```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
```

```
1 data = pd.read_csv('Market_Basket_Optimisation.csv',header=None)
2 data
```

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

7501 rows × 20 columns

```
1 data.describe()
```

	0	1	2	3	4	5	6	7	8	9	10
count	7501	5747	4389	3345	2529	1864	1369	981	654	395	256
unique	115	117	115	114	110	106	102	98	88	80	66
top	mineral water	mineral water	mineral water	mineral water	green tea	french fries	green tea	green tea	green tea	green tea	low fat yogurt

```
1 data.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
```

```
1 data.isnull().sum()
```

```
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
```

```
1 transactions = data.values.reshape(-1).tolist()
2 transactions
```

```
1 data_list = pd.DataFrame(transactions)
```

```
2 data_list
```

	0
0	shrimp
1	almonds
2	avocado
3	vegetables mix
4	green grapes
...	...
150015	NaN
150016	NaN
150017	NaN
150018	NaN
150019	NaN

150020 rows × 1 columns

```
1 data_list['Count'] = 1
2 data_list['Count']
```

```
0         1
1         1
2         1
3         1
4         1
         ..
150015    1
150016    1
150017    1
150018    1
150019    1
Name: Count, Length: 150020, dtype: int64
```

```
1 cnt=0
2 for i in data_list[0]:
3     if i == 'burgers':
4         cnt=cnt+1
5 cnt
```

```
654
```

```
1 # Grouping items and rename columns
2 data_list = data_list.groupby(by=[0], as_index=False).count().sort_values(by
3 data_list
```

	0	Count
0	asparagus	1
112	water spray	3
77	napkins	5
34	cream	7
11	bramble	14
...	...	...
25	chocolate	1230
43	french fries	1282
100	spaghetti	1306
37	eggs	1348
72	mineral water	1788

120 rows × 2 columns

```
data_list['Percentage'] = (data_list['Count']/data_list['Count'].sum())
data_list['Percentage']
```

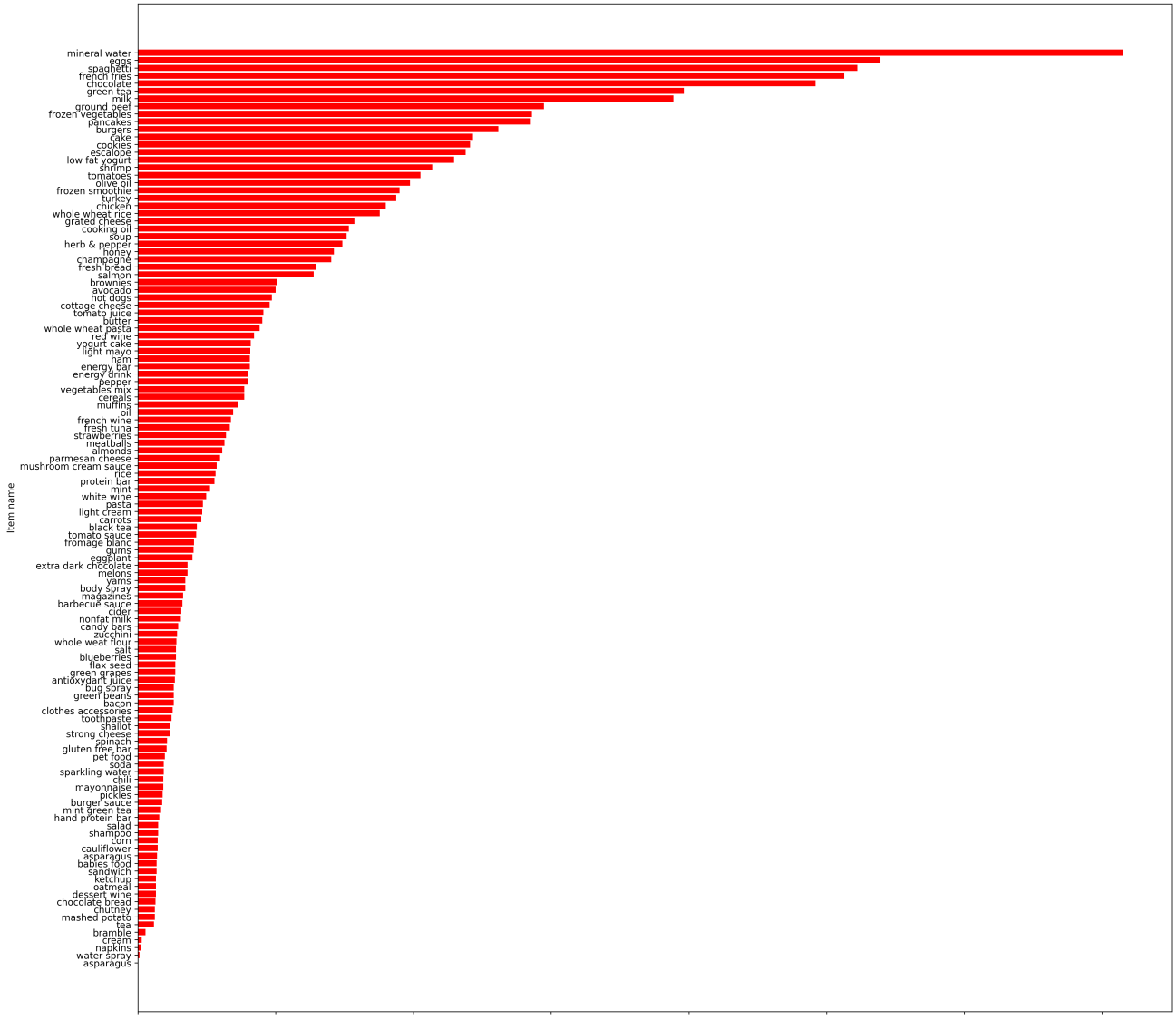
```
0      0.000034
112    0.000102
77     0.000170
34     0.000238
11     0.000477
         ...
25     0.041889
43     0.043660
100    0.044478
37     0.045908
72     0.060893
Name: Percentage, Length: 120, dtype: float64
```

```
data_list = data_list.rename(columns = {0:'Item'})
data_list
```

	Item	Count	Percentage
0	asparagus	1	0.000034
