TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC

BÁO CÁO BÀI TẬP LỚN MÔN HỆ HỖ TRỢ QUYẾT ĐỊNH

Đề tài 12: Basket optimisation

Giảng viên hướng dẫn: TS. Lê Hải Hà

Sinh viên thực hiện: Nguyễn Thị Bích Ngọc

MSSV: 20185388

Lớp: Toán - Tin 01 - K63

Mã học phần: MI4216

Mã lớp học: 133591

HÀ NỘI, THÁNG 07/2022

1. Thuật ngữ "Market Basket Analysis/Optimisation"

Market Basket Analysis/Optimisation (Phân tích/Tối ưu hóa giỏ thị trường) là một kỹ thuật khai thác dữ liệu được các nhà bán lẻ sử dụng để tăng doanh số bán hàng bằng cách hiểu rõ hơn các mô hình mua hàng của khách hàng. Nó liên quan đến việc phân tích các tập dữ liệu lớn, chẳng hạn như lịch sử mua hàng, để tiết lộ các nhóm sản phẩm, cũng như các sản phẩm có khả năng được mua cùng nhau.

Việc áp dụng phân tích giỏ thị trường được hỗ trợ bởi sự ra đời của **hệ thống điểm bán hàng điện tử (POS)**. So với hồ sơ viết tay do chủ cửa hàng lưu giữ, hồ sơ kỹ thuật số do hệ thống POS tạo ra giúp các ứng dụng xử lý và phân tích khối lượng lớn dữ liệu mua hàng dễ dàng hơn.

Việc thực hiện phân tích giỏ thị trường yêu cầu nền tảng về thống kê và khoa học dữ liệu, cũng như một số kỹ năng lập trình máy tính theo thuật toán. Đối với những người không có kỹ năng kỹ thuật cần thiết, các công cụ thương mại, không có sẵn vẫn tồn tại.

Ví dụ: công cụ Phân tích giỏ hàng trong Microsoft Excel, công cụ này phân tích dữ liệu giao dịch có trong bảng tính và thực hiện phân tích giỏ hàng thị trường. Các mục được phân tích phải có liên quan với nhau bằng một ID giao dịch. Sau đó, công cụ Phân tích giỏ hàng sẽ tạo hai bảng tính: bảng tính Nhóm mặt hàng trong giỏ hàng, liệt kê các mặt hàng thường được mua cùng nhau và bảng tính Quy tắc

giỏ hàng, cho biết mức độ liên quan của các mặt hàng (Ví dụ: người mua Sản phẩm A có khả năng mua Sản phẩm B).

*** Các loại phân tích giỏ thị trường**: Có hai loại

Phân tích giỏ thị trường dự đoán (Predictive market basket analysis): Loại này xem xét các mặt hàng được mua theo trình tự để xác định bán chéo

Phân tích giỏ thị trường khác biệt (Differential market basket analysis): Loại này xem xét dữ liệu trên các cửa hàng khác nhau, cũng như các giao dịch mua từ các nhóm khách hàng khác nhau trong các thời điểm khác nhau trong ngày, tháng hoặc năm. Nếu quy tắc nắm giữ ở một thứ nguyên (như cửa hàng, khoảng thời gian hoặc nhóm khách hàng), nhưng không nắm giữ trong các thứ nguyên khác, nhà phân tích có thể xác định các yếu tố gây ra ngoại lệ. Những thông tin chi tiết này có thể dẫn đến việc cung cấp sản phẩm mới giúp thúc đẩy doanh số bán hàng cao hơn.

*** Lợi ích của phân tích giỏ thị trường**

Phân tích giỏ thị trường có thể tăng doanh số bán hàng và sự hài lòng của khách hàng. Sử dụng dữ liệu để xác định rằng các sản phẩm thường được mua cùng nhau, các nhà bán lẻ có thể tối ưu hóa vị trí đặt sản phẩm, đưa ra các ưu đãi đặc biệt và tạo các gói sản phẩm mới để khuyến khích doanh số bán hàng nhiều hơn của các kết hợp này.

Những cải tiến này có thể tạo ra doanh số bán hàng bổ sung cho nhà bán lẻ, đồng thời làm cho trải nghiệm mua sắm hiệu quả hơn và có giá trị hơn đối với khách hàng. Bằng cách sử dụng phân tích giỏ thị trường, khách hàng có thể cảm nhận được tình cảm mạnh mẽ hơn hoặc lòng trung thành với thương hiệu đối với công ty.

Ở đây ta sẽ tiến hành phân tích/tối ưu hóa giỏ thị trường bằng cách sử dụng **Association Rule Mining - Khai thác theo quy tắc liên kết**.

2. Association Rule Mining (ARM)

Khai thác theo quy tắc liên kết là một phương pháp học máy dựa trên quy tắc để khám phá các mối quan hệ thú vị giữa các biến trong cơ sở dữ liệu lớn. Nó nhằm xác định các quy tắc mạnh được phát hiện trong cơ sở dữ liệu bằng cách sử dụng một số biện pháp thống kê.

Ví dụ: sử dụng Khai thác theo quy tắc liên kết để xác định những sản phẩm nào thường được mua cùng nhau tại một cửa hàng tạp hóa dựa trên dữ liệu đang có. Điều này có thể giúp cho việc sắp xếp các mặt hàng trong cửa hàng hoặc tạo ra các sản phẩm mới để chúng có xác suất cao được khách hàng nhặt hoặc mua.

Một sản phẩm đơn lẻ được gọi là **một mặt hàng** và một tập hợp các mặt hàng được mua cùng nhau được gọi là **một giao dịch**. Một tập hợp các giao dịch được gọi là **cơ sở dữ liệu**. Một tập hợp con của các mục được gọi là **tập phổ biến**.

*** Quy tắc ARM**

Khai thác quy tắc kết hợp ARM để tìm các mẫu trong dữ liệu đã cho, chẳng hạn như:

$$\text{Milk} \rightarrow \text{Bread or (Milk, Bread)} \rightarrow \text{Eggs}$$

Phía bên trái mũi tên (LHS) ở trên được gọi là **tiền đề (antecedent)**, còn phía bên phải mũi tên (RHS) được gọi là **hệ quả (consequent)**. Nếu **đồng xuất hiện (co-occurrence)** thì được gọi là **quan hệ nhân quả (causality)**.

*** Một số cách đo lường mức độ quan trọng**

Xét một tập dữ liệu mẫu trong đó mỗi hàng tương ứng với một giao dịch:

1. {Milk, Bread, Butter}
2. {Milk, Eggs}
3. {Bread, Butter}
4. {Bread, Chocolate}
5. {Milk, Chocolate}

và cần phải tìm ra mối quan hệ của quy tắc $X \rightarrow Y$ trong đó X và Y là một tập hợp các mục. Một số cách tìm:

Support (Mức hỗ trợ): số lần xuất hiện của tập hợp mục ($X \cup Y$) chia cho tổng số giao dịch trong cơ sở dữ liệu. Nó chỉ ra mức độ lan truyền của tập phổ biến này trong cơ sở dữ liệu.

$$Supp\left(\{Bread\} \rightarrow \{Butter\}\right) = Supp\left(\{Bread, Butter\}\right) = 2/5 = 0,4$$

Confidence (Mức độ tin cậy): được đo bằng số lần tập hợp mục liên hợp xuất hiện chia cho số lần LHS xuất hiện. Tức là phần Support của tập hợp mục chia cho phần Support của tập hợp LHS. Nó đo lường tần suất quy tắc được tìm thấy là đúng.

$$Conf\left(\{Bread\} \rightarrow \{Butter\}\right) = \frac{Supp\left(\{Bread, Butter\}\right)}{Supp\left(\{Bread\}\right)} = \frac{0,4}{0,6} = 0,67$$

Lift (Mức tăng): được đo bằng cách lấy Support của tập hợp mục chia cho phép nhân của Support riêng lẻ của cả X và Y. Nó đo lường mức độ phụ thuộc vào nhau của hai tập hợp. Giá trị bằng 1 cho biết mức độ độc lập, giá trị > 1 cho biết mức độ chúng cùng xảy ra và giá trị <1 cho biết mức độ chúng phủ định sự tồn tại của nhau.

