา Frequent itemsets จากการสำรวจ Conditional Pattern Bases โดยไม่ต้องสร้างและติดรายการตัวเลือก

4. Generate Association Rules: เช่นเดียวกับอัลกอริทึม Apriori กว่าความสัมพันธ์จะถูกสร้างขึ้นจาก frequent itemsets

FP-Growth

หา frequent itemsets ด้วย FP-Growth

- เรียกใช้ fpgrowth library

```
from mlxtend.frequent_patterns import fpgrowth
```

- หา frequent itemsets

```
fif = fpgrowth(df, min_support=0.001, use_colnames=True)  
fif
```

	support	itemsets
0	0.157923	(whole milk)
1	0.085879	(yogurt)
2	0.060349	(sausage)
3	0.009490	(semi-finished bread)
4	0.051728	(pastry)
...
745	0.001403	(chewing gum, yogurt)
746	0.001069	(chewing gum, other vegetables)
747	0.001002	(chewing gum, soda)
748	0.001069	(whole milk, pasta)
749	0.001002	(rolls/buns, seasonal products)

750 rows × 2 columns

▷ ผู้อ.นักศึกษาท่านใดที่ต้องการเรียนรู้ FP-growth และ Apriori ก็สามารถหาน้ำหน้า

```
fif[fif['itemsets'] == {'sausage', 'UHT-milk'}]
```

	support	itemsets
565	0.001136	(sausage, UHT-milk)

```
fia[fia['length']==2]
```

	support	itemsets	length
149	0.001069	(bottled water, UHT-milk)	2
150	0.002139	(other vegetables, UHT-milk)	2
151	0.001804	(rolls/buns, UHT-milk)	2
152	0.001002	(root vegetables, UHT-milk)	2
153	0.001136	(sausage, UHT-milk)	2
...
736	0.002941	(yogurt, whipped/sour cream)	2
737	0.003141	(whole milk, white bread)	2
738	0.001069	(yogurt, white bread)	2
739	0.001270	(whole milk, white wine)	2
740	0.011161	(whole milk, yogurt)	2

592 rows × 3 columns

▼ Association Rules

Association rules เป็นการแสดงความสัมพันธ์ทางตรรกะระหว่างสินค้าโดยขึ้นอยู่กับการเกิดขึ้นร่วมกันในการทำธุรกรรม กฎเหล่านี้ช่วยให้มีความเข้าใจในพฤติกรรมและความชอบของลูกค้า เพื่อช่วยธุรกิจในการตัดสินใจอย่างมีข้อมูล

Format of Association Rule:

- กฎ: กฎที่เกิดก่อน กฎที่ตามมา
- Antecedent (Left-hand side) -> Consequent (Right-hand side)
ก้า... → ॥ ก้า...

e.g. ล้างหน้า และซื้อขนมปัง

▼ Metrics 9 แห่ง 10 ประวัติการซื้อขาย คือ 5 ค่า (นับครั้งที่เกิด)

1. Support: (ต่อ. 9 ร้อยเปอร์เซนต์)

Support is the proportion of transactions that contain both the antecedent and the consequent of an association rule.

Support(A->B) = (Number of Transactions containing A and B) / (Total Number of Transactions)

มี 2 ชุดที่ A, B 40 รายการ

Example: Suppose out of 100 transactions, 40 contain both items A and B.

∴ Support(A->B) = 40 / 100 = 0.4

2. Confidence:

Confidence is the probability of finding the consequent in a transaction given that the antecedent is already present in that transaction. *ពិនិត្យការងារថា ពីការបានដាក់មិនមែន*

$\text{Confidence}(A \rightarrow B) = (\text{Number of Transactions containing } A \text{ and } B) / (\text{Number of Transactions containing } A)$

នៅលើ 100 នៅទាំង

មាន A

នៅក្នុង 40 នៅលើ 60 នៅទាំង

(ការងារ A នៅលើ 60 នៅលើ 100 នៅទាំង B)

Example: Out of the 60 transactions containing item A, 40 also contain item B.

នៅលើ 60 នៅទាំង 40 នៅលើ 60 នៅទាំង

ដោយ

$\text{Confidence}(A \rightarrow B) = 40 / 60 = \underline{0.6667}$

នៅលើ 60 នៅទាំង 40 នៅលើ 60 នៅទាំង

3. Lift:

Lift measures the ratio of observed support to expected support for an association rule. It indicates how much more likely the items in the antecedent and consequent are to be bought together compared to if they were bought independently.

