Assignment - 7 : Association Rule

Problem Statement-

Assignment on Association Rule Learning Download Market Basket Optimization dataset from below link. Data Set: https://www.kaggle.com/hemanthkumar05/market•basket•optimization. This dataset comprises the list of transactions of a retail company over the period of one week. It contains a total of 7501 transaction records where each record consists of the list of items sold in one transaction. Using this record of transactions and items in each transaction, find the association rules between items. There is no header in the dataset and the first row contains the first transaction, so mentioned header = None here while loading dataset. Follow following steps: A. Data Preprocessing B. Generate the list of transactions from the dataset C. Train Apriori algorithm on the dataset D. Visualize the list of rules E. Generated rules depend on the values of hyper parameters. By increasing the minimum confidence value and find the rules accordingly

importing python libraries

In [1]:

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import *
import numpy as np
```

installing apyori

In [2]:

```
pip install apyori
```

```
Requirement already satisfied: apyori in c:\users\hp\appdata\local\programs\python
\python39\lib\site-packages (1.1.2)
Note: you may need to restart the kernel to use updated packages.
```

```
[notice] A new release of pip available: 22.2.1 -> 22.3
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [3]:

```
from apyori import apriori
```

loading csv file into a dataframe

In [4]:

```
A=pd.read_csv(r"C:\Users\HP\Downloads\Market_Basket_Optimisation.csv",header=None)
A
```

Out[4]:

	0	1	2	3	4	5	6	7	8	9
0	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice
1	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...
7496	butter	light mayo	fresh bread	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7497	burgers	frozen vegetables	eggs	french fries	magazines	green tea	NaN	NaN	NaN	NaN
7498	chicken	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7499	escalope	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7500	eggs	frozen smoothie	yogurt cake	low fat yogurt	NaN	NaN	NaN	NaN	NaN	NaN

7501 rows × 20 columns

counting the total number of null values in each column

In [5]:

```
A.isnull().sum()
```

Out[5]:

```
0        0
1     1754
2     3112
3     4156
4     4972
5     5637
6     6132
7     6520
8     6847
9     7106
10    7245
11    7347
12    7414
13    7454
14    7476
15    7493
16    7497
17    7497
18    7498
19    7500
dtype: int64
```

filling the "NaN" values with "0"

In [6]:
```
A.fillna(0,inplace=True)
A.head()
```

Out[6]:

	0	1	2	3	4	5	6	7	8	9	10
0	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt
1	burgers	meatballs	eggs	0	0	0	0	0	0	0	0
2	chutney	0	0	0	0	0	0	0	0	0	0
3	turkey	avocado	0	0	0	0	0	0	0	0	0
4	mineral water	milk	energy bar	whole wheat rice	green tea	0	0	0	0	0	0

In [7]:
```
A.columns
```

Out[7]:
```
Int64Index([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
            19],
           dtype='int64')
```

In [8]:
```
transactions=[]
for i in range (0,7501):
    transactions.append([str(A.values[i,j]) for j in range (0,20)])

transactions[0]
```

Out[8]:
```
['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']
```

In [9]:
```
rule_list=apriori(transactions,min_support = 0.003, min_confidence=0.003,
                  min_lift=3,min_length=2)
rule_list
```

Out[9]:
```
<generator object apriori at 0x0000020B24D7F7B0>
```

In [10]:
```
Results=list(rule_list)
print(Results[:10])
```

```
[RelationRecord(items=frozenset({'brownies', 'cottage cheese'}), support=0.0034662
045060658577, ordered_statistics=[OrderedStatistic(items_base=frozenset({'brownie
s'}), items_add=frozenset({'cottage cheese'}), confidence=0.10276679841897232, lif
t=3.225329518580382), OrderedStatistic(items_base=frozenset({'cottage cheese'}), i
tems_add=frozenset({'brownies'}), confidence=0.10878661087866107, lift=3.225329518
5803816)]), RelationRecord(items=frozenset({'chicken', 'light cream'}), support=0.
004532728969470737, ordered_statistics=[OrderedStatistic(items_base=frozenset({'ch
icken'}), items_add=frozenset({'light cream'}), confidence=0.07555555555555556, li
ft=4.843950617283951), OrderedStatistic(items_base=frozenset({'light cream'}), ite
ms_add=frozenset({'chicken'}), confidence=0.29059829059829057, lift=4.84395061728
395)]), RelationRecord(items=frozenset({'mushroom cream sauce', 'escalope'}), suppo
rt=0.005732568990801226, ordered_statistics=[OrderedStatistic(items_base=frozenset
({'escalope'}), items_add=frozenset({'mushroom cream sauce'}), confidence=0.072268
9075630252, lift=3.7908326967150496), OrderedStatistic(items_base=frozenset({'mush
room cream sauce'}), items_add=frozenset({'escalope'}), confidence=0.300699300699
3007, lift=3.790832696715049)]), RelationRecord(items=frozenset({'pasta', 'escalop
e'}), support=0.005865884548726837, ordered_statistics=[OrderedStatistic(items_bas
e=frozenset({'escalope'}), items_add=frozenset({'pasta'}), confidence=0.0739495798
3193277, lift=4.700811850163794), OrderedStatistic(items_base=frozenset({'past
a'}), items_add=frozenset({'escalope'}), confidence=0.3728813559322034, lift=4.700
811850163794)]), RelationRecord(items=frozenset({'tomato juice', 'fresh bread'}),
support=0.004266097853619517, ordered_statistics=[OrderedStatistic(items_base=froz
enset({'fresh bread'}), items_add=frozenset({'tomato juice'}), confidence=0.099071
20743034055, lift=3.2593558198902826), OrderedStatistic(items_base=frozenset({'tom
ato juice'}), items_add=frozenset({'fresh bread'}), confidence=0.1403508771929824
5, lift=3.2593558198902826)]), RelationRecord(items=frozenset({'honey', 'fresh tun
a'}), support=0.003999466737768298, ordered_statistics=[OrderedStatistic(items_bas
e=frozenset({'fresh tuna'}), items_add=frozenset({'honey'}), confidence=0.17964071
856287428, lift=3.7850703088205613), OrderedStatistic(items_base=frozenset({'hone
y'}), items_add=frozenset({'fresh tuna'}), confidence=0.08426966292134831, lift=3.
7850703088205613)]), RelationRecord(items=frozenset({'honey', 'fromage blanc'}), s
upport=0.003332888948140248, ordered_statistics=[OrderedStatistic(items_base=froze
nset({'fromage blanc'}), items_add=frozenset({'honey'}), confidence=0.245098039215
6863, lift=5.164270764485569), OrderedStatistic(items_base=frozenset({'honey'}), i
tems_add=frozenset({'fromage blanc'}), confidence=0.0702247191011236, lift=5.16427
076448557)]), RelationRecord(items=frozenset({'ground beef', 'herb & pepper'}), su
pport=0.015997866951073192, ordered_statistics=[OrderedStatistic(items_base=frozen
set({'ground beef'}), items_add=frozenset({'herb & pepper'}), confidence=0.1628222
523744912, lift=3.291993841134928), OrderedStatistic(items_base=frozenset({'herb &
pepper'}), items_add=frozenset({'ground beef'}), confidence=0.3234501347708895, li
ft=3.2919938411349285)]), RelationRecord(items=frozenset({'tomato sauce', 'ground
beef'}), support=0.005332622317024397, ordered_statistics=[OrderedStatistic(items_
base=frozenset({'ground beef'}), items_add=frozenset({'tomato sauce'}), confidence
=0.05427408412483039б, lift=3.840659481324083), OrderedStatistic(items_base=frozen
set({'tomato sauce'}), items_add=frozenset({'ground beef'}), confidence=0.37735849
05660377, lift=3.840659481324083)]), RelationRecord(items=frozenset({'olive oil',
'light cream'}), support=0.003199573390214638, ordered_statistics=[OrderedStatisti
c(items_base=frozenset({'light cream'}), items_add=frozenset({'olive oil'}), confi
dence=0.20512820512820515, lift=3.1147098515519573), OrderedStatistic(items_base=f
rozenset({'olive oil'}), items_add=frozenset({'light cream'}), confidence=0.048582
995951417005, lift=3.114709851551957)])]
```

In [11]:

```
print(len(Results))
```

```
188
```

In [12]:

```
results=pd.DataFrame(Results)
results.head()
```

Out[12]:

	items	support	ordered_statistics
0	(brownies, cottage cheese)	0.003466	[((brownies), (cottage cheese), 0.102766798418...
1	(chicken, light cream)	0.004533	[((chicken), (light cream), 0.0755555555555555...
2	(mushroom cream sauce, escalope)	0.005733	[((escalope), (mushroom cream sauce), 0.072268...
3	(pasta, escalope)	0.005866	[((escalope), (pasta), 0.07394957983193277, 4....
4	(tomato juice, fresh bread)	0.004266	[((fresh bread), (tomato juice), 0.09907120743...

In [13]:

```
support=results.support
```

In [14]:

```
first=[]
second=[]
third=[]
fourth=[]

for i in range(results.shape[0]):
    single_list=results['ordered_statistics'][i][0]
    first.append(list(single_list[0]))
    second.append(list(single_list[1]))
    third.append((single_list[2]))
    fourth.append((single_list[3]))

lhs=pd.DataFrame(first)
rhs=pd.DataFrame(first)
confidence=pd.DataFrame(third,columns=["Confidence"])
lift=pd.DataFrame(fourth,columns=["lift"])
```

In [15]:

```
final=pd.concat([lhs,rhs,support,confidence,lift],axis=1)
final
```

Out[15]:

	0	1	0	1	support	Confidence	lift
0	brownies	None	brownies	None	0.003466	0.102767	3.225330
1	chicken	None	chicken	None	0.004533	0.075556	4.843951
2	escalope	None	escalope	None	0.005733	0.072269	3.790833
3	escalope	None	escalope	None	0.005866	0.073950	4.700812
4	fresh bread	None	fresh bread	None	0.004266	0.099071	3.259356
...	...	...	...	...	...	...	...
183	pancakes	ground beef	pancakes	ground beef	0.003066	0.211009	3.532991
184	ground beef	None	ground beef	None	0.003066	0.031208	3.344117
185	olive oil	None	olive oil	None	0.003333	0.050607	3.216994
186	milk	mineral water	milk	mineral water	0.003066	0.063889	3.014029
187	tomatoes	None	tomatoes	None	0.003333	0.048733	3.097846

188 rows × 7 columns