$$Lift\left(\{Bread\} \rightarrow \{Butter\}\right) = \frac{Supp\left(\{Bread, Butter\}\right)}{Supp\left(\{Bread\}\right) * Supp\left(\{Butter\}\right)} = \frac{0,4}{0,6 * 0,4} = 1,67$$

=> Độ tin cậy (Conviction) của quy luật X → Y là tỷ số giữa tần suất dự đoán của X xảy ra bất chấp Y và tần suất quan sát được của các dự đoán sai.

$$Conv\left(\{Bread\} \rightarrow \{Butter\}\right) = \frac{1 - Supp\left(\{Butter\}\right)}{1 - Conf\left(\{Bread\} \rightarrow \{Butter\}\right)} = \frac{1 - 0,4}{1 - 0,67} = 1,82$$

Quy tắc ({Bread} → {Butter}) trên sẽ không chính xác hơn 82% thường xuyên nếu mối quan hệ giữa {Bread} và {Butter} hoàn toàn là ngẫu nhiên.

3. ARM cho một bộ dữ liệu

Link bộ đữ liệu: https://www.kaggle.com/datasets/devchauhan1/market-basket-optimisationcsv

* Xử lý dữ liệu

Sử dụng thư viện **Numpy** và thư viện **Pandas** để chuyển đổi tệp .csv thành một tập dữ liệu và làm việc trên đó.

```
import pandas as pd
import numpy as np
```

Đọc tệp .csv vào khung dữ liệu:

```
df=pd.read_csv('Market_Basket_Optimisation.csv', header=None)
```

Xem vài hàng đầu tiên trong tập dữ liệu:

```
df.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
0	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt	green tea	honey	salad	mineral water	salmon	antioxydant juice	frozen smoothie	spina
1	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

Vì tất cả các giá trị trong khung dữ liệu cho là các chuỗi nên cần chuyển đổi NaN thành các chuỗi rỗng:

```
df.fillna('',axis=1,inplace=True)
df.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
0	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt	green tea	honey	salad	mineral water	salmon	antioxydant juice	frozen smoothie	spina
1	burgers	meatballs	eggs																
2	chutney																		
3	turkey	avocado																	
4	mineral water	milk	energy bar	whole wheat rice	green tea														

Sử dụng **TransactionEncoder** từ **mlxtend.preprocessing** để chuyển đổi khung dữ liệu trên thành khung dữ liệu với các giá trị True và False cho mọi mục trong giao dịch.

- Chuyển đổi khung dữ liệu thành một danh sách các danh sách trong đó mỗi danh sách bên trong đại diện cho một giao dịch:

```
from mlxtend.preprocessing import TransactionEncoder
df_list = df.to_numpy().tolist()
df_list
dataset = list()
for i in range(len(df_list)) :
    item = list()
    for j in df_list[i] :
        if pd.notna(j):
            item.append(j)
    dataset.append(item)
```

- Sử dụng **TransactionEncoder** để chuyển đổi tập dữ liệu có tên biến thành một mảng có các giá trị True và False cho mỗi giao dịch:

```
te = TransactionEncoder()
te_array = te.fit(dataset).transform(dataset)
te_array
```

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

- Chuyển mảng trên thành khung dữ liệu với tên cột là tên của tất cả các mục:

```
final_df = pd.DataFrame(te_array,columns=te.columns_)
final_df
```

	asparagus	almonds	antioxydant juice	asparagus	avocado	babies food	bacon	barbecue sauce	black tea	...	turkey	vegetables mix	water spray	white wine	whole weat flour	whole wheat pasta	whol whea ric
0	False	False	True	True	False	True	False	False	False	False	...	False	True	False	False	True	False	Fals
1	True	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	Fals
2	True	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	Fals
3	True	False	False	False	False	True	False	False	False	False	...	True	False	False	False	False	False	Fals
4	True	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	Tru
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
7496	True	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	Fals
7497	True	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	Fals
7498	True	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	Fals
7499	True	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	Fals
7500	True	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	Fals

