Loading Necessary Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
```

Importing Dataset

In [2]:
```
#myretaildata = pd.read_excel('http://archive.ics.uci.edu/ml/machine-learning-databases/00352/Online%20Retail.xlsx')
myretaildata = pd.read_excel('Online Retail (1).xlsx')
myretaildata.head()
```

Out[2]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

In [3]:
```
myretaildata.shape
```

Out[3]:
```
(541909, 8)
```

In [4]:
```
myretaildata.describe().T
```

Out[4]:

	count	mean	std	min	25%	50%	75%	max
Quantity	541909.0	9.552250	218.081158	-80995.00	1.00	3.00	10.00	80995.0
UnitPrice	541909.0	4.611114	96.759853	-11062.06	1.25	2.08	4.13	38970.0
CustomerID	406829.0	15287.690570	1713.600303	12346.00	13953.00	15152.00	16791.00	18287.0

In [5]:
```
myretaildata.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype         
---  ------       --------------   -----         
 0   InvoiceNo    541909 non-null  object        
 1   StockCode    541909 non-null  object        
 2   Description  540455 non-null  object        
 3   Quantity     541909 non-null  int64         
 4   InvoiceDate  541909 non-null  datetime64[ns]
 5   UnitPrice    541909 non-null  float64       
 6   CustomerID   406829 non-null  float64       
 7   Country      541909 non-null  object        
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 33.1+ MB
```

Data Preparation

In [6]:
```
#Data Cleaning
myretaildata['Description'] = myretaildata['Description'].str.strip() #removes spaces from beginning and end
myretaildata.dropna(axis=0, subset=['InvoiceNo'], inplace=True) #removes duplicate invoice
myretaildata['InvoiceNo'] = myretaildata['InvoiceNo'].astype('str') #converting invoice number to be string
myretaildata = myretaildata[~myretaildata['InvoiceNo'].str.contains('C')] #remove the credit transactions
myretaildata.head()
```

Out[6]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

In [7]:
```
myretaildata['Country'].unique()
```

Out[7]:
```
array(['United Kingdom', 'France', 'Australia', 'Netherlands', 'Germany',
       'Norway', 'EIRE', 'Switzerland', 'Spain', 'Poland', 'Portugal',
       'Italy', 'Belgium', 'Lithuania', 'Japan', 'Iceland',
       'Channel Islands', 'Denmark', 'Cyprus', 'Sweden', 'Finland',
       'Austria', 'Bahrain', 'Israel', 'Greece', 'Hong Kong', 'Singapore',
       'Lebanon', 'United Arab Emirates', 'Saudi Arabia',
       'Czech Republic', 'Canada', 'Unspecified', 'Brazil', 'USA',
       'European Community', 'Malta', 'RSA'], dtype=object)
```

In [8]:
```
myretaildata['Country'].value_counts()
```

Out[8]:
```
United Kingdom          487622
Germany                   9042
France                    8408
EIRE                      7894
Spain                     2485
Netherlands               2363
Belgium                   2031
Switzerland               1967
Portugal                  1501
Australia                 1185
Norway                    1072
Italy                      758
Channel Islands            748
Finland                    685
Cyprus                     614
Sweden                     451
Unspecified                446
Austria                    398
Denmark                    380
Poland                     330
Japan                      321
Israel                     295
Hong Kong                  284
Singapore                  222
Iceland                    182
USA                        179
Canada                     151
Greece                     145
Malta                      112
United Arab Emirates        68
European Community          60
RSA                         58
Lebanon                     45
Lithuania                   35
Brazil                      32
Czech Republic              25
Bahrain                     18
Saudi Arabia                 9
Name: Country, dtype: int64
```

In [17]:
```
#Separating transactions for Germany
mybasket = (myretaildata[myretaildata['Country'] =="Germany"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))
```

In [18]:
```
#viewing transaction basket
mybasket.head()
```

Out[18]:

Description	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	12 PENCIL SMALL TUBE WOODLAND	12 PENCILS SMALL TUBE RED RETROSPOT	12 PENCILS SMALL TUBE SKULL	12 PENCILS TALL TUBE POSY	12 PENCILS TALL TUBE RED RETROSPOT	12 PENCILS TALL TUBE SKULLS	.
