Lab 8

TEB3123: Machine Learning

Online Retail Market Basket Analysis

Name	**ID**	**Course**
Cheng Pin-Jie	21000548	Computer Science

Mr Abdul Muiz Fayyaz

1.0 Data Understanding

The dataset is in excel format which is downloaded from:

https://archive.ics.uci.edu/ml/machine-learning-databases/00352/Online%20Retail.xlsx

The dataset is loaded by using:

```
# Import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

# Load the dataset
df = pd.read_excel("Online Retail.xlsx")

# Display the first few rows of the dataset
print(df.head())
```

```
  InvoiceNo StockCode                          Description  Quantity  \
0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6
1    536365     71053                  WHITE METAL LANTERN         6
2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8
3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6
4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6

          InvoiceDate  UnitPrice  CustomerID         Country
0 2010-12-01 08:26:00       2.55     17850.0  United Kingdom
1 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
2 2010-12-01 08:26:00       2.75     17850.0  United Kingdom
3 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
4 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
```

2.0 Data Preparation

The dataset needs to be further process for market basket analysis. Thus, data cleaning, including removing extra spaces in product descriptions, dropping missing value row and filtering datasets.

```
# Remove spaces in the descriptions and drop rows with missing
invoice numbers
df['Description'] = df['Description'].str.strip()
df.dropna(axis=0, subset=['InvoiceNo'], inplace=True)
# Filter out records for transactions in France
df = df[df['Country'] == 'France']
```

Before passing the data to the algorithm, encoding is needed to convert values into binary values.

```python
def encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1
```

A matrix is then created to represent ‘Trasaction vs Product”.

```python
basket = (df.groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))
basket_sets = basket.applymap(encode_units)

basket_sets
```

InvoiceNo	# 10 COLOUR SPACEBOY P...	# 12 COLOURED PARTY BA...	# 12 EGG HOUSE PAINTED...	# 12 MESSAGE CARDS WIT...	# 12 PEN
536370	0	0	0	0	
536852	0	0	0	0	
536974	0	0	0	0	
537065	0	0	0	0	
537463	0	0	0	0	
537468	1	0	0	0	
537693	0	0	0	0	
537897	0	0	0	0	
537967	0	0	0	0	
538008	0	0	0	0	

461 rows x 1,564 cols 10 per page Page 1 of 47

3.0 Modeling

The data is now ready to implement association rules. First, *apriori* is used to find the frequent itemsets. Then, association rules are created in the format of *“If customer buys A, they also buy B”*

```python
frequent_itemsets = apriori(basket_sets, min_support=0.07,
use_colnames=True)

rules = association_rules(frequent_itemsets, metric="lift",
min_threshold=1)
```

```
                                         antecedents  \
0                      (ALARM CLOCK BAKELIKE GREEN)
1                                         (POSTAGE)
2                                         (POSTAGE)
3                       (ALARM CLOCK BAKELIKE PINK)
4                        (ALARM CLOCK BAKELIKE RED)
..                                              ...
91  (SET/6 RED SPOTTY PAPER PLATES, SET/20 RED RET...
92  (SET/6 RED SPOTTY PAPER CUPS, SET/20 RED RETRO...
93                   (SET/6 RED SPOTTY PAPER PLATES)
94                     (SET/6 RED SPOTTY PAPER CUPS)
95              (SET/20 RED RETROSPOT PAPER NAPKINS)

                                         consequents  antecedent support
0                                         (POSTAGE)            0.082430
1                      (ALARM CLOCK BAKELIKE GREEN)            0.650759
2                       (ALARM CLOCK BAKELIKE PINK)            0.650759
3                                         (POSTAGE)            0.086768
4                                         (POSTAGE)            0.080260
..                                              ...                 ...
91                     (SET/6 RED SPOTTY PAPER CUPS)            0.086768
92                   (SET/6 RED SPOTTY PAPER PLATES)            0.086768
93  (SET/6 RED SPOTTY PAPER CUPS, SET/20 RED RETRO...            0.108460
94  (SET/6 RED SPOTTY PAPER PLATES, SET/20 RED RET...            0.117137
95  (SET/6 RED SPOTTY PAPER PLATES, SET/6 RED SPOT...            0.112798
...
94  0.074435    3.287636       0.996598  0.709091    0.695830    0.848611
95  0.072854    3.583514       0.970660  0.639344    0.720944    0.781250

[96 rows x 14 columns]
```

