In [3]:
```
!pip install mlxtend
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: mlxtend in /ext/venvs/cocalc/lib/python3.12/site-packages (0.23.4)
Requirement already satisfied: scipy>=1.2.1 in /ext/venvs/cocalc/lib/python3.12/site-packages (from mlxtend) (1.
16.1)
Requirement already satisfied: numpy>=1.16.2 in /ext/venvs/cocalc/lib/python3.12/site-packages (from mlxtend) (2
.2.6)
Requirement already satisfied: pandas>=0.24.2 in /ext/venvs/cocalc/lib/python3.12/site-packages (from mlxtend) (
2.3.1)
Requirement already satisfied: scikit-learn>=1.3.1 in /ext/venvs/cocalc/lib/python3.12/site-packages (from mlxte
nd) (1.6.1)
Requirement already satisfied: matplotlib>=3.0.0 in /ext/venvs/cocalc/lib/python3.12/site-packages (from mlxtend
) (3.10.3)
Requirement already satisfied: joblib>=0.13.2 in /ext/venvs/cocalc/lib/python3.12/site-packages (from mlxtend) (
1.5.1)
Requirement already satisfied: contourpy>=1.0.1 in /ext/venvs/cocalc/lib/python3.12/site-packages (from matplotl
ib>=3.0.0->mlxtend) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /ext/venvs/cocalc/lib/python3.12/site-packages (from matplotlib>=
3.0.0->mlxtend) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /ext/venvs/cocalc/lib/python3.12/site-packages (from matplot
lib>=3.0.0->mlxtend) (4.58.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /ext/venvs/cocalc/lib/python3.12/site-packages (from matplot
lib>=3.0.0->mlxtend) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /ext/venvs/cocalc/lib/python3.12/site-packages (from matplotli
b>=3.0.0->mlxtend) (24.2)
Requirement already satisfied: pillow>=8 in /ext/venvs/cocalc/lib/python3.12/site-packages (from matplotlib>=3.0
.0->mlxtend) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /ext/venvs/cocalc/lib/python3.12/site-packages (from matplotl
ib>=3.0.0->mlxtend) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in /ext/venvs/cocalc/lib/python3.12/site-packages (from matp
lotlib>=3.0.0->mlxtend) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /ext/venvs/cocalc/lib/python3.12/site-packages (from pandas>=0.24
.2->mlxtend) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /ext/venvs/cocalc/lib/python3.12/site-packages (from pandas>=0.
24.2->mlxtend) (2025.2)
Requirement already satisfied: six>=1.5 in /ext/venvs/cocalc/lib/python3.12/site-packages (from python-dateutil>
=2.7->matplotlib>=3.0.0->mlxtend) (1.17.0)
Requirement already satisfied: threadpoolctl>=3.1.0 in /ext/venvs/cocalc/lib/python3.12/site-packages (from scik
it-learn>=1.3.1->mlxtend) (3.6.0)
```

In [13]:
```
import pandas as pd
import csv
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori,association_rules
```

In [14]:
```
dataset=[]
with open("Market_Basket_Optimisation.csv") as file:
    reader=csv.reader(file,delimiter=',')
    for row in reader:
        dataset+=[row]
```

In [17]:
```
dataset[0:10]
```

Out[17]:
```
[['shrimp',
  'almonds',
  'avocado',
  'vegetables mix',
  'green grapes',
  'whole weat flour',
  'yams',
  'cottage cheese',
  'energy drink',
  'tomato juice',
  'low fat yogurt',
  'green tea',
  'honey',
  'salad',
  'mineral water',
  'salmon',
  'antioxydant juice',
  'frozen smoothie',
  'spinach',
  'olive oil'],
 ['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 [18]:
```
len(dataset)
```

Out[18]:
```
7501
```

In [20]:
```
te=TransactionEncoder()
x=te.fit_transform(dataset)
```

In [21]:
```
x
```

Out[21]:
```
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]], shape=(7501, 120))
```

In [36]:
```
df=pd.DataFrame(x,columns=te.columns_)
```

In [37]:
```
len(te.columns_)
```

Out[37]:
```
120
```

In [38]:
```
df.head()
```

Out[38]:

	asparagus	almonds	antioxydant juice	asparagus	avocado	babies food	bacon	barbecue sauce	black tea	blueberries	...	turkey	vegetables mix	wa sp
0	False	True	True	False	True	False	False	False	False	False	...	False	True	Fa
1	False	False	False	False	False	False	False	False	False	False	...	False	False	Fa
2	False	False	False	False	False	False	False	False	False	False	...	False	False	Fa
3	False	False	False	False	True	False	False	False	False	False	...	True	False	Fa
4	False	False	False	False	False	False	False	False	False	False	...	False	False	Fa

5 rows × 120 columns

In [46]:
```
#1.find requent itemsets
freq_itemset=apriori(df,min_support=0.1,use_colnames=True )
```

In [47]:
```
freq_itemset
```

Out[47]:

	support	itemsets
0	0.163845	(chocolate)
1	0.179709	(eggs)
2	0.170911	(french fries)
3	0.132116	(green tea)
4	0.129583	(milk)
5	0.238368	(mineral water)
6	0.174110	(spaghetti)

In [49]:
```
freq_itemset=apriori(df,min_support=0.01,use_colnames=True )
```

In [50]:
```
freq_itemset
```

Out[50]:

	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	(mineral water, ground beef, milk)
253	0.017064	(mineral water, ground beef, spaghetti)
254	0.015731	(mineral water, milk, spaghetti)
255	0.010265	(mineral water, spaghetti, olive oil)
256	0.011465	(mineral water, pancakes, spaghetti)

257 rows × 2 columns

In [56]:
```
#find the rules
rules=association_rules(freq_itemset,metric='confidence',min_threshold=0.30)
```

In [54]:
```
rules=rules[['antecedents','consequents','support','confidence']]
```

In [57]:
```
rules.head()
```

Out[57]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhan
0	(avocado)	(mineral water)	0.033329	0.238368	0.011598	0.348000	1.459926	1.0	0.003654	1.168147	
1	(burgers)	(eggs)	0.087188	0.179709	0.028796	0.330275	1.837830	1.0	0.013128	1.224818	
2	(cake)	(mineral water)	0.081056	0.238368	0.027463	0.338816	1.421397	1.0	0.008142	1.151921	
3	(cereals)	(mineral water)	0.025730	0.238368	0.010265	0.398964	1.673729	1.0	0.004132	1.267198	
4	(chicken)	(mineral water)	0.059992	0.238368	0.022797	0.380000	1.594172	1.0	0.008497	1.228438	

In [63]:
```
rules[rules['antecedents']=={'cake'}]
```

Out[63]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhan
2	(cake)	(mineral water)	0.081056	0.238368	0.027463	0.338816	1.421397	1.0	0.008142	1.151921	

In [64]:
```
rules[rules['antecedents']=={'cake'}]['consequents']
```

Out[64]:
```
2    (mineral water)
Name: consequents, dtype: object
```