InvoiceNo											
536527	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	.
536840	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	.
536861	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	.
536967	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	.
536983	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	.

5 rows × 1695 columns

In [19]:
```
#viewing transaction basket shape
mybasket.shape
```

Out[19]:
```
(457, 1695)
```

In [20]:
```
#converting all positive vaues to 1 and everything else to 0
def my_encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1

my_basket_sets = mybasket.applymap(my_encode_units)
my_basket_sets.drop('POSTAGE', inplace=True, axis=1) #Remove "postage" as an item
```

Training the Model

In [21]:
```
#Generatig frequent itemsets
my_frequent_itemsets = apriori(my_basket_sets, min_support=0.07, use_colnames=True)
```

In [22]:
```
#generating rules
my_rules = association_rules(my_frequent_itemsets, metric="lift", min_threshold=1)
```

In [23]:
```
#viewing top 100 rules
my_rules.head(100)
```

Out[23]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(ROUND SNACK BOXES SET OF4 WOODLAND)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.245077	0.137856	0.074398	0.303571	2.202098	0.040613	1.237951
1	(PLASTERS IN TIN WOODLAND ANIMALS)	(ROUND SNACK BOXES SET OF4 WOODLAND)	0.137856	0.245077	0.074398	0.539683	2.202098	0.040613	1.640006
2	(ROUND SNACK BOXES SET OF 4 FRUITS)	(ROUND SNACK BOXES SET OF4 WOODLAND)	0.157549	0.245077	0.131291	0.833333	3.400298	0.092679	4.529540
3	(ROUND SNACK BOXES SET OF4 WOODLAND)	(ROUND SNACK BOXES SET OF 4 FRUITS)	0.245077	0.157549	0.131291	0.535714	3.400298	0.092679	1.814509
4	(ROUND SNACK BOXES SET OF4 WOODLAND)	(SPACEBOY LUNCH BOX)	0.245077	0.102845	0.070022	0.285714	2.778116	0.044817	1.256018
5	(SPACEBOY LUNCH BOX)	(ROUND SNACK BOXES SET OF4 WOODLAND)	0.102845	0.245077	0.070022	0.680851	2.778116	0.044817	2.365427

In [31]:
```
my_rules['antecedents'].value_counts()
```

Out[31]:
```
(ROUND SNACK BOXES SET OF4 WOODLAND)     3
(ROUND SNACK BOXES SET OF 4 FRUITS)      1
(SPACEBOY LUNCH BOX)                     1
(PLASTERS IN TIN WOODLAND ANIMALS)       1
Name: antecedents, dtype: int64
```

In [24]:
```
my_rules.to_csv('Testing_Apriori.csv')
```

In [25]:
```
my_rules.shape
```

Out[25]:
```
(6, 9)
```

Making Recommendations

In [27]:
```
my_basket_sets['ROUND SNACK BOXES SET OF4 WOODLAND'].sum()
```

Out[27]:
```
112
```

In [28]:
```
my_basket_sets['SPACEBOY LUNCH BOX'].sum()
```

Out[28]:
```
47
```

In [29]:
```
#Filtering rules based on condition
my_rules[ (my_rules['lift'] >= 3) &
        (my_rules['confidence'] >= 0.3) ]
```

Out[29]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
2	(ROUND SNACK BOXES SET OF 4 FRUITS)	(ROUND SNACK BOXES SET OF4 WOODLAND)	0.157549	0.245077	0.131291	0.833333	3.400298	0.092679	4.529540
3	(ROUND SNACK BOXES SET OF4 WOODLAND)	(ROUND SNACK BOXES SET OF 4 FRUITS)	0.245077	0.157549	0.131291	0.535714	3.400298	0.092679	1.814509

In [ ]:
```

```