ត្រឡប់នៅក្នុង Antecedent
និង Consequent

នូវ support

$\text{Lift}(A \rightarrow B) = \frac{\text{Support}(A \rightarrow B)}{\text{Support}(A) * \text{Support}(B)}$

Example:

Suppose we have the following information:

- Total number of transactions: 1000
- Transactions containing item A: 600
- Transactions containing item B: 400
- Transactions containing both A and B: 300

Calculate Lift for Rule A->B:

1. Calculate Support:

- $\text{Support}(A) = \text{Transactions containing } A / \text{Total transactions} = 600 / 1000 = 0.6$
- $\text{Support}(B) = \text{Transactions containing } B / \text{Total transactions} = 400 / 1000 = 0.4$
- $\text{Support}(A \rightarrow B) = \text{Transactions containing } A \text{ and } B / \text{Total transactions} = 300 / 1000 = 0.3$

2. Calculate Lift:

- $\text{Lift}(A \rightarrow B) = \text{Support}(A \rightarrow B) / (\text{Support}(A) * \text{Support}(B))$
- $\text{Lift}(A \rightarrow B) = 0.3 / (0.6 * 0.4) = \underline{1.25}$

→ Interpretation:

- ការងារ $\text{Lift}(A \rightarrow B) = 1$ សេចក្តីថាការកើតិតនៃ A និង B មិនកំពុងក្រោមគ្នានេះ
- ការងារ $\text{Lift}(A \rightarrow B) > 1$ សេចក្តីថាការកើតិតនៃ A ធ្វើការក្នុងការកើតិតនៃ B
- ការងារ $\text{Lift}(A \rightarrow B) < 1$ សេចក្តីថាការកើតិតនៃ A ត្រូវការកើតិតនៃ B

(A ការកើតិតនៃ B)

ในตัวอย่างนี้ Lift(A->B) มากกว่า 1 ชี้งแสดงถึงการเขื่อมโยงบวกระหว่างรายการ A และ B นี้ เป็นการบ่งชี้ว่า ลูกค้าที่ซื้อรายการ A มีโอกาสที่จะซื้อรายการ B ด้วย และการมี A เพิ่มโอกาสในการซื้อ B

4. Leverage:

Leverage measures the difference between the actual occurrence of both the antecedent and the consequent in transactions and the expected occurrence if they were independent.

$$\text{Leverage}(A \rightarrow B) = \text{Support}(A \rightarrow B) - (\text{Support}(A) * \text{Support}(B))$$

=

คลับ กับ lift 1.26
1.98 ลบ

Example:

Suppose we have the following information:

- Total number of transactions: 1000
- Transactions containing item A: 600
- Transactions containing item B: 400
- Transactions containing both A and B: 200

Calculate Leverage for Rule A->B:

1. Calculate Support:

- $\text{Support}(A) = \text{Transactions containing A} / \text{Total transactions} = 600 / 1000 = 0.6$
- $\text{Support}(B) = \text{Transactions containing B} / \text{Total transactions} = 400 / 1000 = 0.4$
- $\text{Support}(A \rightarrow B) = \text{Transactions containing A and B} / \text{Total transactions} = 200 / 1000 = 0.2$

2. Calculate Leverage:

- $\text{Leverage}(A \rightarrow B) = \text{Support}(A \rightarrow B) - (\text{Support}(A) * \text{Support}(B))$
- $\text{Leverage}(A \rightarrow B) = 0.2 - (0.6 * 0.4) = 0.2 - 0.24 = -0.04$

→ Interpretation:

- หาก $\text{Leverage}(A \rightarrow B) = 0$ แสดงว่าการเกิดของ A และ B เป็นไปตามคาดหวัง ถ้า A และ B เป็นอิสระต่อ กัน
- หาก $\text{Leverage}(A \rightarrow B) > 0$ แสดงว่าการเกิดของ A และ B เกินค่าคาดหวัง
- หาก $\text{Leverage}(A \rightarrow B) < 0$ แสดงว่าการเกิดของ A และ B น้อยกว่าค่าคาดหวัง

5. Conviction:

Conviction measures the ratio of the expected frequency that the antecedent implies the consequent to the observed frequency of the implication.

