<u>Aim</u>: Experiment to implement Association mining algorithm(Apriori) using Rapid Miner and Python.

<u>To Do:</u>

1. Preprocess data.
2. Build Association Mining model using inbuilt library function on training data
3. Calculate metrics using inbuilt function
4. Build Association Mining model using Rapid Miner
5. Calculate metrics using Rapid Miner
6. Compare the results of both implementations.

<u>Theory:</u>

Apriori algorithm refers to the algorithm which is used to calculate the association rules between objects. It means how two or more objects are related to one another. In other words, we can say that the apriori algorithm is an association rule learning that analyzes that people who bought product A also bought product B. The primary objective of the apriori algorithm is to create the association rule between different objects. The association rule describes how two or more objects are related to one another. Apriori algorithm is also called frequent pattern mining. Generally, you operate the Apriori algorithm on a database that consists of a huge number of transactions.

Frequent Pattern Mining (FPM)

The frequent pattern mining algorithm is one of the most important techniques of data mining to discover relationships between different items in a dataset. These relationships are represented in the form of association rules. It helps to find the irregularities in data.

FPM has many applications in the field of data analysis, software bugs, cross-marketing, sale campaign analysis, market basket analysis, etc.

Association rules apply to supermarket transaction data, that is, to examine the customer behavior in terms of the purchased products. Association rules describe how often the items are purchased together.

Association Rules

Association Rule Mining is defined as:

"Let I= { …} be a set of 'n' binary attributes called items. Let D= { ….} be set of transaction called databases. Each transaction in D has a unique transaction ID and contains a subset of the items in I. A rule is defined as an implication of form X->Y where X, Y? I and X?Y=?. The set of items X and Y are called antecedent and consequent of the rule respectively."

Learning of Association rules is used to find relationships between attributes in large databases. An association rule, A=> B, will be of the form" for a set of transactions, some value of itemset A determines the values of itemset B under the condition in which minimum support and confidence are met".

Support and Confidence can be represented by the following example:
Bread=> butter [support=2%, confidence-60%]
The above statement is an example of an association rule. This means that there is a 2% transaction that bought bread and butter together and there are 60% of customers who bought bread as well as butter.

Support and Confidence for Itemset A and B are represented by formulas:

$$\text{Support (A)} = \frac{\text{Number of transaction in which A appears}}{\text{Total number of transactions}}$$

$$\text{Confidence } (A \longrightarrow B) = \frac{\text{Support(AUB)}}{\text{Support(A)}}$$

Association rule mining consists of 2 steps:

1. Find all the frequent itemsets.
2. Generate association rules from the above frequent itemsets.

Implementation using Rapid Miner

Step 1: We have this dataset in which 1 donates that product is present in bill and 0 donates it does not.

Open in Turbo Prep Auto Model Filter (50 / 50 examples): all

Row No.	CAKE	MILK	BREAD	BISCUIT	CORNFLAKES	JAM	MANGO	TEA	COF
1	1	1	1	1	0	1	1	0	0
2	0	0	0	1	1	0	1	0	0
3	1	1	0	1	0	0	1	0	0
4	0	0	0	0	1	1	1	0	0
5	1	1	1	1	1	1	1	1	1
6	0	0	0	0	0	0	0	0	0
7	1	1	1	1	1	1	1	1	1
8	0	0	0	0	0	0	0	0	0
9	1	1	1	1	1	1	1	1	1

Step 2: We use fp growth operator to show the growth

Step 3: We use numerical to binomial operator to convert our data in true and false format

Row No.	CAKE	MILK	BREAD	BISCUIT	CORNFLAKES	JAM	MANGO	TEA	COF
1	true	true	true	true	false	true	true	false	false
2	false	false	false	true	true	false	true	false	false
3	true	true	false	true	false	false	true	false	false
4	false	false	false	false	true	true	true	false	false
5	true	true	true	true	true	true	true	true	true
6	false	false	false	false	false	false	false	false	false
7	true	true	true	true	true	true	true	true	true
8	false	false	false	false	false	false	false	false	false

Step 4: We get support of each item set

Size	Support	Item 1	Item 2
1	0.640	MANGO	
1	0.620	BISCUIT	
1	0.620	CAKE	
1	0.600	CORNFLAKES	
1	0.600	JAM	
1	0.580	BREAD	
1	0.480	MILK	
1	0.460	COFFEE	
1	0.460	SALT	
1	0.460	SUGAR	
1	0.420	TEA	
2	0.620	MANGO	BISCUIT
2	0.540	MANGO	CAKE
2	0.600	MANGO	CORNFLAKES
2	0.600	MANGO	JAM

