Apriori Algorithm

Using Marketing Basket Data

Intuition

The apriori algorithm is a type of association learning used to find groups of items that occur together frequently. This is typically done with market-based analysis.

Algorithm

Count all items Filter for frequency

Count all pairs filter for frequent pairs

Count candidate triples Filter for frequent triples

Implementation

In [1]:
```
# Required Packages
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import sklearn as sk
from collections import defaultdict
```

In [2]:
```python
# Load the data
data = pd.read_csv("../../../_resources/data/Market_Basket_Optimisation
data.head() # Display
```

Out[2]:

	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	gre
1	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN	N

In [3]:
```python
transactions = []
n = len(data)
for i in range(0, n):
    transactions.append([str(data.values[i,j]) for j in range(0, 20)])

transactions
```

```
  'nan'],
 ['mineral water',
  'milk',
  'energy bar',
  'whole wheat rice',
  'green tea',
  'nan',
  'nan',
  'nan',
  'nan',

  'nan',
  'nan',
  'nan',
  'nan',
  'nan',
  'nan',
  'nan',
  'nan',
  'nan',
```

Count all items and filter for frequency threshold

In [4]:
```python
# Occurences that makes an itemset 'frequent'
threshold = 50

item_counts = defaultdict(int)

# Find candidate items
for trans in transactions:
    for item in trans:
        item_counts[item] += 1

# filter for frequent items
frequent_items = set()
for key in item_counts:
    if item_counts[key] > threshold:
        frequent_items.add(key)

print("FREQUENT ITEMS IN BASKET : ")
frequent_items
```

```
FREQUENT ITEMS IN BASKET :
```

Out[4]:
```
{'almonds',
 'antioxydant juice',
 'avocado',
 'bacon',
 'barbecue sauce',
 'black tea',
 'blueberries',
 'body spray',
 'brownies',
 'bug spray',
 'burgers',
 'butter',
 'cake',
 'candy bars',
 'carrots',
 'cereals',
 'champagne',
 'chicken',
 'chocolate',
 'cider',
 'clothes accessories',
 'cookies',
 'cooking oil',
 'cottage cheese',
 'eggplant',
 'eggs',
 'energy bar',
 'energy drink',
 'escalope',
 'extra dark chocolate',
 'flax seed',
 'french fries',
 'french wine',
 'fresh bread',
 'fresh tuna',

 'fromage blanc'
```

```
'fromage blanc',
'frozen smoothie',
'frozen vegetables',
'gluten free bar',
'grated cheese',
'green beans',
'green grapes',
'green tea',
'ground beef',
'gums',
'ham',
'herb & pepper',
'honey',
'hot dogs',
'light cream',
'light mayo',
'low fat yogurt',
'magazines',
'meatballs',
'melons',
'milk',
'mineral water',
'mint',
'muffins',
'mushroom cream sauce',
'nan',
'nonfat milk',
'oil',
'olive oil',
'pancakes',
'parmesan cheese',
'pasta',
'pepper',
'protein bar',
'red wine',
'rice',
'salmon',
'salt',
'shallot',
'shrimp',
'soup',
'spaghetti',
'spinach',
'strawberries',
'strong cheese',
'tomato juice',
'tomato sauce',
'tomatoes',
'toothpaste',
'turkey',
'vegetables mix',
'white wine',
'whole weat flour',
'whole wheat pasta',
'whole wheat rice',
'yams',


'yogurt cake'
```

```
 'zucchini'}
```

Count all item PAIRS and filter for frequency threshold

In [5]:
```python
pair_counts = defaultdict(int)
frequent_pairs = set()
```

In [9]:
```python
def normalize_group(*args):
    return str(sorted(args))

# Get counts of candidate pairs
for trans in transactions:
    for items in trans:
        for idx_1 in range(len(items) - 1):
            if items[idx_1] not in frequent_items:
                continue
            for idx_2 in range(idx_1 + 1, len(items)):
                if items[idx_2] not in frequent_items:
                    continue
                pair = normalize_group(items[idx_1], items[idx_2])
                pair_counts[pair] += 1


# Get frequent pairs
for key in pair_counts:
    if pair_counts[key] > threshold:
        frequent_pairs.add(key)

print("FREQUENT PAIRS IN BASKET : ")
frequent_pairs
```

```
FREQUENT PAIRS IN BASKET :
```

Out[9]:
```
set()
```

Count all item TRIPLES and filter for frequency threshold

In [12]:
```python
triple_counts = defaultdict(int)
frequent_triple = set()
```

```
In [13]:
def generate_pairs(*args):
    pairs = []
    for idx_1 in range(len(args) - 1):
        for idx_2 in range(idx_1+1, len(args)):
            pairs.append(normalize_group(args[idx_1], args[idx_1]))
    return pairs

# Get counts of candidate pairs
for trans in transactions:
    for items in trans:
        for idx_1 in range(len(items) - 1):
            if items[idx_1] not in frequent_items:
                continue
            for idx_2 in range(idx_1 + 1, len(items)):
                if items[idx_2] not in frequent_items:
                    continue
                first_pair = normalize_group(items[idx_1], items[idx_2]
                if first_pair not in frequent_pairs:
                    continue
                for idx_3 in range(idx_2 + 1, len(items)):
                    if items[idx_3] not in frequent_items:
                        continue
                    pairs = generate_pairs(items[idx_1], items[idx_2],
                    if any(pair not in frequent_pairs for pair in pairs
                        continue
                    triple = normalize_group(items[idx_1], items[idx_2]
                    triple_counts[triple] += 1



# Get frequent pairs
for key in triple_counts:
    if triple_counts[key] > threshold:
        frequent_triple.add(key)

print("FREQUENT TRIPLE IN BASKET : ")
frequent_triple
```

```
FREQUENT TRIPLE IN BASKET :
```

```
Out[13]: set()
```

```
In [ ]:
```