Installing mlxtend

In [1]:
```
!pip install mlxtend
```

```
Requirement already satisfied: mlxtend in c:\python 3.10\lib\site-packages
(0.23.0)
Requirement already satisfied: joblib>=0.13.2 in c:\python 3.10\lib\site-pack
ages (from mlxtend) (1.1.0)
Requirement already satisfied: scikit-learn>=1.0.2 in c:\python 3.10\lib\site
-packages (from mlxtend) (1.1.2)
Requirement already satisfied: pandas>=0.24.2 in c:\python 3.10\lib\site-pack
ages (from mlxtend) (1.4.3)
Requirement already satisfied: numpy>=1.16.2 in c:\python 3.10\lib\site-packa
ges (from mlxtend) (1.23.2)
Requirement already satisfied: scipy>=1.2.1 in c:\python 3.10\lib\site-packag
es (from mlxtend) (1.9.1)
Requirement already satisfied: matplotlib>=3.0.0 in c:\python 3.10\lib\site-p
ackages (from mlxtend) (3.5.3)
Requirement already satisfied: packaging>=20.0 in c:\python 3.10\lib\site-pac
kages (from matplotlib>=3.0.0->mlxtend) (21.3)
Requirement already satisfied: cycler>=0.10 in c:\python 3.10\lib\site-packag
es (from matplotlib>=3.0.0->mlxtend) (0.11.0)
Requirement already satisfied: pillow>=6.2.0 in c:\python 3.10\lib\site-packa
ges (from matplotlib>=3.0.0->mlxtend) (9.2.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\python 3.10\lib\site-p
ackages (from matplotlib>=3.0.0->mlxtend) (4.37.1)
Requirement already satisfied: pyparsing>=2.2.1 in c:\python 3.10\lib\site-pa
ckages (from matplotlib>=3.0.0->mlxtend) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\python 3.10\lib\sit
e-packages (from matplotlib>=3.0.0->mlxtend) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\python 3.10\lib\site-p
ackages (from matplotlib>=3.0.0->mlxtend) (1.4.4)
Requirement already satisfied: pytz>=2020.1 in c:\python 3.10\lib\site-packag
es (from pandas>=0.24.2->mlxtend) (2022.2.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\python 3.10\lib\sit
e-packages (from scikit-learn>=1.0.2->mlxtend) (3.1.0)
Requirement already satisfied: six>=1.5 in c:\python 3.10\lib\site-packages
(from python-dateutil>=2.7->matplotlib>=3.0.0->mlxtend) (1.16.0)


[notice] A new release of pip is available: 23.0.1 -> 23.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Importing Necessary Libraries

In [3]:
```
import numpy as np
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules
```

Constructing Data

In [4]:
```
import csv
data = []
with open(r"C:\Users\SAHIL PATIL\Downloads\Market_Basket_Optimisation.csv") as
    reader = csv.reader(file, delimiter=',')
    for row in reader:
        data += [row]
```

In [5]:
```
data[1:10]
```

Out[5]:
```
[['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 [6]:
```
len(data)
```

