```
import numpy as np
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules

# Loading the Data from the excel file
df = pd.read_excel('Online Retail.xlsx')
df.head(20)
```

```
   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

5     536365     22752         SET 7 BABUSHKA NESTING BOXES         2

6     536365     21730    GLASS STAR FROSTED T-LIGHT HOLDER         6

7     536366     22633               HAND WARMER UNION JACK         6

8     536366     22632            HAND WARMER RED POLKA DOT         6

9     536367     84879        ASSORTED COLOUR BIRD ORNAMENT        32

10    536367     22745           POPPY'S PLAYHOUSE BEDROOM         6

11    536367     22748            POPPY'S PLAYHOUSE KITCHEN         6

12    536367     22749    FELTCRAFT PRINCESS CHARLOTTE DOLL         8

13    536367     22310              IVORY KNITTED MUG COSY         6

14    536367     84969   BOX OF 6 ASSORTED COLOUR TEASPOONS         6

15    536367     22623        BOX OF VINTAGE JIGSAW BLOCKS         3

16    536367     22622       BOX OF VINTAGE ALPHABET BLOCKS         2

17    536367     21754            HOME BUILDING BLOCK WORD         3

18    536367     21755            LOVE BUILDING BLOCK WORD         3

19    536367     21777          RECIPE BOX WITH METAL HEART         4
```

```
           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
5  2010-12-01 08:26:00       7.65     17850.0  United Kingdom
6  2010-12-01 08:26:00       4.25     17850.0  United Kingdom
7  2010-12-01 08:28:00       1.85     17850.0  United Kingdom
8  2010-12-01 08:28:00       1.85     17850.0  United Kingdom
9  2010-12-01 08:34:00       1.69     13047.0  United Kingdom
10 2010-12-01 08:34:00       2.10     13047.0  United Kingdom
11 2010-12-01 08:34:00       2.10     13047.0  United Kingdom
12 2010-12-01 08:34:00       3.75     13047.0  United Kingdom
13 2010-12-01 08:34:00       1.65     13047.0  United Kingdom
14 2010-12-01 08:34:00       4.25     13047.0  United Kingdom
15 2010-12-01 08:34:00       4.95     13047.0  United Kingdom
16 2010-12-01 08:34:00       9.95     13047.0  United Kingdom
17 2010-12-01 08:34:00       5.95     13047.0  United Kingdom
18 2010-12-01 08:34:00       5.95     13047.0  United Kingdom
19 2010-12-01 08:34:00       7.95     13047.0  United Kingdom
```

```
df.columns
```

```
Index(['InvoiceNo', 'StockCode', 'Description', 'Quantity',
'InvoiceDate',
       'UnitPrice', 'CustomerID', 'Country'],
      dtype='object')
```

```
# Exploring the different regions of transactions
df.Country.unique()
```

```
array(['United Kingdom', 'France', 'Australia', 'Netherlands',
'Germany',
       'Norway', 'EIRE', 'Switzerland', 'Spain', 'Poland', 'Portugal',
       'Italy', 'Belgium', 'Lithuania', 'Japan', 'Iceland',
       'Channel Islands', 'Denmark', 'Cyprus', 'Sweden', 'Austria',
       'Israel', 'Finland', 'Bahrain', 'Greece', 'Hong Kong',
'Singapore',
       'Lebanon', 'United Arab Emirates', 'Saudi Arabia',
       'Czech Republic', 'Canada', 'Unspecified', 'Brazil', 'USA',
       'European Community', 'Malta', 'RSA'], dtype=object)
```

```
# whether Turkey is incluced
"Turkey" in df.Country.unique()
```

```
False
```

```
len_without_droppting = len(df)
# Stripping extra spaces in the description
df['Description'] = df['Description'].str.strip()

# Dropping the rows without any invoice number
# Invoice number is important because it represents the items
purchased
df.dropna(axis = 0, subset = ['InvoiceNo'], inplace = True)
df['InvoiceNo'] = df['InvoiceNo'].astype('str')

# Dropping all transactions which were done on credit
df = df[~df['InvoiceNo'].str.contains('C')]

df["Description"]
```

```
0          WHITE HANGING HEART T-LIGHT HOLDER
1                         WHITE METAL LANTERN
2              CREAM CUPID HEARTS COAT HANGER
3         KNITTED UNION FLAG HOT WATER BOTTLE
4              RED WOOLLY HOTTIE WHITE HEART.
                         ...
541904            PACK OF 20 SPACEBOY NAPKINS
541905            CHILDREN'S APRON DOLLY GIRL
541906           CHILDRENS CUTLERY DOLLY GIRL
541907        CHILDRENS CUTLERY CIRCUS PARADE
541908           BAKING SET 9 PIECE RETROSPOT
Name: Description, Length: 532621, dtype: object
```

```
len_after_dropping = len(df)
print("before dropping: ", len_without_droppting)
print("after dropping: ", len_after_dropping)
print("amount of dropped rows: ", len_without_droppting -
len_after_dropping)
```

```
before dropping:  541909
after dropping:  532621
amount of dropped rows:  9288
```

```
# Transactions done in France
basket_France = (df[df['Country'] =="France"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))

# Transactions done in the United Kingdom
basket_UK = (df[df['Country'] =="United Kingdom"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))

# Transactions done in Portugal
```

```
basket_Por = (df[df['Country'] =="Portugal"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))

basket_Sweden = (df[df['Country'] =="Sweden"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))
# Defining the hot encoding function to make the df suitable
# for the concerned libraries
def hot_encode(x):
    if(x<= 0):
        return 0
    if(x>= 1):
        return 1

# Encoding the dfsets
basket_encoded = basket_France.applymap(hot_encode)
basket_France = basket_encoded

basket_encoded = basket_UK.applymap(hot_encode)
basket_UK = basket_encoded

basket_encoded = basket_Por.applymap(hot_encode)
basket_Por = basket_encoded

basket_encoded = basket_Sweden.applymap(hot_encode)
basket_Sweden = basket_encoded

# Building the model
frq_items = apriori(basket_France, min_support = 0.05, use_colnames =
True)

# Collecting the inferred rules in a dataframe
rules = association_rules(frq_items, metric ="lift", min_threshold =
1)
rules = rules.sort_values(['confidence', 'lift'], ascending =[False,
False])
print(rules.head())

                                          antecedents  \
44                        (JUMBO BAG WOODLAND ANIMALS)
259  (RED TOADSTOOL LED NIGHT LIGHT, PLASTERS IN TI...