112	water spray	3	0.000102
77	napkins	5	0.000170
34	cream	7	0.000238

```
# frequency graph
plt.figure(dpi = 300, figsize=(20,20))
plt.barh(data_list['Item'] ,data_list['Count'],color='red')
#plt.xticks(rotation=90)
plt.xlabel('Count')
plt.ylabel('Item name')

plt.show()
```




```
1 # creating list of lists
2 transactions2 = data.stack().groupby(level=0).apply(list).tolist()
3 transactions2
```

```
     'eggs',
     'low fat yogurt',
     'mint'],
    ['yogurt cake', 'mint'],
    ['sandwich', 'salt'],
    ['shrimp', 'fresh bread', 'green tea'],
    ['pepper', 'tomato sauce', 'escalope'],
    ['mineral water', 'cake', 'protein bar', 'toothpaste'],
    ['candy bars'],
    ['red wine'],
    ['spaghetti', 'mineral water', 'honey', 'strawberries'],
    ['mineral water', 'eggs', 'green tea'],
    ['grated cheese',
     'frozen vegetables',
     'parmesan cheese',
     'french wine',
     'french fries',
     'fresh bread',
     'green tea'],
    ['frozen vegetables',
     'parmesan cheese',
     'whole wheat pasta',
```

```
 'spaghetti',
 'mineral water',
 'chocolate',
 'avocado',
 'milk',
 'olive oil',
 'muffins',
 'cake'],
['chocolate',
 'grated cheese',
 'shrimp',
 'spaghetti',
 'olive oil',
 'honey',
 'strawberries',
 'mint'],
['spaghetti', 'strawberries'],
['burgers', 'chocolate', 'mushroom cream sauce'],
['mineral water', 'olive oil', 'extra dark chocolate'],
['eggs', 'green tea'],
['tomato juice'],
['muffins', 'low fat yogurt'],
['soup',
 'olive oil',
 'pancakes',
 'light cream',
 'champagne',
 'low fat yogurt'],
['mineral water', 'vegetables mix', 'low fat yogurt'],
['pepper',
 'mineral water',
 'avocado',
 'pancakes',
 'eggs',
 'honey',
 'green tea',
```

```
1 !pip install efficient_apriori
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/p
Collecting efficient_apriori
  Downloading efficient_apriori-2.0.1-py3-none-any.whl (14 kB)
Installing collected packages: efficient-apriori
Successfully installed efficient-apriori-2.0.1
```

```
1 from efficient_apriori import apriori
```

```
1 itemsets, rules = apriori(transactions2, min_support = 0.03, min_confidence
```

```
Generating itemsets.
