Market Basket Analysis using Apriori Algorithm in Python

Importing Libraries

In [1]:

```
!pip install seaborn
```

```
Requirement already satisfied: seaborn in c:\users\pc\anaconda3\lib\site-packages (0.11.1)
Requirement already satisfied: pandas>=0.23 in c:\users\pc\anaconda3\lib\site-packages (from seaborn) (1.2.4)
Requirement already satisfied: numpy>=1.15 in c:\users\pc\anaconda3\lib\site-packages (from seaborn) (1.20.1)
Requirement already satisfied: scipy>=1.0 in c:\users\pc\anaconda3\lib\site-packages (from seaborn) (1.6.2)
Requirement already satisfied: matplotlib>=2.2 in c:\users\pc\anaconda3\lib\site-packages (from seaborn) (3.3.4)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (2.4.7)
Requirement already satisfied: cycler>=0.10 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (0.10.0)
Requirement already satisfied: pillow>=6.2.0 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (8.2.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (2.8.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (1.3.1)
Requirement already satisfied: six in c:\users\pc\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib>=2.2->seaborn) (1.15.0)
Requirement already satisfied: pytz>=2017.3 in c:\users\pc\anaconda3\lib\site-packages (from pandas>=0.23->seaborn) (2021.1)
```

In [48]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading the Dataset

In [45]:

```
data = pd.read_csv('Groceries_dataset.csv')
```

In [46]:

```
data
```

Out[46]:

	Member_number	Date	itemDescription
0	1808	21-07-2015	tropical fruit
1	2552	05-01-2015	whole milk
2	2300	19-09-2015	pip fruit
3	1187	12-12-2015	other vegetables
4	3037	01-02-2015	whole milk
...	...	...	...
38760	4471	08-10-2014	sliced cheese
38761	2022	23-02-2014	candy
38762	1097	16-04-2014	cake bar
38763	1510	03-12-2014	fruit/vegetable juice
38764	1521	26-12-2014	cat food

38765 rows × 3 columns

Data Exploration

In [11]:

```
x = data['itemDescription'].value_counts().sort_values(ascending=False)[:10]
```

In [12]:

```
x
```

Out[12]:

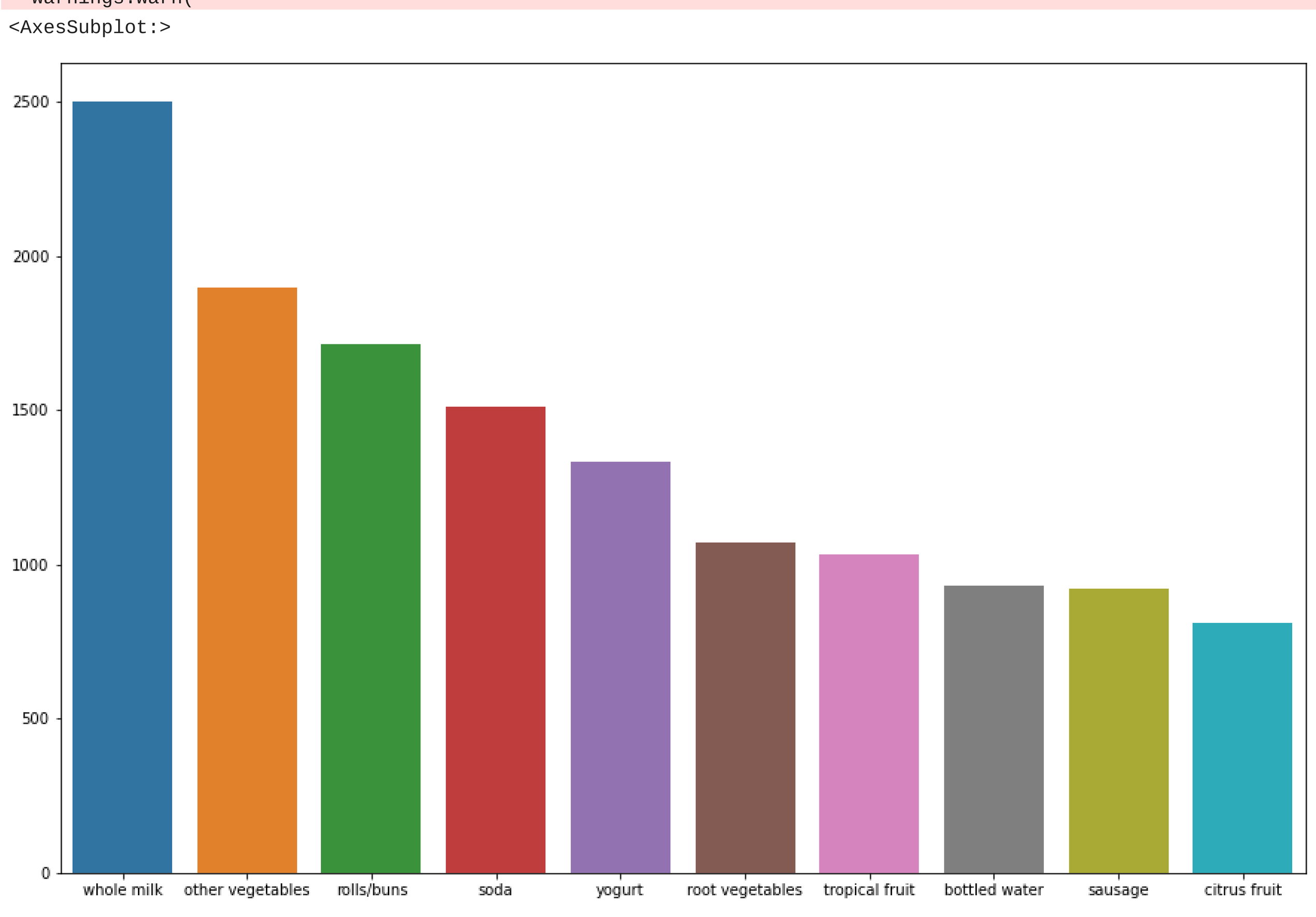
```
whole milk          2502
other vegetables    1898
rolls/buns          1716
soda                1514
yogurt              1334
root vegetables     1071
tropical fruit      1032
bottled water        933
sausage              924
citrus fruit         812
Name: itemDescription, dtype: int64
```

In [63]:

```
plt.figure(figsize=(15,10))
sns.barplot(x.index,x.values)
```

```
C:\Users\PC\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional a
rgument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[63]: <AxesSubplot:>



Apriori Algorithm

It is an algorithm that uses frequent itemset to generate association rules. It is based on the concept that a subset of a frequent itemset must also be a frequent itemset

A and B are products and N is the number of total transactions

Support: Frequency of item A or combination of iteam A and B. =freq(A,B)/N)

Confidence: How often the items A and B occur given that A is bought. =freq(A/B)/freq(A)

Lift: The strength of the rule. = Support/Support(A)*Support(B)

In [18]:

```
!pip install mlxtend
```

```
Collecting mlxtend
  Downloading mlxtend-0.23.0-py3-none-any.whl (1.4 MB)
Requirement already satisfied: numpy>=1.16.2 in c:\users\pc\anaconda3\lib\site-packages (from mlxtend) (1.20.1)
Requirement already satisfied: scipy>=1.2.1 in c:\users\pc\anaconda3\lib\site-packages (from mlxtend) (1.6.2)
Requirement already satisfied: matplotlib>=3.0.0 in c:\users\pc\anaconda3\lib\site-packages (from mlxtend) (3.3.4)
Requirement already satisfied: pandas>=0.24.2 in c:\users\pc\anaconda3\lib\site-packages (from mlxtend) (1.2.4)
Requirement already satisfied: joblib>=0.13.2 in c:\users\pc\anaconda3\lib\site-packages (from mlxtend) (1.0.1)
Collecting scikit-learn>=1.0.2
  Downloading scikit_learn-1.3.2-cp38-cp38-win_amd64.whl (9.3 MB)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.8.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (8.2.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (1.3.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.4.7)
Requirement already satisfied: cycler>=0.10 in c:\users\pc\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (0.10.0)
Requirement already satisfied: six in c:\users\pc\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib>=3.0.0->mlxtend) (1.15.0)
Requirement already satisfied: pytz>=2017.3 in c:\users\pc\anaconda3\lib\site-packages (from pandas>=0.24.2->mlxtend) (2021.1)
Collecting joblib>=0.13.2
  Downloading joblib-1.3.2-py3-none-any.whl (302 kB)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\pc\anaconda3\lib\site-packages (from scikit-learn>=1.0.2->mlxtend) (2.1.0)
Installing collected packages: joblib, scikit-learn, mlxtend
  Attempting uninstall: joblib
    Found existing installation: joblib 1.0.1
    Uninstalling joblib-1.0.1:
      Successfully uninstalled joblib-1.0.1
  Attempting uninstall: scikit-learn
    Found existing installation: scikit-learn 0.24.1
    Uninstalling scikit-learn-0.24.1:
      Successfully uninstalled scikit-learn-0.24.1
Successfully installed joblib-1.3.2 mlxtend-0.23.0 scikit-learn-1.3.2
```

In [20]:

```
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
```

In [50]:

```
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
```

In [51]:

```
data['Quantity'] = 1
```

In [52]:

```
data
```

Out[52]:

	Member_number	Date	itemDescription	Quantity
0	1808	21-07-2015	tropical fruit	1
1	2552	05-01-2015	whole milk	1
2	2300	19-09-2015	pip fruit	1
3	1187	12-12-2015	other vegetables	1
4	3037	01-02-2015	whole milk	1
...	...	...	...	...
38760	4471	08-10-2014	sliced cheese	1
38761	2022	23-02-2014	candy	1
38762	1097	16-04-2014	cake bar	1
38763	1510	03-12-2014	fruit/vegetable juice	1
38764	1521	26-12-2014	cat food	1

38765 rows × 4 columns

In [53]:

```
transactions = data.groupby(['Member_number','itemDescription'])['Quantity'].sum().unstack().reset_index().set_index('Member_number')
```

In [54]:

```
transactions=transactions.fillna(0)
```

In [55]:

```
transactions
```

Out[55]:

itemDescription	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics	bags	baking powder	bathroom cleaner	beef	berries	...	turkey	vinegar	waffles	whipped/sour cream	whisky	white bread	white wine	whole milk	yogurt	zwieback
Member_number																					
1000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	1.0	0.0
1001	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	...	0.0	0.0	0.0	1.0	0.0	1.0	0.0	2.0	0.0	0.0
1002	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
1003	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1004	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4996	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4997	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0
4998	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4999	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	...	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0
5000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

3898 rows × 167 columns

In [56]:

```
def encode(x):
    if (x<=0):
        return 0
    elif x>=0:
        return 1
basket = transactions.applymap(encode)
```

In [57]:

```
basket
```

Out[57]:

itemDescription	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics	bags	baking powder	bathroom cleaner	beef	berries	...	turkey	vinegar	waffles	whipped/sour cream	whisky	white bread	white wine	whole milk	yogurt	zwieback
Member_number																					
1000	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	1	1	0
1001	0	0	0	0	0	0	0	0	1	0	...	0	0	0	1	0	1	0	1	0	0
1002	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	1	0	0
1003	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1004	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4996	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4997	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	1	0	0
4998	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4999	0	0	0	0	0	0	0	0	0	1	...	0	0	0	1	0	0	0	0	1	0
5000	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

3898 rows × 167 columns

In [58]:

```
frequent_itemset =  apriori(basket, min_support=0.06, use_colnames=True)
rules = association_rules(frequent_itemset,metric='lift',min_threshold=1)
```

In [59]:

```
rules.head()
```

Out[59]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(beef)	(whole milk)	0.119548	0.458184	0.064135	0.536481	1.170886	0.009360	1.168919	0.165762
1	(whole milk)	(beef)	0.458184	0.119548	0.064135	0.139978	1.170886	0.009360	1.023754	0.269364
2	(bottled beer)	(other vegetables)	0.158799	0.376603	0.068497	0.431341	1.145345	0.008692	1.096257	0.150857
3	(other vegetables)	(bottled beer)	0.376603	0.158799	0.068497	0.181880	1.145345	0.008692	1.028212	0.203563
4	(bottled beer)	(rolls/buns)	0.158799	0.349666	0.063109	0.397415	1.136555	0.007582	1.079240	0.142829

In [60]:

```
rules[(rules['confidence']>0.4) & (rules['lift']>1)]
```

Out[60]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(beef)	(whole milk)	0.119548	0.458184	0.064135	0.536481	1.170886	0.009360	1.168919	0.165762
2	(bottled beer)	(other vegetables)	0.158799	0.376603	0.068497	0.431341	1.145345	0.008692	1.096257	0.150857
6	(bottled beer)	(whole milk)	0.158799	0.458184	0.085428	0.537964	1.174124	0.012669	1.172672	0.176297
8	(bottled water)	(other vegetables)	0.213699	0.376603	0.093894	0.439376	1.166680	0.013414	1.111969	0.181695
14	(bottled water)	(whole milk)	0.213699	0.458184	0.112365	0.525810	1.147597	0.014452	1.142615	0.163569
18	(brown bread)	(whole milk)	0.135967	0.458184	0.069779	0.513208	1.120091	0.007481	1.113034	0.124087
20	(butter)	(whole milk)	0.126475	0.458184	0.066188	0.523327	1.142176	0.008239	1.136661	0.142501
23	(canned beer)	(other vegetables)	0.165213	0.376603	0.067214	0.406832	1.080267	0.004994	1.050962	0.089008
25	(canned beer)	(rolls/buns)	0.165213	0.349666	0.066701	0.403727	1.154605	0.008931	1.090663	0.160404
27	(canned beer)	(whole milk)	0.165213	0.458184	0.087224	0.527950	1.152268	0.011526	1.147795	0.158299
29	(citrus fruit)	(other vegetables)	0.185480	0.376603	0.077476	0.417704	1.109135	0.007623	1.070584	0.120803
35	(citrus fruit)	(whole milk)	0.185480	0.458184	0.092355	0.497925	1.086737	0.007371	1.079155	0.097990
36	(curd)	(whole milk)	0.120831	0.458184	0.063622	0.526539	1.149188	0.008259	1.144374	0.147663
39	(domestic eggs)	(whole milk)	0.133145	0.458184	0.070292	0.527938	1.152242	0.009287	1.147766	0.152421
41	(frankfurter)	(other vegetables)	0.137506	0.376603	0.061057	0.444030	1.179038	0.009272	1.121277	0.176061
43	(frankfurter)	(whole milk)	0.137506	0.458184	0.067984	0.494403	1.079050	0.004980	1.071637	0.084938
45	(fruit/vegetable juice)	(whole milk)	0.124936	0.458184	0.062340	0.498973	1.089025	0.005096	1.081412	0.093418
47	(newspapers)	(whole milk)	0.139815	0.458184	0.072345	0.517431	1.129310	0.008284	1.122775	0.133115
49	(pastry)	(other vegetables)	0.177527	0.376603	0.071575	0.403179	1.070567	0.004718	1.044529	0.080143
51	(pip fruit)	(other vegetables)	0.170600	0.376603	0.072345	0.424060	1.126013	0.008096	1.082399	0.134929
53	(rolls/buns)	(other vegetables)	0.349666	0.376603	0.146742	0.419663	1.114335	0.015056	1.074197	0.157772
54	(root vegetables)	(other vegetables)	0.230631	0.376603	0.094151	0.408231	1.083982	0.007294	1.053447	0.100700
57	(sausage)	(other vegetables)	0.206003	0.376603	0.092868	0.450809	1.197040	0.015287	1.135119	0.207314