Out[6]:
```
7501
```

Transforming to logical dataframe

In [7]:
```
TE = TransactionEncoder()
X = TE.fit_transform(data)
```

In [8]:
```
X
```

Out[8]:
```
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]])
```

Dataframe Column Names

In [9]:
```
TE.columns_
```

Out[9]:
```
[' asparagus',
 'almonds',
 'antioxydant juice',
 'asparagus',
 'avocado',
 'babies food',
 'bacon',
 'barbecue sauce',
 'black tea',
 'blueberries',
 'body spray',
 'bramble',
 'brownies',
 'bug spray',
 'burger sauce',
 'burgers',
 'butter',
 'cake',
 'candy bars',
 'carrots',
 'cauliflower',
 'cereals',
 'champagne',
 'chicken',
 'chili',
 'chocolate',
 'chocolate bread',
 'chutney',
 'cider',
 'clothes accessories',
 'cookies',
 'cooking oil',
 'corn',
 'cottage cheese',
 'cream',
 'dessert wine',
 'eggplant',
 'eggs',
 'energy bar',
 'energy drink',
 'escalope',
 'extra dark chocolate',
 'flax seed',
 'french fries',
 'french wine',
 'fresh bread',
 'fresh tuna',
 'fromage blanc',
 'frozen smoothie',
 'frozen vegetables',
 'gluten free bar',
 'grated cheese',
 'green beans',
 'green grapes',
 'green tea',
 'ground beef',
 'gums',
```

```
'ham',
'hand protein bar',
'herb & pepper',
'honey',
'hot dogs',
'ketchup',
'light cream',
'light mayo',
'low fat yogurt',
'magazines',
'mashed potato',
'mayonnaise',
'meatballs',
'melons',
'milk',
'mineral water',
'mint',
'mint green tea',
'muffins',
'mushroom cream sauce',
'napkins',
'nonfat milk',
'oatmeal',
'oil',
'olive oil',
'pancakes',
'parmesan cheese',
'pasta',
'pepper',
'pet food',
'pickles',
'protein bar',
'red wine',
'rice',
'salad',
'salmon',
'salt',
'sandwich',
'shallot',
'shampoo',
'shrimp',
'soda',
'soup',
'spaghetti',
'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']
```

Overall DataFrame

In [10]:
```
df = pd.DataFrame(X, columns = TE.columns_)
```

In [11]:
```
df
```

Out[11]:

	asparagus	almonds	antioxydant juice	asparagus	avocado	babies food	bacon	barbecue sauce	black tea	b
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

Applying Apriori and Association Rules

In [12]:
```
freq_of_itemset = apriori(df, min_support=0.003, use_colnames=True)
```

In [13]:
```
freq_of_itemset
```

Out[13]:

	support	itemsets
0	0.020397	(almonds)
1	0.008932	(antioxydant juice)
2	0.004666	(asparagus)
3	0.033329	(avocado)
4	0.004533	(babies food)
...	...	...
1438	0.003066	(mineral water, spaghetti, ground beef, pancakes)
1439	0.003066	(mineral water, spaghetti, tomatoes, ground beef)
1440	0.003333	(milk, mineral water, olive oil, spaghetti)
1441	0.003066	(milk, mineral water, spaghetti, shrimp)
1442	0.003333	(milk, mineral water, spaghetti, tomatoes)

1443 rows × 2 columns

In [14]:
```
rules = association_rules(freq_of_itemset, metric='lift', min_threshold=0.1)
```

In [15]:
```
rules
```

Out[15]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	levera
0	(almonds)	(burgers)	0.020397	0.087188	0.005199	0.254902	2.923577	0.003
1	(burgers)	(almonds)	0.087188	0.020397	0.005199	0.059633	2.923577	0.003
2	(almonds)	(cake)	0.020397	0.081056	0.003066	0.150327	1.854607	0.001
3	(cake)	(almonds)	0.081056	0.020397	0.003066	0.037829	1.854607	0.001
4	(almonds)	(chocolate)	0.020397	0.163845	0.005999	0.294118	1.795099	0.002
...	...	...	...	...	...	...	...	
5123	(spaghetti, tomatoes)	(milk, mineral water)	0.020931	0.047994	0.003333	0.159236	3.317852	0.002
5124	(milk)	(mineral water, spaghetti, tomatoes)	0.129583	0.009332	0.003333	0.025720	2.756099	0.002
5125	(mineral water)	(milk, spaghetti, tomatoes)	0.238368	0.005866	0.003333	0.013982	2.383631	0.001
5126	(spaghetti)	(milk, mineral water, tomatoes)	0.174110	0.006532	0.003333	0.019142	2.930353	0.002
5127	(tomatoes)	(milk, mineral water, spaghetti)	0.068391	0.015731	0.003333	0.048733	3.097846	0.002

5128 rows × 10 columns

In [ ]:
```

```