Market basket insights datasets

The dataset contains a list of transactions and the items purchased in each transaction in a Super-Market.

First, we import the libraries:

NumPy - for data manipulation.

Pandas - for data manipulation.

MatPlotLib - for data visualization.

Seaborn - for data visualization.

CSV Reader - to read the dataset.

MLXtend - to apply the Apriori Algorithm.

In[1]:

```
# importing the libraries

import numpy as np

import pandas as pd

from matplotlib import pyplot as plt
```

```
import seaborn as sns

from csv import reader

from mlxtend.preprocessing import TransactionEncoder

from mlxtend.frequent_patterns import apriori, association_rules
```

Next, we read the dataset into a list. Here, we use the csv_reader() function fromt the CSV Reader library instead of the read_csv() function from the Pandas library because the number of columns in each row is different (ragged arrays).

In[2]:

```
# reading the dataset

groceries = []

with open('../input/groceries/groceries.csv', 'r') as read_obj:

    csv_reader = reader(read_obj)

    for row in csv_reader:

        groceries.append(row)
```

In[3]:

```
# items = set(sum(groceries, []))
```

In[4]:

```
# df = pd.DataFrame(columns=items)
```

In[5]:

```
# for i in range(len(groceries)):
```

```
#     transaction = []
#     for item in items:
#         if item in groceries[i]:
#             transaction. append(1)
#         else:
#             transaction. append(0)
#     print(transaction)
#     df = df. append(transaction, ignore_index=True)
```

The above method works, but is very time inefficient. To solve the problem in significantly less time, we use the TransactionEncoder class from the MLXtend library. We fit the object of the class on the list and convert it into a Pandas dataframe with 1 representing item is purchased in that transaction, and 0 representing that the item is not purchased in that transaction

In[6]:

```
# fitting the list and converting the transactions to true and false
encoder = TransactionEncoder()
transactions = encoder. fit(groceries). transform(groceries)
```

In[7]:

```
# converting the true and false to 1 and 0
transactions = transactions. astype('int')
```

In[8]:

```
# converting the transactions array to a datafrmae

df = pd. DataFrame(transactions,  columns=encoder. columns_)
```

In[9]:

```
# viewing the first few rows of the dataframe

df. head()
```

Out[9]:

Instant food products r	UHT - milk	abrasive cleaner	artif. sweetener	baby cosmetics	...	baby food	bags baking	powder	bath room	cleaner bees	whole ack	milk	yogurt	zwieb
0	0	0	1	0	..	0	1	1	0	1	2	0	1	0
0	0	0	0	1	..	1	1	0	0	0	0	0	0	0
0	1	1	0	0	..		0	3	0	1	0	1	0	1
0		0			..		1		1		1		0	0

In[10]:

```
# finding top 5 items with minimum support of 2%

frequent_itemsets[ (frequent_itemsets['length'] == 1) &

                  (frequent_itemsets['support'] >= 0. 02) ][0:5]
```

In[11]:

```
# applying the apriori algorithm
frequent_itemsets = apriori(df, min_support=0.02, use_colnames=True)
frequent_itemsets['length'] = frequent_itemsets['itemsets'].apply(lambda x: len(x))
frequent_itemsets
```

In[12]:

```
# finding itemsets having length more than 1 and minimum support of 5%
frequent_itemsets[(frequent_itemsets['length'] > 1) &
                  (frequent_itemsets['support'] >= 0.05)]
```

In[13]:

```
# finding itemsets having length 2 and minimum support of 2%
frequent_itemsets[(frequent_itemsets['length'] == 2) &
                  (frequent_itemsets['support'] >= 0.02)]
```