 Counting itemsets of length 1.
  Found 120 candidate itemsets of length 1.
  Found 36 large itemsets of length 1.
 Counting itemsets of length 2.
  Found 630 candidate itemsets of length 2.
  Found 18 large itemsets of length 2.
```

```
 Counting itemsets of length 3.
  Found 14 candidate itemsets of length 3.
Itemset generation terminated.

Generating rules from itemsets.
 Generating rules of size 2.
Rule generation terminated.
```

1 itemsets

```
{1: {('shrimp',): 536,
  ('avocado',): 250,
  ('cottage cheese',): 239,
  ('tomato juice',): 228,
  ('low fat yogurt',): 574,
  ('green tea',): 991,
  ('honey',): 356,
  ('mineral water',): 1788,
  ('salmon',): 319,
  ('frozen smoothie',): 475,
  ('olive oil',): 494,
  ('burgers',): 654,
  ('eggs',): 1348,
  ('turkey',): 469,
  ('milk',): 972,
  ('whole wheat rice',): 439,
  ('french fries',): 1282,
  ('soup',): 379,
  ('frozen vegetables',): 715,
  ('spaghetti',): 1306,
  ('cookies',): 603,
  ('cooking oil',): 383,
  ('champagne',): 351,
  ('chocolate',): 1229,
  ('chicken',): 450,
  ('tomatoes',): 513,
  ('pancakes',): 713,
  ('grated cheese',): 393,
  ('fresh bread',): 323,
  ('ground beef',): 737,
  ('escalope',): 595,
  ('herb & pepper',): 371,
  ('cake',): 608,
  ('hot dogs',): 243,
  ('brownies',): 253,
  ('butter',): 226},
 2: {('chocolate', 'eggs'): 249,
  ('chocolate', 'french fries'): 258,
  ('chocolate', 'milk'): 241,
  ('chocolate', 'mineral water'): 395,
  ('chocolate', 'spaghetti'): 294,
  ('eggs', 'french fries'): 273,
  ('eggs', 'milk'): 231,
  ('eggs', 'mineral water'): 382,
  ('eggs', 'spaghetti'): 274,
  ('french fries', 'mineral water'): 253,
  ('frozen vegetables', 'mineral water'): 268,
  ('green tea', 'mineral water'): 233,
```

```
 ('ground beef', 'mineral water'): 307,
 ('ground beef', 'spaghetti'): 294,
 ('milk', 'mineral water'): 360,
 ('milk', 'spaghetti'): 266,
 ('mineral water', 'pancakes'): 253,
 ('mineral water', 'spaghetti'): 448}}
```

```
1 rules
```

```
[{chocolate} -> {eggs},
 {french fries} -> {chocolate},
 {chocolate} -> {french fries},
 {milk} -> {chocolate},
 {mineral water} -> {chocolate},
 {chocolate} -> {mineral water},
 {spaghetti} -> {chocolate},
 {chocolate} -> {spaghetti},
 {french fries} -> {eggs},
 {eggs} -> {french fries},
 {milk} -> {eggs},
 {mineral water} -> {eggs},
 {eggs} -> {mineral water},
 {spaghetti} -> {eggs},
 {eggs} -> {spaghetti},
 {frozen vegetables} -> {mineral water},
 {green tea} -> {mineral water},
 {ground beef} -> {mineral water},
 {spaghetti} -> {ground beef},
 {ground beef} -> {spaghetti},
 {mineral water} -> {milk},
 {milk} -> {mineral water},
 {spaghetti} -> {milk},
 {milk} -> {spaghetti},
 {pancakes} -> {mineral water},
 {spaghetti} -> {mineral water},
 {mineral water} -> {spaghetti}]
```

```
1 for item in sorted(rules, key=lambda item: (item.lift), reverse=True):
2     print(item)
```

```
{spaghetti} -> {ground beef} (conf: 0.225, supp: 0.039, lift: 2.291, conv: 1.164)
{ground beef} -> {spaghetti} (conf: 0.399, supp: 0.039, lift: 2.291, conv: 1.374)
{ground beef} -> {mineral water} (conf: 0.417, supp: 0.041, lift: 1.748, conv: 1.305)
{frozen vegetables} -> {mineral water} (conf: 0.375, supp: 0.036, lift: 1.572, conv:
{spaghetti} -> {milk} (conf: 0.204, supp: 0.035, lift: 1.572, conv: 1.093)
{milk} -> {spaghetti} (conf: 0.274, supp: 0.035, lift: 1.572, conv: 1.137)
{mineral water} -> {milk} (conf: 0.201, supp: 0.048, lift: 1.554, conv: 1.090)
{milk} -> {mineral water} (conf: 0.370, supp: 0.048, lift: 1.554, conv: 1.210)
{milk} -> {chocolate} (conf: 0.248, supp: 0.032, lift: 1.513, conv: 1.112)
{pancakes} -> {mineral water} (conf: 0.355, supp: 0.034, lift: 1.489, conv: 1.181)
{spaghetti} -> {mineral water} (conf: 0.343, supp: 0.060, lift: 1.439, conv: 1.159)
{mineral water} -> {spaghetti} (conf: 0.251, supp: 0.060, lift: 1.439, conv: 1.102)
{spaghetti} -> {chocolate} (conf: 0.225, supp: 0.039, lift: 1.374, conv: 1.079)
{chocolate} -> {spaghetti} (conf: 0.239, supp: 0.039, lift: 1.374, conv: 1.086)
{mineral water} -> {chocolate} (conf: 0.221, supp: 0.053, lift: 1.348, conv: 1.073)
{chocolate} -> {mineral water} (conf: 0.321, supp: 0.053, lift: 1.348, conv: 1.122)
{milk} -> {eggs} (conf: 0.238, supp: 0.031, lift: 1.322, conv: 1.076)
```

```
{french fries} -> {chocolate} (conf: 0.201, supp: 0.034, lift: 1.228, conv: 1.047)
{chocolate} -> {french fries} (conf: 0.210, supp: 0.034, lift: 1.228, conv: 1.049)
{mineral water} -> {eggs} (conf: 0.214, supp: 0.051, lift: 1.189, conv: 1.043)
{eggs} -> {mineral water} (conf: 0.283, supp: 0.051, lift: 1.189, conv: 1.063)
{french fries} -> {eggs} (conf: 0.213, supp: 0.036, lift: 1.185, conv: 1.042)
{eggs} -> {french fries} (conf: 0.203, supp: 0.036, lift: 1.185, conv: 1.040)
{spaghetti} -> {eggs} (conf: 0.210, supp: 0.037, lift: 1.167, conv: 1.038)
{eggs} -> {spaghetti} (conf: 0.203, supp: 0.037, lift: 1.167, conv: 1.037)
{chocolate} -> {eggs} (conf: 0.203, supp: 0.033, lift: 1.127, conv: 1.029)
{green tea} -> {mineral water} (conf: 0.235, supp: 0.031, lift: 0.986, conv: 0.996)
```

```
for item in sorted(rules, key=lambda item: (item.support), reverse=True):
    print(item)
```

```
{spaghetti} -> {mineral water} (conf: 0.343, supp: 0.060, lift: 1.439, conv: 1.159)
{mineral water} -> {spaghetti} (conf: 0.251, supp: 0.060, lift: 1.439, conv: 1.102)
{mineral water} -> {chocolate} (conf: 0.221, supp: 0.053, lift: 1.348, conv: 1.073)
{chocolate} -> {mineral water} (conf: 0.321, supp: 0.053, lift: 1.348, conv: 1.122)
{mineral water} -> {eggs} (conf: 0.214, supp: 0.051, lift: 1.189, conv: 1.043)
{eggs} -> {mineral water} (conf: 0.283, supp: 0.051, lift: 1.189, conv: 1.063)
{mineral water} -> {milk} (conf: 0.201, supp: 0.048, lift: 1.554, conv: 1.090)
{milk} -> {mineral water} (conf: 0.370, supp: 0.048, lift: 1.554, conv: 1.210)
{ground beef} -> {mineral water} (conf: 0.417, supp: 0.041, lift: 1.748, conv: 1.305)
{spaghetti} -> {chocolate} (conf: 0.225, supp: 0.039, lift: 1.374, conv: 1.079)
{chocolate} -> {spaghetti} (conf: 0.239, supp: 0.039, lift: 1.374, conv: 1.086)
{spaghetti} -> {ground beef} (conf: 0.225, supp: 0.039, lift: 2.291, conv: 1.164)
{ground beef} -> {spaghetti} (conf: 0.399, supp: 0.039, lift: 2.291, conv: 1.374)
{spaghetti} -> {eggs} (conf: 0.210, supp: 0.037, lift: 1.167, conv: 1.038)
{eggs} -> {spaghetti} (conf: 0.203, supp: 0.037, lift: 1.167, conv: 1.037)
{french fries} -> {eggs} (conf: 0.213, supp: 0.036, lift: 1.185, conv: 1.042)
{eggs} -> {french fries} (conf: 0.203, supp: 0.036, lift: 1.185, conv: 1.040)
{frozen vegetables} -> {mineral water} (conf: 0.375, supp: 0.036, lift: 1.572, conv:
{spaghetti} -> {milk} (conf: 0.204, supp: 0.035, lift: 1.572, conv: 1.093)
{milk} -> {spaghetti} (conf: 0.274, supp: 0.035, lift: 1.572, conv: 1.137)
{french fries} -> {chocolate} (conf: 0.201, supp: 0.034, lift: 1.228, conv: 1.047)
{chocolate} -> {french fries} (conf: 0.210, supp: 0.034, lift: 1.228, conv: 1.049)
{pancakes} -> {mineral water} (conf: 0.355, supp: 0.034, lift: 1.489, conv: 1.181)
{chocolate} -> {eggs} (conf: 0.203, supp: 0.033, lift: 1.127, conv: 1.029)
{milk} -> {chocolate} (conf: 0.248, supp: 0.032, lift: 1.513, conv: 1.112)
{green tea} -> {mineral water} (conf: 0.235, supp: 0.031, lift: 0.986, conv: 0.996)
{milk} -> {eggs} (conf: 0.238, supp: 0.031, lift: 1.322, conv: 1.076)
```

```
for item in sorted(rules, key=lambda item: (item.confidence), reverse=True):

    print(item)
```

```
{ground beef} -> {mineral water} (conf: 0.417, supp: 0.041, lift: 1.748, conv: 1.305)
{ground beef} -> {spaghetti} (conf: 0.399, supp: 0.039, lift: 2.291, conv: 1.374)
{frozen vegetables} -> {mineral water} (conf: 0.375, supp: 0.036, lift: 1.572, conv:
{milk} -> {mineral water} (conf: 0.370, supp: 0.048, lift: 1.554, conv: 1.210)
{pancakes} -> {mineral water} (conf: 0.355, supp: 0.034, lift: 1.489, conv: 1.181)
{spaghetti} -> {mineral water} (conf: 0.343, supp: 0.060, lift: 1.439, conv: 1.159)
{chocolate} -> {mineral water} (conf: 0.321, supp: 0.053, lift: 1.348, conv: 1.122)
{eggs} -> {mineral water} (conf: 0.283, supp: 0.051, lift: 1.189, conv: 1.063)
{milk} -> {spaghetti} (conf: 0.274, supp: 0.035, lift: 1.572, conv: 1.137)
```

```
{mineral water} -> {spaghetti} (conf: 0.251, supp: 0.060, lift: 1.439, conv: 1.102)
{milk} -> {chocolate} (conf: 0.248, supp: 0.032, lift: 1.513, conv: 1.112)
{chocolate} -> {spaghetti} (conf: 0.239, supp: 0.039, lift: 1.374, conv: 1.086)
{milk} -> {eggs} (conf: 0.238, supp: 0.031, lift: 1.322, conv: 1.076)
{green tea} -> {mineral water} (conf: 0.235, supp: 0.031, lift: 0.986, conv: 0.996)
{spaghetti} -> {chocolate} (conf: 0.225, supp: 0.039, lift: 1.374, conv: 1.079)
{spaghetti} -> {ground beef} (conf: 0.225, supp: 0.039, lift: 2.291, conv: 1.164)
{mineral water} -> {chocolate} (conf: 0.221, supp: 0.053, lift: 1.348, conv: 1.073)
{mineral water} -> {eggs} (conf: 0.214, supp: 0.051, lift: 1.189, conv: 1.043)
{french fries} -> {eggs} (conf: 0.213, supp: 0.036, lift: 1.185, conv: 1.042)
{chocolate} -> {french fries} (conf: 0.210, supp: 0.034, lift: 1.228, conv: 1.049)
{spaghetti} -> {eggs} (conf: 0.210, supp: 0.037, lift: 1.167, conv: 1.038)
{spaghetti} -> {milk} (conf: 0.204, supp: 0.035, lift: 1.572, conv: 1.093)
{eggs} -> {spaghetti} (conf: 0.203, supp: 0.037, lift: 1.167, conv: 1.037)
{chocolate} -> {eggs} (conf: 0.203, supp: 0.033, lift: 1.127, conv: 1.029)
{eggs} -> {french fries} (conf: 0.203, supp: 0.036, lift: 1.185, conv: 1.040)
{mineral water} -> {milk} (conf: 0.201, supp: 0.048, lift: 1.554, conv: 1.090)
{french fries} -> {chocolate} (conf: 0.201, supp: 0.034, lift: 1.228, conv: 1.047)
```

```
1 tabular = pd.DataFrame(rules[:])
2 tabular
```

```
1 tabular[0]
```

```
0     {chocolate} -> {eggs} (conf: 0.203, supp: 0.03...
1     {french fries} -> {chocolate} (conf: 0.201, su...
2     {chocolate} -> {french fries} (conf: 0.210, su...
3     {milk} -> {chocolate} (conf: 0.248, supp: 0.03...
4     {mineral water} -> {chocolate} (conf: 0.221, s...
5     {chocolate} -> {mineral water} (conf: 0.321, s...
6     {spaghetti} -> {chocolate} (conf: 0.225, supp:...
7     {chocolate} -> {spaghetti} (conf: 0.239, supp:...
8     {french fries} -> {eggs} (conf: 0.213, supp: 0...
9     {eggs} -> {french fries} (conf: 0.203, supp: 0...
10    {milk} -> {eggs} (conf: 0.238, supp: 0.031, li...
11    {mineral water} -> {eggs} (conf: 0.214, supp: ...
12    {eggs} -> {mineral water} (conf: 0.283, supp: ...
13    {spaghetti} -> {eggs} (conf: 0.210, supp: 0.03...
14    {eggs} -> {spaghetti} (conf: 0.203, supp: 0.03...
15    {frozen vegetables} -> {mineral water} (conf: ...
16    {green tea} -> {mineral water} (conf: 0.235, s...
17    {ground beef} -> {mineral water} (conf: 0.417,...
18    {spaghetti} -> {ground beef} (conf: 0.225, sup...
19    {ground beef} -> {spaghetti} (conf: 0.399, sup...
20    {mineral water} -> {milk} (conf: 0.201, supp: ...
21    {milk} -> {mineral water} (conf: 0.370, supp: ...
22    {spaghetti} -> {milk} (conf: 0.204, supp: 0.03...
23    {milk} -> {spaghetti} (conf: 0.274, supp: 0.03...
24    {pancakes} -> {mineral water} (conf: 0.355, su...
25    {spaghetti} -> {mineral water} (conf: 0.343, s...
26    {mineral water} -> {spaghetti} (conf: 0.251, s...
Name: 0, dtype: object
```

```
1
```