The rules generated are then further filtered for:

- Life ⩾ 6
- Confidence ⩾ 0.8

It Is because we want rules that has a strong positive association (lift) and itemsets that appear at least 80% of the time (confidence).

```
filtered_rules = rules[(rules['lift'] >= 6) &
(rules['confidence'] >= 0.8)]

print(filtered_rules)
```

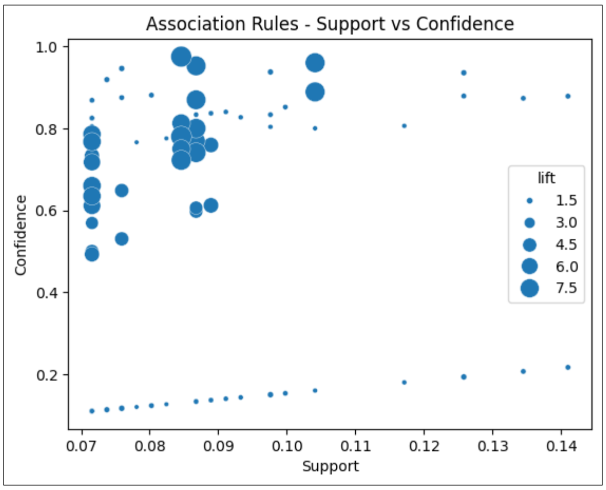
```
                                     antecedents  \
58                    (SET/6 RED SPOTTY PAPER PLATES)
59                      (SET/6 RED SPOTTY PAPER CUPS)
85           (SET/6 RED SPOTTY PAPER PLATES, POSTAGE)
86             (SET/6 RED SPOTTY PAPER CUPS, POSTAGE)
90  (SET/6 RED SPOTTY PAPER PLATES, SET/6 RED SPOT...
91  (SET/6 RED SPOTTY PAPER PLATES, SET/20 RED RET...
92  (SET/6 RED SPOTTY PAPER CUPS, SET/20 RED RETRO...

                         consequents  antecedent support  \
58         (SET/6 RED SPOTTY PAPER CUPS)            0.108460
59       (SET/6 RED SPOTTY PAPER PLATES)            0.117137
85         (SET/6 RED SPOTTY PAPER CUPS)            0.091106
86       (SET/6 RED SPOTTY PAPER PLATES)            0.099783
90  (SET/20 RED RETROSPOT PAPER NAPKINS)            0.104121
91         (SET/6 RED SPOTTY PAPER CUPS)            0.086768
92       (SET/6 RED SPOTTY PAPER PLATES)            0.086768

    consequent support   support  confidence      lift  representativity  \
58            0.117137  0.104121    0.960000  8.195556               1.0
59            0.108460  0.104121    0.888889  8.195556               1.0
85            0.117137  0.086768    0.952381  8.130511               1.0
86            0.108460  0.086768    0.869565  8.017391               1.0
90            0.112798  0.084599    0.812500  7.203125               1.0
91            0.117137  0.084599    0.975000  8.323611               1.0
...
86  0.075945    6.835141       0.972289  0.714286  0.853697    0.834783
90  0.072854    4.731743       0.961259  0.639344  0.788661    0.781250
91  0.074435   35.314534       0.963457  0.709091  0.971683    0.848611
92  0.075188   35.661605       0.973202  0.764706  0.971959    0.877500
```

Lastly, a scatterplot is visualized to view the rules:

```python
sns.scatterplot(x='support', y='confidence', data=rules,
size='lift', sizes=(10, 200))
plt.title('Association Rules - Support vs Confidence')
plt.xlabel('Support')
plt.ylabel('Confidence')
plt.show()
```



4.0 Findings & Conclusion

Based on the *filtered_rules*, I can spot that there is a strong relationship among red spotty tableware in France. For example:

Transaction	Lift	Confidence
Red spotty paper plates -- Red spotty paper cups	8.2	0.96
Red spotty paper cups -- Red spotty paper plates	8.2	0.89
Red spotty paper cups -- Red spotty paper plates -- Red retrospot paper napkins	7.2	0.81

These are the top transaction in this online retail business located in France, which suggest that these items can be displayed together in online store layout. A "Party Package" can be created with promotion price to further boost the France market. Besides, business owner can maintain the stocks for all these items for future inventory management.

In conclusion, these association rules let us understand the customer behaviour and their preference when come into our online retail store.