Association Analysis- Apriori Implementation

Step 1: Import the packages

In [13]:

```
# Import numpy and pandas
import numpy as np
import pandas as pd
```

Step 2: Import the dataset

In [14]:

```
# https://www.kaggle.com/shazadudwadia/supermarket#GroceryStoreDataSet.csv
# Note: I have added column name "Products" before importing into python environment
df=pd.read_csv("GroceryStoreDataSet.csv")
df
```

Out[14]:

	Products
0	MILK,BREAD,BISCUIT
1	BREAD,MILK,BISCUIT,CORNFLAKES
2	BREAD,TEA,BOURNVITA
3	JAM,MAGGI,BREAD,MILK
4	MAGGI,TEA,BISCUIT
5	BREAD,TEA,BOURNVITA
6	MAGGI,TEA,CORNFLAKES
7	MAGGI,BREAD,TEA,BISCUIT
8	JAM,MAGGI,BREAD,TEA
9	BREAD,MILK
10	COFFEE,COCK,BISCUIT,CORNFLAKES
11	COFFEE,COCK,BISCUIT,CORNFLAKES
12	COFFEE,SUGER,BOURNVITA
13	BREAD,COFFEE,COCK
14	BREAD,SUGER,BISCUIT
15	COFFEE,SUGER,CORNFLAKES
16	BREAD,SUGER,BOURNVITA
17	BREAD,COFFEE,SUGER
18	BREAD,COFFEE,SUGER
19	TEA,MILK,COFFEE,CORNFLAKES

Step 3: Perform data pre-processing

In [15]:

```
# Consider column "products"
df.columns
```

Out[15]:

```
Index(['Products'], dtype='object')
```

In [16]:

```
# Convert the column in the dataset into list of lists.

data = list(df["Products"].apply(lambda x:x.split(',')))
data
```

Out[16]:

```
[['MILK', 'BREAD', 'BISCUIT'],
 ['BREAD', 'MILK', 'BISCUIT', 'CORNFLAKES'],
 ['BREAD', 'TEA', 'BOURNVITA'],
 ['JAM', 'MAGGI', 'BREAD', 'MILK'],
 ['MAGGI', 'TEA', 'BISCUIT'],
 ['BREAD', 'TEA', 'BOURNVITA'],
 ['MAGGI', 'TEA', 'CORNFLAKES'],
 ['MAGGI', 'BREAD', 'TEA', 'BISCUIT'],
 ['JAM', 'MAGGI', 'BREAD', 'TEA'],
 ['BREAD', 'MILK'],
 ['COFFEE', 'COCK', 'BISCUIT', 'CORNFLAKES'],
 ['COFFEE', 'COCK', 'BISCUIT', 'CORNFLAKES'],
 ['COFFEE', 'SUGER', 'BOURNVITA'],
 ['BREAD', 'COFFEE', 'COCK'],
 ['BREAD', 'SUGER', 'BISCUIT'],
 ['COFFEE', 'SUGER', 'CORNFLAKES'],
 ['BREAD', 'SUGER', 'BOURNVITA'],
 ['BREAD', 'COFFEE', 'SUGER'],
 ['BREAD', 'COFFEE', 'SUGER'],
 ['TEA', 'MILK', 'COFFEE', 'CORNFLAKES']]
```

In [17]:

```
# import package to preprocess the data.
from mlxtend.preprocessing import TransactionEncoder
```

In [18]:

```
# Transaction encoder converts the data into a form like "one hot encoding". Algorithm
 wants the data to be in this format.

te = TransactionEncoder()
te_data = te.fit(data).transform(data)
te_data

df = pd.DataFrame(te_data,columns=te.columns_)
df
```

Out[18]:

	BISCUIT	BOURNVITA	BREAD	COCK	COFFEE	CORNFLAKES	JAM	MAGGI	MILK	SI
0	True	False	True	False	False	False	False	False	True	
1	True	False	True	False	False	True	False	False	True	
2	False	True	True	False	False	False	False	False	False	
3	False	False	True	False	False	False	True	True	True	
4	True	False	False	False	False	False	False	True	False	
5	False	True	True	False	False	False	False	False	False	
6	False	False	False	False	False	True	False	True	False	
7	True	False	True	False	False	False	False	True	False	
8	False	False	True	False	False	False	True	True	False	
9	False	False	True	False	False	False	False	False	True	
10	True	False	False	True	True	True	False	False	False	
11	True	False	False	True	True	True	False	False	False	
12	False	True	False	False	True	False	False	False	False	
13	False	False	True	True	True	False	False	False	False	
14	True	False	True	False	False	False	False	False	False	
15	False	False	False	False	True	True	False	False	False	
16	False	True	True	False	False	False	False	False	False	
17	False	False	True	False	True	False	False	False	False	
18	False	False	True	False	True	False	False	False	False	
19	False	False	False	False	True	True	False	False	True	

Step 4: Find the frequent item sets.

In [19]:

```
# import the package to import apriori algorithm.
from mlxtend.frequent_patterns import apriori
```

In [20]:

```python
# Here we can define the minimum support expected by the user.
frequent_itemsets = apriori(df, min_support=0.2, use_colnames=True)
frequent_itemsets
```

Out[20]:

	support	itemsets
0	0.35	(BISCUIT)
1	0.20	(BOURNVITA)
2	0.65	(BREAD)
3	0.40	(COFFEE)
4	0.30	(CORNFLAKES)
5	0.25	(MAGGI)
6	0.25	(MILK)
7	0.30	(SUGER)
8	0.35	(TEA)
9	0.20	(BREAD, BISCUIT)
10	0.20	(BREAD, MILK)
11	0.20	(BREAD, SUGER)
12	0.20	(BREAD, TEA)
13	0.20	(COFFEE, CORNFLAKES)
14	0.20	(SUGER, COFFEE)
15	0.20	(TEA, MAGGI)

Step 5:Now get the frequent association rules from frequent itemsets.

In [21]:

```python
# Now get the association rules satisfing confidence defined by the user.
# import the package to find association rules
from mlxtend.frequent_patterns import association_rules

association_rules(frequent_itemsets, metric="confidence", min_threshold=0.7)
```

Out[21]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	levera
0	(MILK)	(BREAD)	0.25	0.65	0.2	0.8	1.230769	0.03
1	(MAGGI)	(TEA)	0.25	0.35	0.2	0.8	2.285714	0.11

The frequent association rules are:

{Milk}-> {Bread} [s=0.2, c=0.8]

{Maggi}-> {Tea} [s=o.2, c=0.8]

In [ ]: