Assignment 6

DMBI

Roll No. – 191310132092

Batch – B2

In [1]:

```python
import numpy as np
import pandas as pd

df = pd.read_csv("https://raw.githubusercontent.com/viktree/curly-octo-chainsaw/master/BreadBasket_DMS.csv")
df
```

Out[1]:

	Date	Time	Transaction	Item
0	2016-10-30	09:58:11	1	Bread
1	2016-10-30	10:05:34	2	Scandinavian
2	2016-10-30	10:05:34	2	Scandinavian
3	2016-10-30	10:07:57	3	Hot chocolate
4	2016-10-30	10:07:57	3	Jam
...	...	...	...	...
21288	2017-04-09	14:32:58	9682	Coffee
21289	2017-04-09	14:32:58	9682	Tea
21290	2017-04-09	14:57:06	9683	Coffee
21291	2017-04-09	14:57:06	9683	Pastry
21292	2017-04-09	15:04:24	9684	Smoothies

21293 rows × 4 columns

In [4]:

```python
products = df['Item'].unique()
```

In [5]:

```python
dummy = pd.get_dummies(df['Item'])
df.drop(['Item'], inplace=True, axis=1)
df = df.join(dummy)
df
```

Out[5]:

	Date	Time	Transaction	Adjustment	Afternoon with the baker	Alfajores	Argentina Night	Art Tray	Bacon	Baguette	...	The BART	The Nomad	Tiffin	Toast	Tr
0	2016-10-30	09:58:11	1	0	0	0	0	0	0	0	...	0	0	0	0	
1	2016-10-30	10:05:34	2	0	0	0	0	0	0	0	...	0	0	0	0	
2	2016-10-30	10:05:34	2	0	0	0	0	0	0	0	...	0	0	0	0	
3	2016-10-30	10:07:57	3	0	0	0	0	0	0	0	...	0	0	0	0	
4	2016-10-30	10:07:57	3	0	0	0	0	0	0	0	...	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
21288	2017-04-09	14:32:58	9682	0	0	0	0	0	0	0	...	0	0	0	0	
21289	2017-04-09	14:32:58	9682	0	0	0	0	0	0	0	...	0	0	0	0	
21290	2017-04-09	14:57:06	9683	0	0	0	0	0	0	0	...	0	0	0	0	
21291	2017-04-09	14:57:06	9683	0	0	0	0	0	0	0	...	0	0	0	0	
21292	2017-04-09	15:04:24	9684	0	0	0	0	0	0	0	...	0	0	0	0	

21293 rows × 98 columns

In [6]:

```python
data = df.groupby(['Transaction', 'Date'])[products[:]].sum()
data = data.reset_index()[products]
data
```

Out[6]:

	Bread	Scandinavian	Hot chocolate	Jam	Cookies	Muffin	Coffee	Pastry	Medialuna	Tea	...	Coffee granules	Drinking chocolate spoons	Christmas common	Argentina Night
0	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0
1	0	2	0	0	0	0	0	0	0	0	...	0	0	0	0
2	0	0	1	1	1	0	0	0	0	0	...	0	0	0	0
3	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0
4	1	0	0	0	0	0	1	1	0	0	...	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9526	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0
9527	0	0	0	0	0	0	0	0	0	1	...	0	0	1	0
9528	0	0	0	0	0	1	1	0	0	1	...	0	0	0	0
9529	0	0	0	0	0	0	1	1	0	0	...	0	0	0	0
9530	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0

9531 rows × 95 columns

In [7]:

```python
def product(x):
    for product in products:
        if x[product]>0:
            x[product] = product
    return x
```

In [8]:

```python
data = data.apply(product, axis=1)
data
```

Out[8]:

	Bread	Scandinavian	Hot chocolate	Jam	Cookies	Muffin	Coffee	Pastry	Medialuna	Tea	...	Coffee granules	Drinking chocolate spoons	Christmas common	Argentina Night
0	Bread	0	0	0	0	0	0	0	0	0	...	0	0	0	0
1	0	Scandinavian	0	0	0	0	0	0	0	0	...	0	0	0	0
2	0	0	Hot chocolate	Jam	Cookies	0	0	0	0	0	...	0	0	0	0
3	0	0	0	0	0	Muffin	0	0	0	0	...	0	0	0	0
4	Bread	0	0	0	0	0	Coffee	Pastry	0	0	...	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9526	Bread	0	0	0	0	0	0	0	0	0	...	0	0	0	0
9527	0	0	0	0	0	0	0	0	0	Tea	...	0	0	Christmas common	0
9528	0	0	0	0	0	Muffin	Coffee	0	0	Tea	...	0	0	0	0
9529	0	0	0	0	0	0	Coffee	Pastry	0	0	...	0	0	0	0
9530	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0

