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:

1. Data Preprocessing
2. Generate the list of transactions from the dataset
3. Train Apriori algorithm on the dataset
4. Visualize the list of rules
5. Generated rules depend on the values of hyper parameters. By increasing the minimum confidence value and find the rules accordingly

Importing Libraries

In [42]:
```
import pandas as pd
```

Loading the dataset

In [43]:
```
A = pd.read_csv(r"C:\Users\User1\Downloads\archive (10)\Market_Basket_Optimisation
```

In [44]:
```
A.head()
```

Out[44]:

	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	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN

In [45]:
```
len(A)
```

Out[45]:
```
7501
```

Data Preprocessing

In [46]:
```
A.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7501 entries, 0 to 7500
Data columns (total 20 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   0       7501 non-null   object
 1   1       5747 non-null   object
 2   2       4389 non-null   object
 3   3       3345 non-null   object
 4   4       2529 non-null   object
 5   5       1864 non-null   object
 6   6       1369 non-null   object
 7   7       981 non-null    object
 8   8       654 non-null    object
 9   9       395 non-null    object
 10  10      256 non-null    object
 11  11      154 non-null    object
 12  12      87 non-null     object
 13  13      47 non-null     object
 14  14      25 non-null     object
 15  15      8 non-null      object
 16  16      4 non-null      object
 17  17      4 non-null      object
 18  18      3 non-null      object
 19  19      1 non-null      object
dtypes: object(20)
memory usage: 1.1+ MB
```

In [47]:
```
A.describe()
```

Out[47]:

	0	1	2	3	4	5	6	7	8	9	10	1
count	7501	5747	4389	3345	2529	1864	1369	981	654	395	256	15
unique	115	117	115	114	110	106	102	98	88	80	66	5
top	mineral water	mineral water	mineral water	mineral water	green tea	french fries	green tea	green tea	green tea	green tea	low fat yogurt	gree te
freq	577	484	375	201	153	107	96	67	57	31	22	1

Generating Transaction List

In [48]:
```
transactions = []
for i in range(0, 7501):
    transactions.append([str(A.values[i,j]) for j in range(0, 20)])
```

Creating apriori model

In [58]:
```
from apyori import apriori
tran_rules = apriori(transactions, min_support = 0.003, min_confidence = 0.2, min_
```

In [59]:
```
results = list(tran_rules)
print(results[:10])
```

```
[RelationRecord(items=frozenset({'light cream', 'chicken'}), support=0.00453272896
9470737, ordered_statistics=[OrderedStatistic(items_base=frozenset({'light crea
m'}), items_add=frozenset({'chicken'}), confidence=0.29059829059829057, lift=4.843
95061728395)]), RelationRecord(items=frozenset({'mushroom cream sauce', 'escalop
e'}), support=0.005732568990801226, ordered_statistics=[OrderedStatistic(items_bas
e=frozenset({'mushroom cream sauce'}), items_add=frozenset({'escalope'}), confiden
ce=0.3006993006993007, lift=3.790832696715049)]), RelationRecord(items=frozenset
({'pasta', 'escalope'}), support=0.005865884548726837, ordered_statistics=[Ordered
Statistic(items_base=frozenset({'pasta'}), items_add=frozenset({'escalope'}), conf
idence=0.3728813559322034, lift=4.700811850163794)]), RelationRecord(items=frozens
et({'fromage blanc', 'honey'}), support=0.003332888948140248, ordered_statistics=
[OrderedStatistic(items_base=frozenset({'fromage blanc'}), items_add=frozenset({'h
oney'}), confidence=0.2450980392156863, lift=5.164270764485569)]), RelationRecord
(items=frozenset({'ground beef', 'herb & pepper'}), support=0.015997866951073192,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'herb & pepper'}), item
s_add=frozenset({'ground beef'}), confidence=0.3234501347708895, lift=3.2919938411
349285)]), RelationRecord(items=frozenset({'ground beef', 'tomato sauce'}), suppor
t=0.005332622317024397, ordered_statistics=[OrderedStatistic(items_base=frozenset
({'tomato sauce'}), items_add=frozenset({'ground beef'}), confidence=0.37735849056
60377, lift=3.840659481324083)]), RelationRecord(items=frozenset({'olive oil', 'li
ght cream'}), support=0.003199573390214638, ordered_statistics=[OrderedStatistic(i
tems_base=frozenset({'light cream'}), items_add=frozenset({'olive oil'}), confiden
ce=0.20512820512820515, lift=3.1147098515519573)]), RelationRecord(items=frozenset
({'whole wheat pasta', 'olive oil'}), support=0.007998933475536596, ordered_statis
tics=[OrderedStatistic(items_base=frozenset({'whole wheat pasta'}), items_add=froz
enset({'olive oil'}), confidence=0.2714932126696833, lift=4.122410097642296)]), Re
lationRecord(items=frozenset({'shrimp', 'pasta'}), support=0.005065991201173177, o
rdered_statistics=[OrderedStatistic(items_base=frozenset({'pasta'}), items_add=fro
zenset({'shrimp'}), confidence=0.3220338983050847, lift=4.506672147735896)]), Rela
tionRecord(items=frozenset({'spaghetti', 'milk', 'avocado'}), support=0.0033328889
48140248, ordered_statistics=[OrderedStatistic(items_base=frozenset({'spaghetti',
'avocado'}), items_add=frozenset({'milk'}), confidence=0.41666666666666663, lift=
3.215449245541838)])]
```

