BÁO CÁO

Thực hành Khai thác dữ liệu – Tuần 05

MSSV – Họ tên : 20280071 – Nguyễn Tiến Phong

Thuật toán Vertical Apriori :

Các bước thực hiện :

- **Đọc dữ liệu và chuyển dữ liệu từ dạng horizontal thành vertical**

	0	1	2	3	4	5	6
0	1	Wine	Chips	Bread	Butter	Milk	Apple
1	2	Wine	NaN	Bread	Butter	Milk	NaN
2	3	NaN	NaN	Bread	Butter	Milk	NaN
3	4	NaN	Chips	NaN	Butter	NaN	Apple
4	5	Wine	Chips	Bread	Butter	Milk	Apple
5	6	Wine	Chips	NaN	NaN	Milk	NaN
6	7	Wine	Chips	Bread	Butter	NaN	Apple
7	8	Wine	Chips	NaN	NaN	Milk	NaN
8	9	Wine	NaN	Bread	NaN	NaN	Apple
9	10	Wine	NaN	Bread	Butter	Milk	NaN
10	11	NaN	Chips	Bread	Butter	NaN	Apple
11	12	Wine	NaN	NaN	Butter	Milk	Apple
12	13	Wine	Chips	Bread	Butter	Milk	NaN
13	14	Wine	NaN	Bread	NaN	Milk	Apple
14	15	Wine	NaN	Bread	Butter	Milk	Apple
15	16	Wine	Chips	Bread	Butter	Milk	Apple
16	17	NaN	Chips	Bread	Butter	Milk	Apple
17	18	NaN	Chips	NaN	Butter	Milk	Apple
18	19	Wine	Chips	Bread	Butter	Milk	Apple
19	20	Wine	NaN	Bread	Butter	Milk	Apple
20	21	Wine	Chips	Bread	NaN	Milk	Apple
21	22	NaN	Chips	Bread	NaN	Milk	NaN

```
Wine : [1, 2, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 19, 20, 21]
Chips : [1, 4, 5, 6, 7, 8, 11, 13, 16, 17, 18, 19, 21, 22]
Bread : [1, 2, 3, 5, 7, 9, 10, 11, 13, 14, 15, 16, 17, 19, 20, 21, 22]
Butter : [1, 2, 3, 4, 5, 7, 10, 11, 12, 13, 15, 16, 17, 18, 19, 20]
Milk : [1, 2, 3, 5, 6, 8, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22]
Apple : [1, 4, 5, 7, 9, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21]
```

- **Tính Support của các frequent_itemsets :**

```python
# Tính support cho frequent_itemsets
def calculate_support(frequent_itemsets, data):
    count = 0
    support = {}
    for itemset in frequent_itemsets:
        cnt = {}
        for i in itemset:
            cnt[i] = data[i] # Lưu các item và values vào cnt
        count = count_item(cnt) # Đếm số lượng values giống nhau giữa các item (tần suất cùng xuất hiện của các item)
        support[itemset] = count / len(df[0]) # Tính support của itemset
    return support
```

✓ 0.0s

Trong đó, hàm count_item để đếm tần suất xuất hiện của itemset:

```python
# Đếm các phần tử giống nhau của array trong dict
def count_item(cnt):
    common_elements = set()

    for array_name, array_data in cnt.items():
        if not common_elements:
            common_elements.update(array_data)
        else:
            common_elements.intersection_update(array_data)

    num_common_elements = len(common_elements)
    return num_common_elements
```

✓ 0.0s

- **Tìm các frequent_itemsets có kích thước 1**

```python
# Tìm các frequent_itemsets có kích thước 1 dựa trên min_support
def find_frequent_itemsets(data, min_support):
    frequent_itemsets = {}
    for item in data.keys():
        support = len(data[item]) / len(df[0])
        if support >= min_support:
            frequent_itemsets[(item,)] = support
    return frequent_itemsets
```

✓ 0.1s

- **Sinh các ứng viên có kích thước lớn hơn từ các frequent_itemsets trước đó**

```python
# Sinh các ứng viên cho frequent_itemsets có kích thước lớn hơn từ các frequent_itemsets trước đó
def generate_candidate_itemsets(pre_itemsets):
    candidate_itemsets = set()
    for itemset1 in pre_itemsets:
        for itemset2 in pre_itemsets:
            if itemset1 != itemset2 :
                new_itemset = tuple(sorted(set(itemset1) | set(itemset2))) # Kết hợp các itemset lại
                if len(new_itemset) == len(itemset1) + 1: # Kích thước được tăng lên 1
                    candidate_itemsets.add(new_itemset)
    return candidate_itemsets
```

✓ 0.1s

- **Triển khai thuật toán Vertical Apriori**

```python
# Triển khai thuật toán Vertical Apriori
def vertical_apriori(data, min_support, max_length):
    # Tìm các ứng viên có kích thước bằng max_length
    frequent_itemsets = find_frequent_itemsets(data, min_support)
    k = 0
    k = max(len(itemset) for itemset in frequent_itemsets)
    while k < max_length:
        frequent_itemsets = generate_candidate_itemsets(frequent_itemsets)
        k = max(len(itemset) for itemset in frequent_itemsets)

    # Tính support và lọc các support nhỏ hơn ngưỡng min_support
    support_itemsets = calculate_support(frequent_itemsets, data)
    support = {}
    frequent_itemset = []
    for itemset, supp in support_itemsets.items():
        if supp > min_support:
            frequent_itemset.append(itemset)
            support[itemset] = supp
    return support
```

✓ 0.1s

- **In ra kết quả**

```
# In kết quả
min_support = 0.3
max_length = 3
support = vertical_apriori(vertical_df, min_support, max_length)
data = pd.DataFrame()
data['Frequent_Itemset'] = support.keys()
data['Support'] = support.values()
data
```

✓ 0.1s

So sánh với kết quả khi sử dụng thư viện pyECLAT : <u>Kết quả tương tự nhau</u>

	Frequent_Itemset	Support
0	(Apple, Bread, Milk)	0.409091
1	(Bread, Butter, Milk)	0.500000
2	(Butter, Milk, Wine)	0.454545
3	(Bread, Chips, Wine)	0.318182
4	(Apple, Chips, Milk)	0.318182
5	(Apple, Bread, Butter)	0.409091
6	(Apple, Milk, Wine)	0.409091
7	(Apple, Butter, Wine)	0.363636
8	(Bread, Chips, Milk)	0.363636
9	(Bread, Milk, Wine)	0.500000
10	(Apple, Butter, Chips)	0.409091
11	(Apple, Bread, Wine)	0.454545
12	(Bread, Butter, Wine)	0.454545
13	(Apple, Butter, Milk)	0.409091
14	(Chips, Milk, Wine)	0.363636
15	(Butter, Chips, Milk)	0.318182
16	(Apple, Bread, Chips)	0.363636
17	(Bread, Butter, Chips)	0.363636

```
eclat = ECLAT(data=df, verbose=True)
frequent_itemsets, support = eclat.fit(min_support=0.3, min_combination=3, separator=' & ', verbose=True)

support
```

✓ 0.7s

```
100%|██████████| 28/28 [00:00<00:00, 139.96it/s]
100%|██████████| 28/28 [00:00<?, ?it/s]
100%|██████████| 28/28 [00:00<00:00, 1025.25it/s]
Combination 3 by 3
20it [00:00, 77.24it/s]

{'Milk & Chips & Bread': 0.36363636363636365,
 'Milk & Chips & Butter': 0.3181818181818182,
 'Milk & Chips & Apple': 0.3181818181818182,
 'Milk & Chips & Wine': 0.36363636363636365,
 'Milk & Bread & Butter': 0.5,
 'Milk & Bread & Apple': 0.4090909090909091,
 'Milk & Bread & Wine': 0.5,
 'Milk & Butter & Apple': 0.4090909090909091,
 'Milk & Butter & Wine': 0.45454545454545453,
 'Milk & Apple & Wine': 0.4090909090909091,
 'Chips & Bread & Butter': 0.36363636363636365,
 'Chips & Bread & Apple': 0.36363636363636365,
 'Chips & Bread & Wine': 0.3181818181818182,
 'Chips & Butter & Apple': 0.4090909090909091,
 'Bread & Butter & Apple': 0.4090909090909091,
 'Bread & Butter & Wine': 0.45454545454545453,
 'Bread & Apple & Wine': 0.45454545454545453,
 'Butter & Apple & Wine': 0.36363636363636365}
```