9531 rows × 95 columns

In [9]:

```python
x = data.values
x = [sub[~(sub==0)].tolist() for sub in x if sub[sub != 0].tolist()]
transactions = x
transactions[:100]
```

Out[9]:

```
[['Bread'],
 ['Scandinavian'],
 ['Hot chocolate', 'Jam', 'Cookies'],
 ['Muffin'],
 ['Bread', 'Coffee', 'Pastry'],
 ['Muffin', 'Pastry', 'Medialuna'],
 ['Coffee', 'Pastry', 'Medialuna', 'Tea'],
 ['Bread', 'Pastry'],
 ['Bread', 'Muffin'],
 ['Scandinavian', 'Medialuna'],
 ['Bread', 'Medialuna', 'NONE'],
 ['Jam', 'Coffee', 'Pastry', 'Tea', 'Tartine'],
 ['Bread', 'Coffee', 'Basket'],
 ['Bread', 'Pastry', 'Medialuna'],
 ['Scandinavian', 'NONE', 'Mineral water'],
 ['Bread', 'Coffee', 'Medialuna'],
 ['Hot chocolate'],
 ['Farm House'],
 ['Bread', 'Farm House'],
 ['Bread', 'Medialuna'],
 ['Bread', 'Coffee', 'Medialuna'],
 ['Jam'],
 ['Scandinavian', 'Muffin'],
 ['Bread'],
 ['Scandinavian'],
 ['Fudge'],
 ['Scandinavian'],
 ['Bread', 'Coffee'],
 ['Bread', 'Jam', 'NONE'],
 ['Bread'],
 ['Basket'],
 ['Scandinavian', 'Muffin'],
 ['Coffee'],
 ['Muffin', 'Coffee'],
 ['Scandinavian', 'Muffin'],
 ['Bread', 'Tea'],
 ['Bread', 'Coffee', 'NONE'],
 ['Bread', 'Tea'],
 ['Scandinavian'],
 ['Muffin', 'Coffee', 'NONE', 'Tartine', 'Juice'],
 ['Scandinavian'],
 ['Bread', 'Tea'],
 ['Scandinavian', 'Fudge'],
 ['Coffee', 'Medialuna'],
 ['Hot chocolate', 'Coffee', 'Medialuna'],
 ['Coffee'],
 ['Bread', 'Jam', 'Muffin', 'Juice', "Ella's Kitchen Pouches"],
 ['Coffee'],
 ['Coffee', 'Medialuna'],
 ['Bread', 'Victorian Sponge'],
 ['Bread'],
 ['Scandinavian'],
 ['Bread'],
 ['Coffee', 'Tea', 'Frittata', 'Hearty & Seasonal'],
 ['Coffee', 'Frittata'],
 ['Scandinavian'],
 ['Hot chocolate', 'Tea', 'Victorian Sponge', 'Soup'],
 ['Tea', 'NONE'],
 ['Cookies', 'Coffee', 'Juice'],
 ['Coffee'],
 ['Coffee', 'Hearty & Seasonal', 'Pick and Mix Bowls', 'Smoothies'],
 ['Coffee'],
 ['Cake'],
 ['Coffee', 'Tea', 'NONE', 'Tartine', 'Mighty Protein'],
 ['Mineral water', 'Frittata', 'Hearty & Seasonal'],
 ['Muffin', 'NONE', 'Mineral water', 'Hearty & Seasonal'],
 ['Scandinavian', 'Coffee', 'Tea', 'Frittata', 'Chicken sand'],
 ['Bread', 'Tea', 'Victorian Sponge'],
 ['Fudge'],
 ['Muffin'],
 ['Bread', 'Coffee'],
 ['Bread'],
 ['Bread', 'Coffee'],
 ['Jam', 'NONE', 'Frittata'],
 ['Coffee'],
 ['Scandinavian'],
 ['Fudge'],
 ['Muffin', 'Coffee', 'Tea', 'Fudge'],
 ['Bread', 'Coffee', 'Frittata'],
 ['Coffee', 'Cake'],
 ['Bread', 'NONE', 'Tartine'],
 ['Bread', 'Coffee'],
 ['Bread'],
 ['Coffee', 'Pastry', 'Medialuna'],
 ['Juice'],
 ['Bread'],
 ['Jam', 'Coffee'],
 ['Bread'],
 ['Bread', 'Coffee'],
 ['Tea', 'NONE'],
 ['Coffee'],
 ['Bread', 'Coke'],
 ['Coffee'],
 ['Pastry', 'Tea', 'Farm House'],
 ['Coffee', 'Pastry', 'Juice'],
 ['Coffee', 'Pastry', 'Juice'],
 ['Farm House'],
 ['Bread', 'Coffee', 'Pastry'],
 ['Bread'],
 ['Coffee', 'Pastry']]
```

In [10]:

```python
from apyori import apriori
rules = apriori(transactions, min_support = 0.04, min_confidence = 0.5, max_length = 4, target = "rules")
association_results = list(rules)
print(association_results[0])
```

```
RelationRecord(items=frozenset({'Coffee', 'Cake'}), support=0.054348966530269646, ordered_statistics=[OrderedSt
atistic(items_base=frozenset({'Cake'}), items_add=frozenset({'Coffee'}), confidence=0.5269582909460834, lift=1.
1091959962471556)])
```

In [11]:

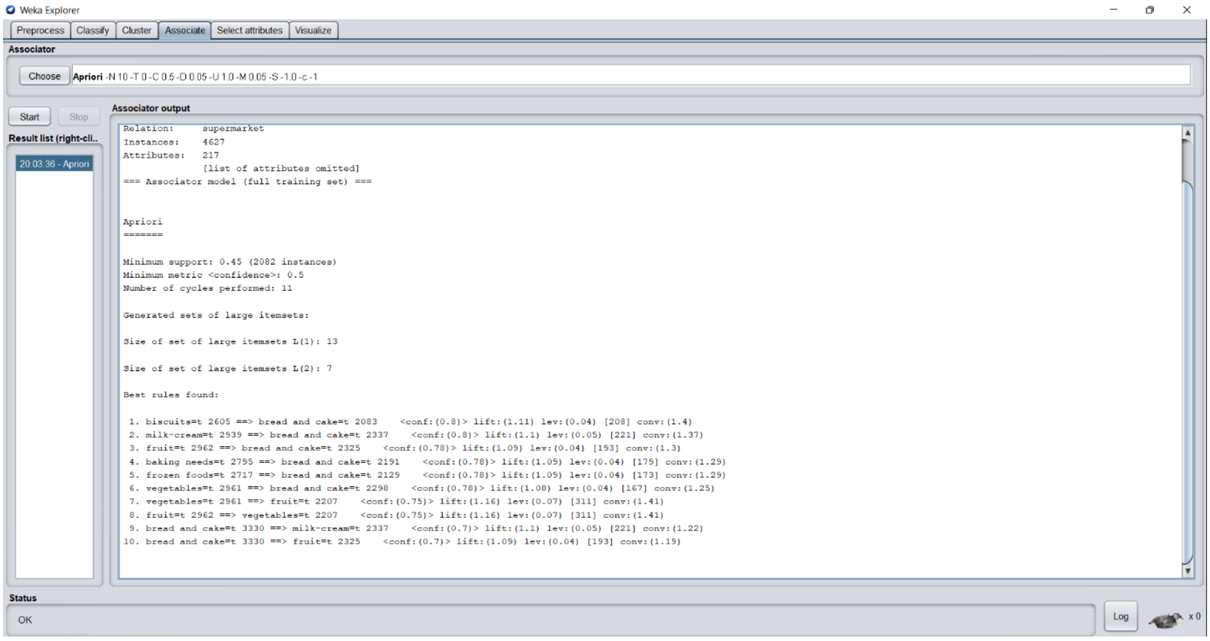
```python
for item in association_results:
    pair = item[0]
    items = [x for x in pair]

    print("Rules : ", items[0], " ----> " + items[1])
    print("Support : ", str(item[1]))
    print("Confidence : ", str(item[2][0][2]))
    print("Lift : ", str(item[2][0][3]))
    print("+----------------------------------------+")
```

```
Rules :  Coffee  ----> Cake
Support :  0.054348966530269646
Confidence :  0.5269582909460834
Lift :  1.1091959962471556
+----------------------------------------+
Rules :  NONE  ----> Coffee
Support :  0.042073234707795615
Confidence :  0.5325365205843293
Lift :  1.1209376275815466
+----------------------------------------+
Rules :  Coffee  ----> Pastry
Support :  0.047214353163361665
Confidence :  0.5521472392638037
Lift :  1.1622162847666326
+----------------------------------------+
```

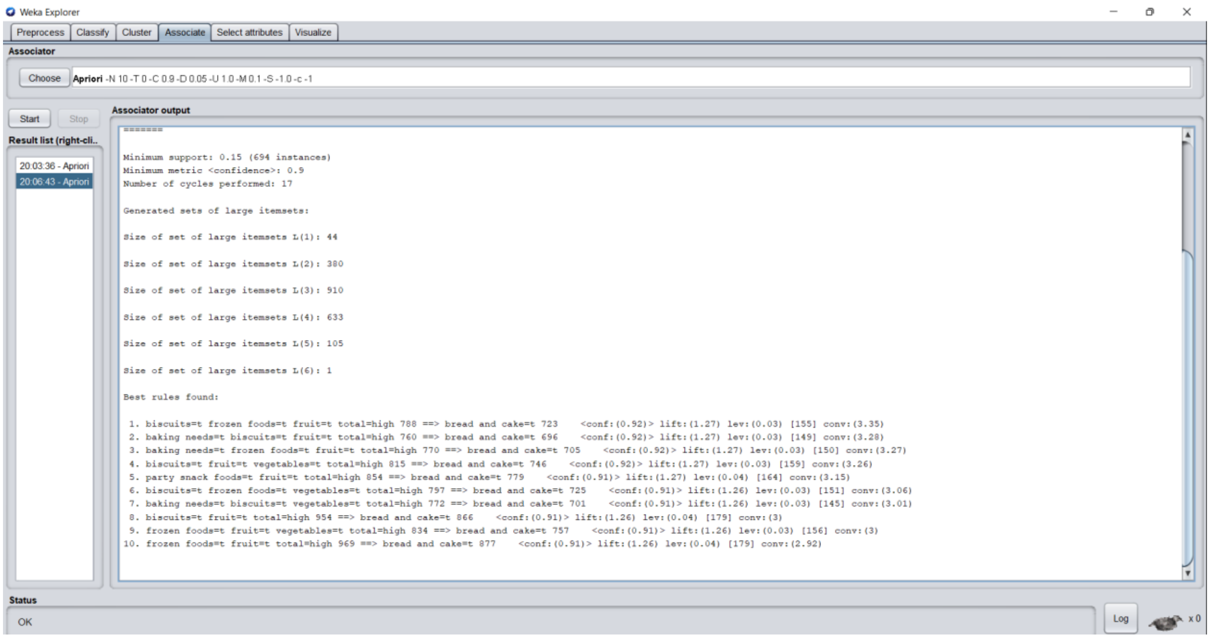
Weka

Supermarket Dataset

1) Here min_support = 0.45 and min_confidence = 0.5.
   And to generate rule total 11 cycles are performed.

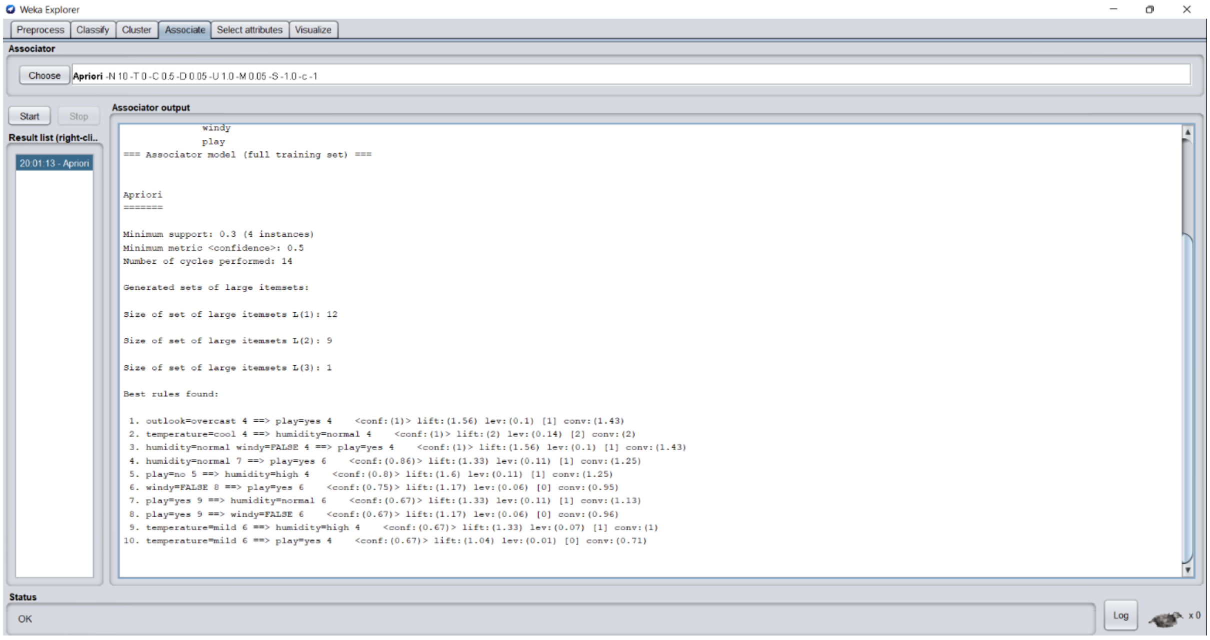


2) Here min_support = 0.15 and min_confidence = 0.9.
   And to generate rule total 17 cycles are performed.



Weather.nominal Dataset

1) Here min_support = 0.3 and min_confidence = 0.5.
   And to generate rule total 14 cycles are performed.



2) Here min_support = 0.15 and min_confidence = 0.9.
   And to generate rule total 17 cycles are performed.