271  (RED TOADSTOOL LED NIGHT LIGHT, PLASTERS IN TI...
300  (SET/20 RED RETROSPOT PAPER NAPKINS, SET/6 RED...
301  (SET/20 RED RETROSPOT PAPER NAPKINS, SET/6 RED...

                        consequents  antecedent support  consequent
support  \
```

```
44                          (POSTAGE)           0.076531
0.765306
259                         (POSTAGE)           0.051020
0.765306
271                         (POSTAGE)           0.053571
0.765306
300  (SET/6 RED SPOTTY PAPER PLATES)            0.102041
0.127551
301    (SET/6 RED SPOTTY PAPER CUPS)            0.102041
0.137755

      support  confidence      lift  leverage  conviction
44   0.076531       1.000  1.306667  0.017961         inf
259  0.051020       1.000  1.306667  0.011974         inf
271  0.053571       1.000  1.306667  0.012573         inf
300  0.099490       0.975  7.644000  0.086474   34.897959
301  0.099490       0.975  7.077778  0.085433   34.489796
```

From the above output, it can be seen that paper cups and paper and plates are bought together in France. This is because the French have a culture of having a get-together with their friends and family atleast once a week. Also, since the French government has banned the use of plastic in the country, the people have to purchase the paper-based alternatives.

```
rules

                                              antecedents  \
44                        (JUMBO BAG WOODLAND ANIMALS)
259  (RED TOADSTOOL LED NIGHT LIGHT, PLASTERS IN TI...
271  (RED TOADSTOOL LED NIGHT LIGHT, PLASTERS IN TI...
300  (SET/20 RED RETROSPOT PAPER NAPKINS, SET/6 RED...
301  (SET/20 RED RETROSPOT PAPER NAPKINS, SET/6 RED...
..                                                 ...
36                                           (POSTAGE)
27                                           (POSTAGE)
96                                           (POSTAGE)
226                                          (POSTAGE)
215                                          (POSTAGE)

                                             consequents  antecedent
support  \
44                                           (POSTAGE)
0.076531
259                                          (POSTAGE)
0.051020
271                                          (POSTAGE)
0.053571
300                    (SET/6 RED SPOTTY PAPER PLATES)
0.102041
301                      (SET/6 RED SPOTTY PAPER CUPS)
0.102041
```

```
..                                                  ...
...
36                            (JAM MAKING SET PRINTED)
0.765306
27                   (CIRCUS PARADE CHILDRENS EGG CUP)
0.765306
96                                        (PARTY BUNTING)
0.765306
226      (LUNCH BAG WOODLAND, LUNCH BAG RED RETROSPOT)
0.765306
215  (LUNCH BAG APPLE DESIGN, LUNCH BAG SPACEBOY DE...
0.765306

     consequent support   support  confidence      lift  leverage
conviction
44             0.765306  0.076531    1.000000  1.306667  0.017961
inf
259            0.765306  0.051020    1.000000  1.306667  0.011974
inf
271            0.765306  0.053571    1.000000  1.306667  0.012573
inf
300            0.127551  0.099490    0.975000  7.644000  0.086474
34.897959
301            0.137755  0.099490    0.975000  7.077778  0.085433
34.489796
..                  ...       ...         ...       ...       ...
...
36             0.053571  0.051020    0.066667  1.244444  0.010022
1.014031
27             0.056122  0.051020    0.066667  1.187879  0.008070
1.011297
96             0.056122  0.051020    0.066667  1.187879  0.008070
1.011297
226            0.056122  0.051020    0.066667  1.187879  0.008070
1.011297
215            0.063776  0.051020    0.066667  1.045333  0.002213
1.003098

[348 rows x 9 columns]
```

```
frq_items = apriori(basket_UK, min_support = 0.70, use_colnames =
True)
rules = association_rules(frq_items, metric ="lift", min_threshold =
1)
rules = rules.sort_values(['confidence', 'lift'], ascending =[False,
False])
print(rules.head())
```

```
frq_items = apriori(basket_Por, min_support = 0.05, use_colnames =
True)
```

```
rules = association_rules(frq_items, metric ="lift", min_threshold =
1)
rules = rules.sort_values(['confidence', 'lift'], ascending =[False,
False])
print(rules.head())

frq_items = apriori(basket_Sweden, min_support = 0.05, use_colnames =
True)
rules = association_rules(frq_items, metric ="lift", min_threshold =
1)
rules = rules.sort_values(['confidence', 'lift'], ascending =[False,
False])
print(rules.head())
```