In [2]:
```
!pip install mlxtend
```

```
Collecting mlxtend
  Downloading mlxtend-0.23.1-py3-none-any.whl (1.4 MB)
     ---------------------------------------- 1.4/1.4 MB 1.7 MB/s eta 0:00:00
Requirement already satisfied: matplotlib>=3.0.0 in c:\users\admin\anaconda3\l
ib\site-packages (from mlxtend) (3.5.2)
Requirement already satisfied: joblib>=0.13.2 in c:\users\admin\anaconda3\lib
\site-packages (from mlxtend) (1.1.0)
Requirement already satisfied: scikit-learn>=1.0.2 in c:\users\admin\anaconda3
\lib\site-packages (from mlxtend) (1.0.2)
Requirement already satisfied: scipy>=1.2.1 in c:\users\admin\anaconda3\lib\si
te-packages (from mlxtend) (1.9.1)
Requirement already satisfied: numpy>=1.16.2 in c:\users\admin\anaconda3\lib\s
ite-packages (from mlxtend) (1.21.5)
Requirement already satisfied: pandas>=0.24.2 in c:\users\admin\anaconda3\lib
\site-packages (from mlxtend) (1.4.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\admin\anaconda3\l
ib\site-packages (from matplotlib>=3.0.0->mlxtend) (1.4.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\admin\anaconda
3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.8.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\admin\anaconda3\lib\s
ite-packages (from matplotlib>=3.0.0->mlxtend) (9.2.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\admin\anaconda3\li
b\site-packages (from matplotlib>=3.0.0->mlxtend) (3.0.9)
Requirement already satisfied: cycler>=0.10 in c:\users\admin\anaconda3\lib\si
te-packages (from matplotlib>=3.0.0->mlxtend) (0.11.0)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\anaconda3\lib
\site-packages (from matplotlib>=3.0.0->mlxtend) (21.3)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\admin\anaconda3\l
ib\site-packages (from matplotlib>=3.0.0->mlxtend) (4.25.0)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\anaconda3\lib\si
te-packages (from pandas>=0.24.2->mlxtend) (2022.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\admin\anaconda
3\lib\site-packages (from scikit-learn>=1.0.2->mlxtend) (2.2.0)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\lib\site-p
ackages (from python-dateutil>=2.7->matplotlib>=3.0.0->mlxtend) (1.16.0)
Installing collected packages: mlxtend
Successfully installed mlxtend-0.23.1
```

In [20]:
```
import csv
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules
import pandas as pd
import numpy as np
```

In [8]:
```
data=[]
with open('../admin/Desktop/TE IT/Market_Basket_Optimisation.csv') as file:
    reader = csv.reader(file, delimiter=',')
    for row in reader:
        data +=[row]
```

In [9]:
```
data[1:10] #List of list
```

Out[9]:
```
[['burgers', 'meatballs', 'eggs'],
 ['chutney'],
 ['turkey', 'avocado'],
 ['mineral water', 'milk', 'energy bar', 'whole wheat rice', 'green tea'],
 ['low fat yogurt'],
 ['whole wheat pasta', 'french fries'],
 ['soup', 'light cream', 'shallot'],
 ['frozen vegetables', 'spaghetti', 'green tea'],
 ['french fries']]
```

In [10]:
```
len(data)
```

Out[10]:
```
7501
```

In [11]:
```
te = TransactionEncoder()
x = te.fit_transform(data)
```

In [12]:
```
x
```

Out[12]:
```
array([[False,  True,  True, ...,  True, False, False],
       [False, False, False, ..., False, False, False],
       [False, False, False, ..., False, False, False],
       ...,
       [False, False, False, ..., False, False, False],
       [False, False, False, ..., False, False, False],
       [False, False, False, ..., False,  True, False]])
```

In [13]:
```
te.columns_
```

```
 'sparkling water',
 'spinach',
 'strawberries',
 'strong cheese',
 'tea',
 'tomato juice',
 'tomato sauce',
 'tomatoes',
 'toothpaste',
 'turkey',
 'vegetables mix',
 'water spray',
 'white wine',
 'whole weat flour',
 'whole wheat pasta',
 'whole wheat rice',
 'yams',
 'yogurt cake',
 'zucchini']
```

In [21]:
```
df = pd.DataFrame(x, columns=te.columns_)
```

In [22]:
```
df
```

Out[22]:

	asparagus	almonds	antioxydant juice	asparagus	avocado	babies food	bacon	barbecue sauce	black tea	bl
0	False	True	True	False	True	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	True	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	
7496	False	False	False	False	False	False	False	False	False	
7497	False	False	False	False	False	False	False	False	False	
7498	False	False	False	False	False	False	False	False	False	
7499	False	False	False	False	False	False	False	False	False	
7500	False	False	False	False	False	False	False	False	False	

7501 rows × 120 columns

In [23]:
```
# Finding frequent itemsets
freq_itemset = apriori(df, min_support=0.01, use_colnames=True)
```

In [24]:
```
freq_itemset
```

Out[24]:

	support	itemsets
0	0.020397	(almonds)
1	0.033329	(avocado)
2	0.010799	(barbecue sauce)
3	0.014265	(black tea)
4	0.011465	(body spray)
...	...	...
252	0.011065	(milk, ground beef, mineral water)
253	0.017064	(spaghetti, ground beef, mineral water)
254	0.015731	(milk, spaghetti, mineral water)
255	0.010265	(spaghetti, mineral water, olive oil)
256	0.011465	(pancakes, spaghetti, mineral water)

257 rows × 2 columns

In [25]:
```
#Find the rules
rules = association_rules(freq_itemset, metric='confidence', min_threshold=0.10
```

In [26]:
```
rules = rules[['antecedents','consequents','support','confidence']]
rules
```

Out[26]:

	antecedents	consequents	support	confidence
0	(avocado)	(mineral water)	0.011598	0.348000
1	(cake)	(burgers)	0.011465	0.141447
2	(burgers)	(cake)	0.011465	0.131498
3	(burgers)	(chocolate)	0.017064	0.195719
4	(chocolate)	(burgers)	0.017064	0.104150
...	...	...	...	...
315	(olive oil)	(spaghetti, mineral water)	0.010265	0.155870
316	(pancakes, spaghetti)	(mineral water)	0.011465	0.455026
317	(pancakes, mineral water)	(spaghetti)	0.011465	0.339921
318	(spaghetti, mineral water)	(pancakes)	0.011465	0.191964
319	(pancakes)	(spaghetti, mineral water)	0.011465	0.120617

320 rows × 4 columns

In [27]:
```
rules[rules['antecedents'] == {'cake'}]['consequents']
```

Out[27]:
```
1              (burgers)
25           (chocolate)
27                (eggs)
29         (french fries)
30    (frozen vegetables)
32           (green tea)
35                (milk)
36        (mineral water)
39             (pancakes)
40           (spaghetti)
Name: consequents, dtype: object
```

In [ ]:
```

```