58	(shopping bags)	(other vegetables)	0.168291	0.376603	0.073114	0.434451	1.153604	0.009735	1.102286	0.160094
65	(whipped/sour cream)	(other vegetables)	0.154695	0.376603	0.066957	0.432836	1.149315	0.008699	1.099147	0.153692
66	(whole milk)	(other vegetables)	0.458184	0.376603	0.191380	0.417693	1.109106	0.018827	1.070564	0.181562
67	(other vegetables)	(whole milk)	0.376603	0.458184	0.191380	0.508174	1.109106	0.018827	1.101643	0.157802
69	(yogurt)	(other vegetables)	0.282966	0.376603	0.120318	0.425204	1.129050	0.013752	1.084553	0.159406
75	(pastry)	(whole milk)	0.177527	0.458184	0.091072	0.513006	1.119651	0.009732	1.112572	0.129931
81	(pip fruit)	(whole milk)	0.170600	0.458184	0.086968	0.509774	1.112598	0.008801	1.105239	0.122020
83	(pork)	(whole milk)	0.132376	0.458184	0.066957	0.505814	1.103955	0.006305	1.096381	0.108533
88	(shopping bags)	(rolls/buns)	0.168291	0.349666	0.068753	0.408537	1.168361	0.009907	1.099533	0.173258
95	(rolls/buns)	(whole milk)	0.349666	0.458184	0.178553	0.510638	1.114484	0.018342	1.107190	0.157955
100	(root vegetables)	(whole milk)	0.230631	0.458184	0.113135	0.490545	1.070630	0.007464	1.063522	0.085746
107	(sausage)	(whole milk)	0.206003	0.458184	0.106978	0.519303	1.133394	0.012591	1.127146	0.148230
112	(shopping bags)	(whole milk)	0.168291	0.458184	0.091329	0.542683	1.184422	0.014220	1.184772	0.187213
117	(soda)	(whole milk)	0.313494	0.458184	0.151103	0.481997	1.051973	0.007465	1.045971	0.071966
120	(tropical fruit)	(whole milk)	0.233710	0.458184	0.116470	0.498353	1.087672	0.009388	1.080076	0.105189
125	(whipped/sour cream)	(whole milk)	0.154695	0.458184	0.079785	0.515755	1.125650	0.008906	1.118888	0.132052
127	(yogurt)	(whole milk)	0.282966	0.458184	0.150590	0.532185	1.161510	0.020940	1.158185	0.193926
128	(whole milk, rolls/buns)	(other vegetables)	0.178553	0.376603	0.082093	0.459770	1.220834	0.014850	1.153947	0.220206
129	(whole milk, other vegetables)	(rolls/buns)	0.191380	0.349666	0.082093	0.428954	1.226753	0.015174	1.138847	0.228587
130	(other vegetables, rolls/buns)	(whole milk)	0.146742	0.458184	0.082093	0.559441	1.220996	0.014859	1.229837	0.212124
134	(whole milk, soda)	(other vegetables)	0.151103	0.376603	0.069266	0.458404	1.217206	0.012360	1.151036	0.210210
136	(other vegetables, soda)	(whole milk)	0.124166	0.458184	0.069266	0.557851	1.217528	0.012375	1.225416	0.203992
140	(whole milk, yogurt)	(other vegetables)	0.150590	0.376603	0.071832	0.477002	1.266589	0.015119	1.191967	0.247793
142	(other vegetables, yogurt)	(whole milk)	0.120318	0.458184	0.071832	0.597015	1.303003	0.016704	1.344507	0.264348
146	(whole milk, soda)	(rolls/buns)	0.151103	0.349666	0.065162	0.431239	1.233288	0.012326	1.143422	0.222829
148	(soda, rolls/buns)	(whole milk)	0.119805	0.458184	0.065162	0.543897	1.187072	0.010269	1.187926	0.179041
153	(whole milk, yogurt)	(rolls/buns)	0.150590	0.349666	0.065931	0.437819	1.252106	0.013275	1.156805	0.237041
154	(rolls/buns, yogurt)	(whole milk)	0.111339	0.458184	0.065931	0.592166	1.292420	0.014917	1.328521	0.254605