Visualising the results

In [34]:
```
# Visualising the results
def inspect(results):
    lhs         = [tuple(result[2][0][0])[0] for result in results]
    rhs         = [tuple(result[2][0][1])[0] for result in results]
    supports    = [result[1] for result in results]
    return list(zip(lhs, rhs, supports))
```

In [35]:
```
resultsinDataFrame = pd.DataFrame(inspect(results), columns = ['Product 1', 'Produ
```

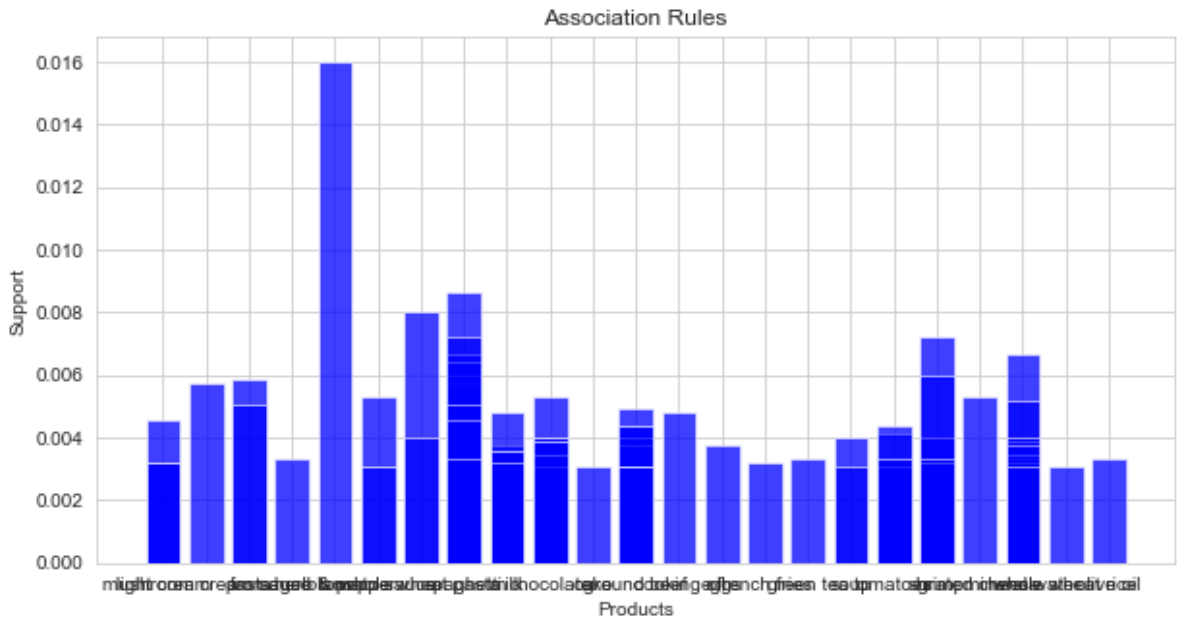
In [36]:
```
resultsinDataFrame.nlargest(n = 10, columns = 'Support')
```

Out[36]:

	Product 1	Product 2	Support
4	herb & pepper	ground beef	0.015998
43	herb & pepper	nan	0.015998
30	spaghetti	ground beef	0.008666
95	spaghetti	nan	0.008666
7	whole wheat pasta	olive oil	0.007999
60	whole wheat pasta	nan	0.007999
34	shrimp	frozen vegetables	0.007199
55	spaghetti	olive oil	0.007199
102	shrimp	nan	0.007199
128	spaghetti	nan	0.007199

In [39]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('whitegrid')
plt.figure(figsize=(10, 5))
plt.title('Association Rules')
plt.xlabel('Products')
plt.ylabel('Support')
plt.bar(resultsinDataFrame['Product 1'], resultsinDataFrame['Support'], color='blu
```

Out[39]:
```
<BarContainer object of 160 artists>
```



In [ ]: