1821221005 FURKAN GÜNDOĞAN

APRIORI ALGORITHM -> ASSOCIATION RULE ANALYSIS

1.) Data Understanding

I created a Stationery Products Dataset.

Reading data and naming column to "products".

In [1]:
```
import pandas as pd
df = pd.read_csv("StationeryDataSet.csv", names=['products'], header = None)
```

Displaying first five data and column length.

In [2]:
```
df.head()
```

Out[2]:

	products
0	Notebook,Pen
1	Eraser,Pencil
2	Eraser,Pencil
3	Ink,Glue,Cardboard
4	Glue,Cardboard

Displaying column length and column count to understand dataset size.

In [3]:
```
df.shape
```

Out[3]:
```
(55, 1)
```

Displaying column name and data-type.

In [4]:
```
df.columns
```

Out[4]:
```
Index(['products'], dtype='object')
```

Displaying all values of dataset

In [5]:
```
df.values
```

Out[5]:
```
array([['Notebook,Pen'],
       ['Eraser,Pencil'],
       ['Eraser,Pencil'],
       ['Ink,Glue,Cardboard'],
       ['Glue,Cardboard'],
       ['Tape,Painting Book'],
       ['Marker,Painting Book,Eraser,Pencil'],
       ['Marker,Colored Pens Set,Painting Book,Glue'],
       ['Ink,Glue,Cardboard'],
```

```
       ['Painting Book,Ink,Cardboard,Colored Pens Set'],
       ['Marker,Ink,Painting Book,Pencil'],
       ['Notebook, Pencil'],
       ['Notebook,Pen,Glue,Painting Book,Marker'],
       ['Eraser,Pen'],
       ['Tape,Marker,Colored Pens Set,Glue,Cardboard,Scissors'],
       ['Ink,Cardboard,Glue,Colored Pens Set'],
       ['Scissors,Tape'],
       ['Scissors,Tape'],
       ['Scissors,Marker,Colored Pens Set,Painting Book'],
       ['Notebook,Scissors,Tape'],
       ['Notebook,Colored Pens Set,Pen,Glue,Painting Book,Marker'],
       ['Colored Pens Set,Glue'],
       ['Notebook,Ink,Pen,Colored Pens Set'],
       ['Glue,Notebook,Eraser'],
       ['Eraser,Pencil'],
       ['Scissors,Pencil'],
       ['Marker,Ink,Glue,Tape'],
       ['Cardboard,Glue,Ink,Notebook,Scissors,Tape'],
       ['Scissors,Colored Pens Set,Painting Book'],
       ['Eraser,Pen,Cardboard'],
       ['Ink,Marker,Glue,Colored Pens Set'],
       ['Notebook,Pen,Scissors'],
       ['Notebook,Pen,Tape,Glue'],
       ['Ink,Pen,Colored Pens Set'],
       ['Glue,Notebook'],
       ['Eraser,Pencil'],
       ['Painting Book,Cardboard,Colored Pens Set'],
       ['Painting Book,Cardboard,Colored Pens Set'],
       ['Painting Book,Cardboard,Colored Pens Set,Glue'],
       ['Marker,Ink'],
       ['Notebook, Pencil,Pen,Eraser'],
       ['Glue,Cardboard'],
       ['Glue,Cardboard,Scissors,Pen'],
       ['Glue,Pen'],
       ['Eraser,Pencil,Pen'],
       ['Eraser,Pencil,Pen'],
       ['Notebook,Pen'],
       ['Notebook,Pen'],
       ['Pencil,Pen'],
       ['Eraser,Glue'],
       ['Eraser,Glue'],
       ['Ink,Glue,Cardboard'],
       ['Glue,Notebook'],
       ['Scissors,Tape,Marker'],
       ['Scissors,Tape,Marker']], dtype=object)
```

Dataset has been converted to a list which every element also a list.