ก. เชื่อมต่อ B

$$\text{Conviction}(A \rightarrow B) = (1 - \underline{\text{Support}(B)}) / (1 - \text{Confidence}(A \rightarrow B))$$

ค. เชื่อมต่อ ไม่ซึ่ง B

Example:

Suppose we have the following information:

- Total number of transactions: 1000
- Transactions containing item A: 600
- Transactions containing item B: 400
- Transactions containing both A and B: 300

Calculate Conviction for Rule A->B:

1. Calculate Support and Confidence (same as in previous examples).

2. Calculate Conviction:

- Conviction(A->B) = $(1 - \text{Support}(B)) / (1 - \text{Confidence}(A \rightarrow B))$
- Conviction(A->B) = $(1 - 0.4) / (1 - 0.5) = 0.6 / 0.5 = 1.2$

→ Interpretation:

คลับกัน

- หาก Conviction(A->B) = 1 แสดงว่า A และ B เป็นอิสระกัน
- หาก Conviction(A->B) > 1 แสดงว่าการเกิดของ A และ B ร่วมกันมากกว่าที่คาดหวัง
- หาก Conviction(A->B) < 1 แสดงว่าการเกิดของ A และ B ร่วมกันน้อยกว่าที่คาดหวัง

- สร้างจาก frequent itemsets
- เรียกใช้ association_rules library

from mlxtend.frequent_patterns import association_rules

from mlxtend.frequent_patterns import association_rules

- กำหนดเกณฑ์ในการพิจารณากฎ ห้า 5 ค่า
- supported metrics are 'support', 'confidence', 'lift', 'leverage', and 'conviction'
 - support(A->C) = support(A+C) [aka 'support'], range: [0, 1]
 - confidence(A->C) = support(A+C) / support(A), range: [0, 1]
 - lift(A->C) = confidence(A->C) / support(C), range: [0, inf]
 - leverage(A->C) = support(A->C) - support(A)*support(C), range: [-1, 1]
 - conviction = $[1 - \text{support}(C)] / [1 - \text{confidence}(A \rightarrow C)]$, range: [0, inf]

rules = association_rules(fif, metric='confidence', min_threshold=0.2)

confidence ต้องต่ำกว่า 0.2
min_threshold (20%) หมายความว่า กฎต้องมีความน่าเชื่อถือ 20% ขึ้นไป

rules[rules['lift'] > 1]

	antecedents	consequents	antecedent support	consequent support	support	confidence
0	(sausage, yogurt)	(whole milk)	0.005748	0.157923	0.001470	0.2558
1	(rolls/buns, sausage)	(whole milk)	0.005347	0.157923	0.001136	0.2125

- จึงได้ผลลัพธ์ rules กด 0
= กฎที่มีความน่าเชื่อถือ 20% ขึ้นไป คือ (sausage, yogurt) หรือ (whole milk) ต้องมีความน่าเชื่อถือ 20% ขึ้นไป

ดังนั้น กฎที่มีความน่าเชื่อถือ 20% ขึ้นไป คือ (sausage, yogurt) หรือ (whole milk)

ตัววัดประสิทธิภาพของกฎ

ค่า Confidence เป็นตัววัดประสิทธิภาพสำหรับ Association Rule โดยเป็นตัวเลขที่แสดงความน่าจะเป็นที่กลุ่มของสินค้า RHS จะถูกหยิบเข้าตะกร้าด้วย หลังจากที่กลุ่มของสินค้า LHS ถูกหยิบเข้าตะกร้าไปแล้ว

หรืออาจกล่าวได้ว่า “หากเราพิจารณาเฉพาะตะกร้าการซื้อขายที่มีกลุ่มสินค้า LHS มีตะกร้าการซื้อขายที่มี RHS อยู่ด้วยเป็นสัดส่วนเท่าไหร่”

ในตัวอย่างร้านค้าของเรา หากพิจารณา $LHS = \{\text{Cereal}\}$ และ $RHS = \{\text{Apple}\}$ จะได้ว่า $\text{Confidence}(\{\text{Cereal}\} \Rightarrow \{\text{Apple}\}) = 2/3$ (ภายใน 3 ตะกร้าที่มีการซื้อขาย Cereal มีจำนวน 2 ตะกร้าที่มีการซื้อขาย Apple ด้วย)

หาก RHS เป็นกลุ่มของสินค้าที่พบเจอในแทบทุกตะกร้าการซื้อขาย (a very frequent itemset) ไม่ว่า LHS จะเป็นกลุ่มของสินค้าใดก็ตาม $\text{Confidence}(LHS \Rightarrow RHS)$ จะมีค่าค่อนข้างสูงเสมอ ด้วยเหตุนี้ จึงมีอีกหนึ่งตัววัดประสิทธิภาพที่พิจารณาความถี่ในการพบกลุ่มของสินค้า RHS ร่วมกับค่า confidence ด้วย