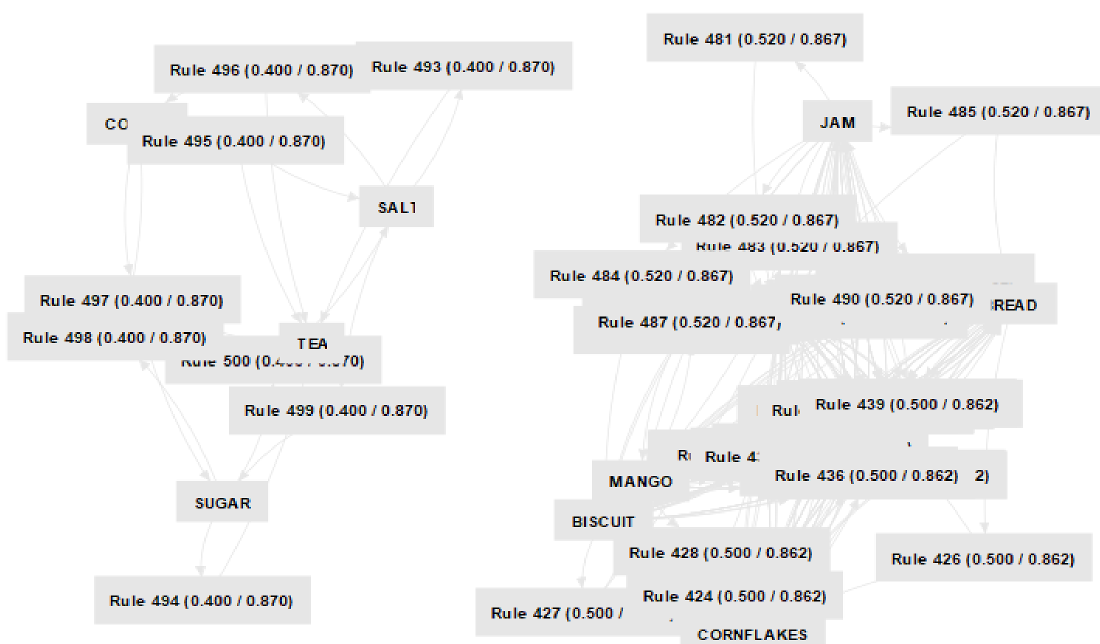
2	0.580	BISCUIT	BREAD
2	0.420	BISCUIT	MILK
2	0.380	BISCUIT	COFFEE
2	0.500	CAKE	CORNFLAKES
2	0.520	CAKE	JAM
2	0.520	CAKE	BREAD
2	0.420	CAKE	MILK
2	0.580	CORNFLAKES	JAM
2	0.560	CORNFLAKES	BREAD
2	0.380	CORNFLAKES	MILK
2	0.380	CORNFLAKES	COFFEE
2	0.580	JAM	BREAD
2	0.400	JAM	MILK
2	0.380	JAM	COFFEE
2	0.400	BREAD	MILK

Step 5: Now these are the conclusions we get
Like if someone buys coffee , they will definitely get a biscuit too.

No.	Premises	Conclusion	Support
35	COFFEE	MANGO	0.380
36	COFFEE	BISCUIT	0.380
37	COFFEE	CORNFLAKES	0.380
38	COFFEE	JAM	0.380
39	COFFEE	BREAD	0.380
40	COFFEE	MANGO, BISCUIT	0.380
41	COFFEE	MANGO, CORNFLAKES	0.380
42	COFFEE	MANGO, JAM	0.380
43	COFFEE	MANGO, BREAD	0.380
44	COFFEE	BISCUIT, CORNFLAKES	0.380
45	COFFEE	BISCUIT, JAM	0.380
46	COFFEE	BISCUIT, BREAD	0.380
47	COFFEE	CORNFLAKES, JAM	0.380

Confidence	LaPlace	Gain
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540
0.826	0.945	-0.540

Step 6: Here we have our graph

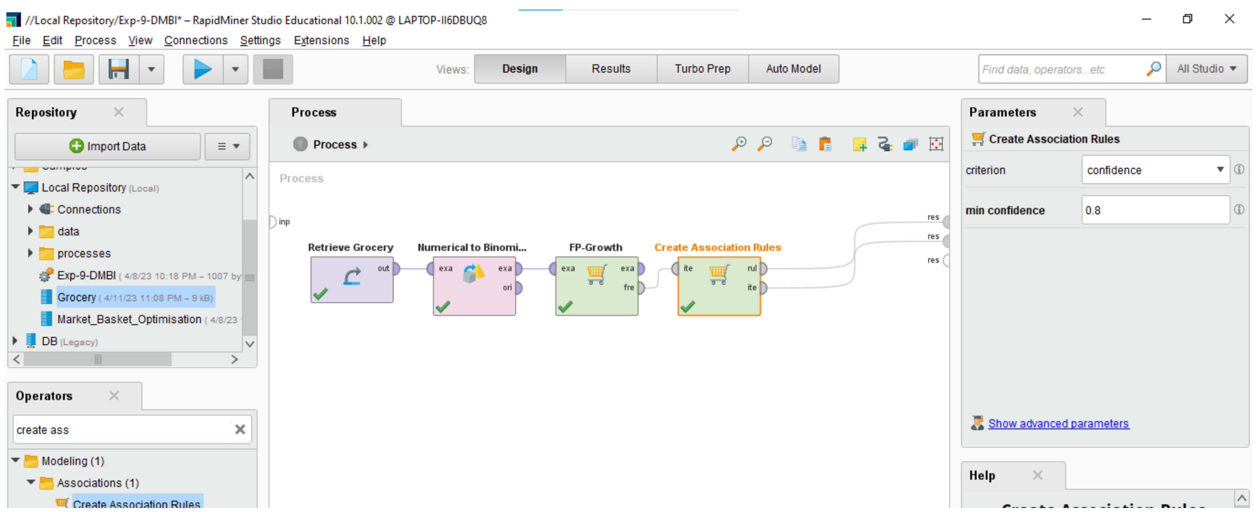


Step 7: Here is the description

AssociationRules

```
Association Rules
[CAKE] --> [CORNFLAKES] (confidence: 0.806)
[CAKE] --> [MANGO, CORNFLAKES] (confidence: 0.806)
[BISCUIT] --> [CAKE, CORNFLAKES] (confidence: 0.806)
[CAKE] --> [BISCUIT, CORNFLAKES] (confidence: 0.806)
[CAKE] --> [CORNFLAKES, JAM] (confidence: 0.806)
[CAKE] --> [CORNFLAKES, BREAD] (confidence: 0.806)
[BISCUIT] --> [MANGO, CAKE, CORNFLAKES] (confidence: 0.806)
[MANGO, BISCUIT] --> [CAKE, CORNFLAKES] (confidence: 0.806)
[CAKE] --> [MANGO, BISCUIT, CORNFLAKES] (confidence: 0.806)
[CAKE] --> [MANGO, CORNFLAKES, JAM] (confidence: 0.806)
```

Final Connection



<u>Implementation using Python</u>

```
[37] for i in range(0, 7501):
         transactions.append([str(dataset.values[i,j]) for j in range(0,20)])
```

```
from apyori import apriori
rules = apriori(transactions = transactions, min_support = 0.003, min_cinfidence = 0.2, min_lift = 3, min_length = 2, max_length = 2)
```

```
[39] results = list(rules)
```

```
[43] def inspect(results):
         supports    =[result[1] for result in results]
         confidences =[result[2][0][2] for result in results]
         lifts        =[result[2][0][3] for result in results]
         return list (zip( supports, confidences, lifts))
     resultsinDataFrame = pd.DataFrame(inspect(results), columns = [ "Support", "Confidence", "Lift"])
```

```
resultsinDataFrame
```

	Support	Confidence	Lift
0	0.003466	0.102767	3.225330
1	0.004533	0.075556	4.843951
2	0.005733	0.072269	3.790833
3	0.005866	0.073950	4.700812
4	0.004266	0.099071	3.259356
5	0.003999	0.179641	3.785070
6	0.003333	0.245098	5.164271
7	0.015998	0.162822	3.291994
8	0.005333	0.054274	3.840659
9	0.003200	0.205128	3.114710
10	0.007999	0.121457	4.122410
11	0.005066	0.322034	4.506672

Model	**Support**	**Confidence**
Rapid Miner	0.38	0.82
Python	0.003	0.1

<u>Conclusion:</u> In this experiment we have performed Implementation of Apriori algorithm in Rapid Miner and Python as well. For the implementation we have used the Grocery dataset where the association between entities has been performed. Thus support and confidence is more for a rapid miner than python thus it's better.