In [6]:

```
data = list(df["products"].apply(lambda x:x.split(',')))
data
```

Out[6]:

```
[['Notebook', 'Pen'],
 ['Eraser', 'Pencil'],
 ['Eraser', 'Pencil'],
 ['Ink', 'Glue', 'Cardboard'],
 ['Glue', 'Cardboard'],
 ['Tape', 'Painting Book'],
 ['Marker', 'Painting Book', 'Eraser', 'Pencil'],
 ['Marker', 'Colored Pens Set', 'Painting Book', 'Glue'],
 ['Ink', 'Glue', 'Cardboard'],
 ['Painting Book', 'Ink', 'Cardboard', 'Colored Pens Set'],
 ['Marker', 'Ink', 'Painting Book', 'Pencil'],
 ['Notebook', ' Pencil'],
 ['Notebook', 'Pen', 'Glue', 'Painting Book', 'Marker'],
```

```
['Eraser', 'Pen'],
['Tape', 'Marker', 'Colored Pens Set', 'Glue', 'Cardboard', 'Scissors'],
['Ink', 'Cardboard', 'Glue', 'Colored Pens Set'],
['Scissors', 'Tape'],
['Scissors', 'Tape'],
['Scissors', 'Marker', 'Colored Pens Set', 'Painting Book'],
['Notebook', 'Scissors', 'Tape'],
['Notebook', 'Colored Pens Set', 'Pen', 'Glue', 'Painting Book', 'Marker'],
['Colored Pens Set', 'Glue'],
['Notebook', 'Ink', 'Pen', 'Colored Pens Set'],
['Glue', 'Notebook', 'Eraser'],
['Eraser', 'Pencil'],
['Scissors', 'Pencil'],
['Marker', 'Ink', 'Glue', 'Tape'],
['Cardboard', 'Glue', 'Ink', 'Notebook', 'Scissors', 'Tape'],
['Scissors', 'Colored Pens Set', 'Painting Book'],
['Eraser', 'Pen', 'Cardboard'],
['Ink', 'Marker', 'Glue', 'Colored Pens Set'],
['Notebook', 'Pen', 'Scissors'],
['Notebook', 'Pen', 'Tape', 'Glue'],
['Ink', 'Pen', 'Colored Pens Set'],
['Glue', 'Notebook'],
['Eraser', 'Pencil'],
['Painting Book', 'Cardboard', 'Colored Pens Set'],
['Painting Book', 'Cardboard', 'Colored Pens Set'],
['Painting Book', 'Cardboard', 'Colored Pens Set', 'Glue'],
['Marker', 'Ink'],
['Notebook', ' Pencil', 'Pen', 'Eraser'],
['Glue', 'Cardboard'],
['Glue', 'Cardboard', 'Scissors', 'Pen'],
['Glue', 'Pen'],
['Eraser', 'Pencil', 'Pen'],
['Eraser', 'Pencil', 'Pen'],
['Notebook', 'Pen'],
['Notebook', 'Pen'],
['Pencil', 'Pen'],
['Eraser', 'Glue'],
['Eraser', 'Glue'],
['Ink', 'Glue', 'Cardboard'],
['Glue', 'Notebook'],
['Scissors', 'Tape', 'Marker'],
['Scissors', 'Tape', 'Marker']]
```

2.) Data Preprocessing

In [7]:
```
# !pip install mlxtend -   i did installation only once, before starting this task.
```

In [8]:
```
from mlxtend.preprocessing import TransactionEncoder
```

Using True-False array, I was able to determine which product was purchased for each purchase.

In [9]:
```
te = TransactionEncoder()
te_data = te.fit(data).transform(data)
df = pd.DataFrame(te_data,columns=te.columns_)
df
```

Out[9]:

	Pencil	Cardboard	Colored Pens Set	Eraser	Glue	Ink	Marker	Notebook	Painting Book	Pen	Pencil	Scissors	Tape