7501 rows × 121 columns

Xóa cột đầu tiên vì nó không chứa bất kỳ thông tin nào:

```
final_df.drop(columns=[''],axis=1,inplace=True)
final_df
```

	asparagus	almonds	antioxydant juice	asparagus	avocado	babies food	bacon	barbecue sauce	black tea	blueberries	...	turkey	vegetables mix	water spray	white wine	whole weat flour	whole wheat pasta
0	False	True	True	False	True	False	False	False	False	False	...	False	True	False	False	True	False
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
3	False	False	False	False	True	False	False	False	False	False	...	True	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7496	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
7497	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
7498	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
7499	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
7500	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False

7501 rows × 120 columns

Sau các thao tác trên ta thu được khung dữ liệu cuối cùng là **final_df** với tên các cột là tên các mục và True, False cho biết liệu chúng có hiện diện trong giao dịch đó hay không. Mỗi hàng trong khung dữ liệu đại diện cho một giao dịch.

*** ARM**

Sử dụng thuật toán **apriori** từ thư viện **mlxtend** để trích xuất các mục hoặc nhóm mục có **mức hỗ trợ (support)** lớn hơn mức hỗ trợ tối thiểu 0,01:

```
from mlxtend.frequent_patterns import apriori
frequent_itemsets_ap = apriori(final_df, min_support= 0.01 , use_colnames=True)
frequent_itemsets_ap
```

	support	itemsets
0	0.020397	(almonds)
1	0.033329	(avocado)
2	0.010799	(barbecue sauce)
3	0.014265	(black tea)
4	0.011465	(body spray)
...	...	...
252	0.011065	(milk, ground beef, mineral water)
253	0.017064	(ground beef, spaghetti, mineral water)
254	0.015731	(milk, spaghetti, mineral water)
255	0.010265	(spaghetti, olive oil, mineral water)
256	0.011465	(pancakes, spaghetti, mineral water)

257 rows × 2 columns

Kết quả ta thu được một khung dữ liệu với 2 cột thể hiện mức độ lan truyền của tập phổ biến này trong cơ sở dữ liệu. Trong bảng trên ta có thể thấy **quả hạnh (almonds)** có mức độ lan truyền là khoảng 0,02; tập **ground beef, spaghetti, mineral water (thịt bò xay, mỳ ý, nước khoáng)** có mức độ lan truyền là khoảng 0,017....

Sử dụng **association_rules** từ **mlxtend.frequent_patterns** để tìm các quy tắc kết hợp giữa các mục/nhóm mặt hàng có mức hỗ trợ lớn hơn mức hỗ trợ tối thiểu. Sử dụng số liệu làm độ tin cậy và **min_threshold (ngưỡng tối thiểu)** để lọc ra các quy tắc kết hợp dựa trên các thông số này.

```
from mlxtend.frequent_patterns import association_rules
rules_ap = association_rules(frequent_itemsets_ap, metric="confidence", min_threshold=0.2)
rules_ap
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(avocado)	(mineral water)	0.033329	0.238368	0.011598	0.348000	1.459926	0.003654	1.168147
1	(burgers)	(eggs)	0.087188	0.179709	0.028796	0.330275	1.837830	0.013128	1.224818
2	(burgers)	(french fries)	0.087188	0.170911	0.021997	0.252294	1.476173	0.007096	1.108844
3	(burgers)	(green tea)	0.087188	0.132116	0.017464	0.200306	1.516139	0.005945	1.085270
4	(burgers)	(milk)	0.087188	0.129583	0.017864	0.204893	1.581175	0.006566	1.094717
...	...	...	...	...	...	...	...	...	...
157	(spaghetti, mineral water)	(milk)	0.059725	0.129583	0.015731	0.263393	2.032623	0.007992	1.181657
158	(spaghetti, olive oil)	(mineral water)	0.022930	0.238368	0.010265	0.447674	1.878079	0.004799	1.378954
159	(olive oil, mineral water)	(spaghetti)	0.027596	0.174110	0.010265	0.371981	2.136468	0.005460	1.315071
160	(pancakes, spaghetti)	(mineral water)	0.025197	0.238368	0.011465	0.455026	1.908923	0.005459	1.397557
161	(pancakes, mineral water)	(spaghetti)	0.033729	0.174110	0.011465	0.339921	1.952333	0.005593	1.251198

162 rows × 9 columns

Kết quả ta thu được một khung dữ liệu với 2 cột đầu thể hiện mối quan hệ X → Y, khả năng mà khách hàng mua sản phẩm X cũng sẽ mua kèm với sản phẩm Y với mức độ lan truyền của X, Y được thể hiện ở 2 cột tiếp theo; 3 cột **support, confidence, lift** lần lượt thể hiện mức độ lan truyền, tần suất tìm thấy và khả năng xảy ra nếu khách hàng mua X cũng sẽ mua Y; cột **conviction** cho thấy độ tin cậy của quy tắc X → Y.

Xếp các giá trị theo thứ tự giảm dần dựa trên **mức tăng (lift)** chỉ số:

```
result = pd.DataFrame(rules_ap)
result.sort_values(by='lift',inplace=True,ascending=False)
result
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
75	(herb & pepper)	(ground beef)	0.049460	0.098254	0.015998	0.323450	3.291994	0.011138	1.332860
154	(spaghetti, mineral water)	(ground beef)	0.059725	0.098254	0.017064	0.285714	2.907928	0.011196	1.262445
69	(tomatoes)	(frozen vegetables)	0.068391	0.095321	0.016131	0.235867	2.474464	0.009612	1.183930
67	(shrimp)	(frozen vegetables)	0.071457	0.095321	0.016664	0.233209	2.446574	0.009853	1.179825
143	(milk, mineral water)	(frozen vegetables)	0.047994	0.095321	0.011065	0.230556	2.418737	0.006490	1.175757
...	...	...	...	...	...	...	...	...	...
74	(green tea)	(spaghetti)	0.132116	0.174110	0.026530	0.200807	1.153335	0.003527	1.033405
46	(ground beef)	(eggs)	0.098254	0.179709	0.019997	0.203528	1.132539	0.002340	1.029905
19	(chocolate)	(eggs)	0.163845	0.179709	0.033196	0.202604	1.127397	0.003751	1.028711
73	(green tea)	(mineral water)	0.132116	0.238368	0.031063	0.235116	0.986357	-0.000430	0.995748
57	(escalope)	(mineral water)	0.079323	0.238368	0.017064	0.215126	0.902495	-0.001844	0.970387

162 rows × 9 columns

Khung dữ liệu **'result'** cuối cùng bao gồm các quy tắc kết hợp cuối cùng dựa trên dữ liệu đã cho.

Một số nhận xét từ kết quả cuối cùng:

- **Thịt bò xay (ground beef)** có khả năng được mua cùng với **thảo mộc và hạt tiêu (herb & pepper)** cao hơn gấp 3,2 lần so với mua chính nó.
- Những người mua **mì ý và nước khoáng (spaghetti, mineral water)** thường mua **thịt bò xay (ground beef)** cùng với nó.
- Những người mua **chocolate** cũng mua **trứng (eggs)** cùng nhau thay vì chỉ mua riêng lẻ....

4. Tổng kết

Phân tích giỏ thị trường là ví dụ điển hình nhất của khai thác liên kết. Dữ liệu được thu thập bằng máy quét mã vạch ở hầu hết các siêu thị. Cơ sở dữ liệu này, được gọi là cơ sở dữ liệu "giỏ thị trường", bao gồm một số lượng lớn các bản ghi về các giao dịch trong quá khứ. Một bản ghi duy nhất liệt kê tất cả các mặt hàng được khách hàng mua trong một lần bán hàng. Biết được nhóm nào nghiêng về nhóm mặt hàng nào cho phép các cửa hàng này tự do điều chỉnh bố cục cửa hàng và danh mục cửa hàng để đặt các mặt hàng tương ứng với nhau một cách tối ưu.