	Pencil	Cardboard	Colored Pens Set	Eraser	Glue	Ink	Marker	Notebook	Painting Book	Pen	Pencil	Scissors	Tape
0	False	False	False	False	False	False	False	True	False	True	False	False	False
1	False	False	False	True	False	False	False	False	False	False	True	False	False
2	False	False	False	True	False	False	False	False	False	False	True	False	False
3	False	True	False	False	True	True	False	False	False	False	False	False	False
4	False	True	False	False	True	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	True	False	False	False	True
6	False	False	False	True	False	False	True	False	True	False	True	False	False
7	False	False	True	False	True	False	True	False	True	False	False	False	False
8	False	True	False	False	True	True	False	False	False	False	False	False	False
9	False	True	True	False	False	True	False	False	True	False	False	False	False
10	False	False	False	False	False	True	True	False	True	False	True	False	False
11	True	False	False	False	False	False	False	True	False	False	False	False	False
12	False	False	False	False	True	False	True	True	True	True	False	False	False
13	False	False	False	True	False	False	False	False	False	True	False	False	False
14	False	True	True	False	True	False	True	False	False	False	False	True	True
15	False	True	True	False	True	True	False	False	False	False	False	False	False
16	False	False	False	False	False	False	False	False	False	False	False	True	True
17	False	False	False	False	False	False	False	False	False	False	False	True	True
18	False	False	True	False	False	False	True	False	True	False	False	True	False
19	False	False	False	False	False	False	False	True	False	False	False	True	True
20	False	False	True	False	True	False	True	True	True	True	False	False	False
21	False	False	True	False	True	False	False	False	False	False	False	False	False
22	False	False	True	False	False	True	False	True	False	True	False	False	False
23	False	False	False	True	True	False	False	True	False	False	False	False	False
24	False	False	False	True	False	False	False	False	False	False	True	False	False
25	False	False	False	False	False	False	False	False	False	False	True	True	False
26	False	False	False	False	True	True	True	False	False	False	False	False	True
27	False	True	False	False	True	True	False	True	False	False	False	True	True
28	False	False	True	False	False	False	False	False	True	False	False	True	False
29	False	True	False	True	False	False	False	False	False	True	False	False	False
30	False	False	True	False	True	True	True	False	False	False	False	False	False
31	False	False	False	False	False	False	False	True	False	True	False	True	False
32	False	False	False	False	True	False	False	True	False	True	False	False	True
33	False	False	True	False	False	True	False	False	False	True	False	False	False
34	False	False	False	False	True	False	False	True	False	False	False	False	False
35	False	False	False	True	False	False	False	False	False	False	True	False	False

	Pencil	Cardboard	Colored Pens Set	Eraser	Glue	Ink	Marker	Notebook	Painting Book	Pen	Pencil	Scissors	Tape
36	False	True	True	False	False	False	False	False	True	False	False	False	False
37	False	True	True	False	False	False	False	False	True	False	False	False	False
38	False	True	True	False	True	False	False	False	True	False	False	False	False
39	False	False	False	False	False	True	True	False	False	False	False	False	False
40	True	False	False	True	False	False	False	True	False	True	False	False	False
41	False	True	False	False	True	False	False	False	False	False	False	False	False
42	False	True	False	False	True	False	False	False	False	True	False	True	False
43	False	False	False	False	True	False	False	False	False	True	False	False	False
44	False	False	False	True	False	False	False	False	False	True	True	False	False
45	False	False	False	True	False	False	False	False	False	True	True	False	False
46	False	False	False	False	False	False	False	True	False	True	False	False	False
47	False	False	False	False	False	False	False	True	False	True	False	False	False
48	False	False	False	False	False	False	False	False	False	True	True	False	False
49	False	False	False	True	True	False	False	False	False	False	False	False	False
50	False	False	False	True	True	False	False	False	False	False	False	False	False
51	False	True	False	False	True	True	False	False	False	False	False	False	False
52	False	False	False	False	True	False	False	True	False	False	False	False	False
53	False	False	False	False	False	False	True	False	False	False	False	True	True
54	False	False	False	False	False	False	True	False	False	False	False	True	True

3.) Data Association Rules

In [10]:
```
from mlxtend.frequent_patterns import apriori
```

Apriori:

I filtered dataset according to "support"

Then display sorted version of list

In [11]:
```
df1 = apriori(df, min_support=0.05, use_colnames=True, verbose = 1)
df1 = df1.sort_values(['support'], ascending =[False])
df1
```

```
Processing 132 combinations | Sampling itemset size 4
```

Out[11]:

	support	itemsets
3	0.418182	(Glue)
8	0.309091	(Pen)
6	0.272727	(Notebook)
0	0.254545	(Cardboard)

	support	itemsets
1	0.254545	(Colored Pens Set)
2	0.236364	(Eraser)
4	0.218182	(Ink)
5	0.218182	(Marker)
7	0.218182	(Painting Book)
10	0.218182	(Scissors)
13	0.181818	(Cardboard, Glue)
9	0.181818	(Pencil)
11	0.181818	(Tape)
37	0.163636	(Pen, Notebook)
20	0.145455	(Colored Pens Set, Painting Book)
41	0.127273	(Scissors, Tape)
28	0.127273	(Glue, Notebook)
25	0.127273	(Eraser, Pencil)
26	0.127273	(Glue, Ink)
17	0.127273	(Colored Pens Set, Glue)
12	0.109091	(Cardboard, Colored Pens Set)
27	0.109091	(Marker, Glue)
14	0.109091	(Cardboard, Ink)
34	0.109091	(Marker, Painting Book)
19	0.090909	(Marker, Colored Pens Set)
44	0.090909	(Cardboard, Glue, Ink)
24	0.090909	(Pen, Eraser)
30	0.090909	(Pen, Glue)
18	0.090909	(Colored Pens Set, Ink)
33	0.072727	(Marker, Ink)
46	0.072727	(Marker, Colored Pens Set, Glue)
43	0.072727	(Cardboard, Painting Book, Colored Pens Set)
36	0.072727	(Tape, Marker)
35	0.072727	(Scissors, Marker)
32	0.072727	(Tape, Glue)
29	0.072727	(Painting Book, Glue)
15	0.072727	(Cardboard, Painting Book)
22	0.054545	(Scissors, Colored Pens Set)
50	0.054545	(Pen, Glue, Notebook)
49	0.054545	(Marker, Painting Book, Glue)

	support	itemsets
48	0.054545	(Marker, Colored Pens Set, Painting Book)
47	0.054545	(Colored Pens Set, Painting Book, Glue)
21	0.054545	(Pen, Colored Pens Set)
45	0.054545	(Scissors, Cardboard, Glue)
23	0.054545	(Eraser, Glue)
31	0.054545	(Scissors, Glue)
42	0.054545	(Cardboard, Glue, Colored Pens Set)
16	0.054545	(Scissors, Cardboard)
40	0.054545	(Pen, Pencil)
39	0.054545	(Tape, Notebook)
38	0.054545	(Scissors, Notebook)
51	0.054545	(Scissors, Marker, Tape)

Results:

-> 4 out of 10 purchases have at least one "Glue".

-> There is a "Pen" in 30 % of purchases.

-> There is a "Notebook" in 27 % of purchases.

-> "Glue" and "Cardboard" were been together in 18 % of purchases.

I filtered dataset according to "confidence"

"Confidence" gives the probability that the second product will be sold when the first is sold

In [12]:
```
from mlxtend.frequent_patterns import association_rules
```

In [13]:
```
association_rules(df1, metric = "confidence", min_threshold = 0.70)
```

Out[13]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Cardboard)	(Glue)	0.254545	0.418182	0.181818	0.714286	1.708075	0.075372	2.036364
1	(Tape)	(Scissors)	0.181818	0.218182	0.127273	0.700000	3.208333	0.087603	2.606061
2	(Pencil)	(Eraser)	0.181818	0.236364	0.127273	0.700000	2.961538	0.084298	2.545455
3	(Cardboard, Ink)	(Glue)	0.109091	0.418182	0.090909	0.833333	1.992754	0.045289	3.490909
4	(Glue, Ink)	(Cardboard)	0.127273	0.254545	0.090909	0.714286	2.806122	0.058512	2.609091
5	(Marker, Colored Pens Set)	(Glue)	0.090909	0.418182	0.072727	0.800000	1.913043	0.034711	2.909091

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
6	(Cardboard, Painting Book)	(Colored Pens Set)	0.072727	0.254545	0.072727	1.000000	3.928571	0.054215	inf
7	(Painting Book, Glue)	(Marker)	0.072727	0.218182	0.054545	0.750000	3.437500	0.038678	3.127273
8	(Painting Book, Glue)	(Colored Pens Set)	0.072727	0.254545	0.054545	0.750000	2.946429	0.036033	2.981818
9	(Scissors, Cardboard)	(Glue)	0.054545	0.418182	0.054545	1.000000	2.391304	0.031736	inf
10	(Scissors, Glue)	(Cardboard)	0.054545	0.254545	0.054545	1.000000	3.928571	0.040661	inf
11	(Scissors, Marker)	(Tape)	0.072727	0.181818	0.054545	0.750000	4.125000	0.041322	3.272727
12	(Marker, Tape)	(Scissors)	0.072727	0.218182	0.054545	0.750000	3.437500	0.038678	3.127273

Results:

-> Customers who buy "Cardboard" are 71 percent likely to buy "Glue" as well.

-> 3 of the 4 customers who buy "Pencil" also buy "Eraser".

-> 3 of the 4 customers who buy "Tape" also buy "Scissors".

-> If a customer buys the "Cardboard, Scissors" duo, then definitely gets "Glue" as well.

-> If a customer buys the "Cardboard, Painting Book" duo, then definitely gets "Colored Pen Set" as well.

4.) Selecting and Filtering Results

with combinations of "Support" and "Confidence" metrics

a.) Confidence >= 0.7 and Support >=0.125

In [14]:
```
rules = association_rules(df1, metric = "confidence", min_threshold = 0)
rules[ (rules['confidence'] >= 0.7) & (rules['support'] >= 0.125) ]
```

Out[14]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Cardboard)	(Glue)	0.254545	0.418182	0.181818	0.714286	1.708075	0.075372	2.036364
7	(Tape)	(Scissors)	0.181818	0.218182	0.127273	0.700000	3.208333	0.087603	2.606061
11	(Pencil)	(Eraser)	0.181818	0.236364	0.127273	0.700000	2.961538	0.084298	2.545455

Reporting

```
Probabilities of being seen alone in shopping; "Pencil": 18% , "Eraser": 23%
"Pencil" and "Eraser" are purchased together in 12 out of 100 purchases.
```

When "Pencil" is sold, the probability of selling "Eraser" is 70%
In sales where "Pencil" is sold, the probability of selling "Eraser" increases 2.96 times.
Action Idea:
We can increase customer satisfaction by making discounts in sales where both products are purchased

Probabilities of being seen alone in shopping; "Cardboard": 25% , "Glue": 41%
"Cardboard" and "Glue" are purchased together in 18 out of 100 purchases.
When "Cardboard" is sold, the probability of selling "Glue" is 71%
In sales where "Cardboard" is sold, the probability of selling "Glue" increases 1.7 times.
Action Idea:
    We can put these products away from each other, that makes customers
    spends more time in our stationery and increase possibility of buying different stuffs.

Probabilities of being seen alone in shopping; "Tape": 18% , "Scissors": 21%
"Tape" and "Scissors" are purchased together in 12 out of 100 purchases.
When "Tape" is sold, the probability of selling "Scissors" is 70%
In sales where "Tape" is sold, the probability of selling "Scissors" increases 3.2 times.
Action Idea:
The lift value of 3.2 is the highest value between the two products.
To get more income we should work on increase the sales of "Tape";
We can make campaigns such as buy three, pay for two.

b.) Confidence >= 1 and Support >=0.05 and Lift => 3

---- Higher confidence, lower support

In [15]:
```
rules = association_rules(df1, metric = "confidence", min_threshold = 0)
rules[ (rules['confidence'] >= 1) & (rules['support'] >= 0.05) & (rules['lift'] >= 3) ]
```

Out[15]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
46	(Cardboard, Painting Book)	(Colored Pens Set)	0.072727	0.254545	0.072727	1.0	3.928571	0.054215	inf
91	(Scissors, Glue)	(Cardboard)	0.054545	0.254545	0.054545	1.0	3.928571	0.040661	inf

Reporting

"Scissors, Glue" duo are in 7% of purchases. "Cardboard" is in 25% of purchases.
They are together in 5% of purchases.
When "Scissors, Glue" duo are sold, the probability of selling "Cardboard" is 100%
In sales where "Scissors, Glue" duo are sold, the probability of selling "Cardboard" increases 3.92 times.
Action Idea:
    If we want higher profit from this relationship which has 1 confidence ratio,
    we must increase "Scissors, Glue" duo sales.
    We can make a package discount that including scissors and glue.
    Then we can get benefit from "lift" value of selling "Cardboard" too.

"Cardboard, Painting Book" duo are in 7% of purchases. "Colored Pens Set" is in 25% of purchases.
They are together in 7% of purchases.
When "Cardboard, Painting Book" duo are sold, the probability of selling "Colored Pens Set" is 100%
In sales where "Cardboard, Painting Book" duo are sold,
the probability of selling "Colored Pens Set" increases 3.92 times.
Action Idea:
    This relation also has high confidence and lift.
    But "Cardboard, Painting Book" duo sales are very low.
    This cause can be price of Painting Book.
    If we can set good discount on Painting Book we can get more profit.
    Because customers who buy them are very likely to buy "Color Pencil Set" as well.
    Discount should be determined by according to the income to be obtained by "lift" value of "Color Pencil Set".

In [ ]:

In [ ]: