Name: Jigar Siddhpura **SAPID:** 60004210155

DIV: C/C2 **Branch:** Computer Engineering

DMW - EXPERIMENT 6 - ASSOCIATION RULE MINING

Aim: Implementation of Association rule mining Using

1. Apriori Algorithm

2. FPTree

Theory:

Association rule learning is a type of unsupervised learning technique that checks for the dependency of one data item on another data item and maps accordingly so that it can be more profitable. It tries to find some interesting relations or associations among the variables of the dataset. It is based on different rules to discover the interesting relations between variables in the database. Association rule learning is one of the very important concepts of machine learning, and it is employed in Market Basket analysis, Web usage mining, continuous production, etc. Here market basket analysis is a technique used by various big retailers to discover the associations between items. We can understand it by taking an example of a supermarket, as in a supermarket, all products that are purchased together are put together. For example, if a customer buys bread, he most likely can also buy butter, eggs, or milk, so these products are stored on a shelf or mostly nearby.

Association rule learning can be divided into three types of algorithms:

1. Apriori
2. Eclat
3. F-P Growth Algorithm

Association rule learning works on the concept of If and Else Statements, such as if A then B. Here the If the element is called antecedent, then the statement is called as Consequent. These types of relationships where we can find out some association or relation between two items are known as single cardinality. It is all about creating rules, and if the number of items increases, then cardinality also increases accordingly. So, to measure the associations between thousands of data items, there are several metrics. These metrics are given below:

1. Support
2. Confidence
3. Lif

Apriori

1. It is an array based algorithm.
2. It uses Join and Prune technique.
3. Apriori uses a breadth-first search
4. Apriori utilizes a level-wise approach where it generates patterns containing 1 item, then 2 items, then 3 items, and so on.
5. Candidate generation is extremely slow. Runtime increases exponentially depending on the number of different items.
6. Candidate generation is very parallelizable.
7. It requires large memory space due to large number of candidate generation.
8. It scans the database multiple times for generating candidate sets.

FP Growth

1. It is a tree based algorithm.
2. It constructs conditional frequent pattern tree and conditional pattern base from database which satisfy minimum support.
3. FP Growth uses a depth-first search
4. FP Growth utilizes a pattern-growth approach means that, it only considers patterns actually existing in the database.
5. Runtime increases linearly, depending on the number of transactions and items
6. Data are very interdependent, each node needs the root.
7. It requires less memory space due to compact structure and no candidate generation.
8. It scans the database only twice for constructing frequent pattern tree.

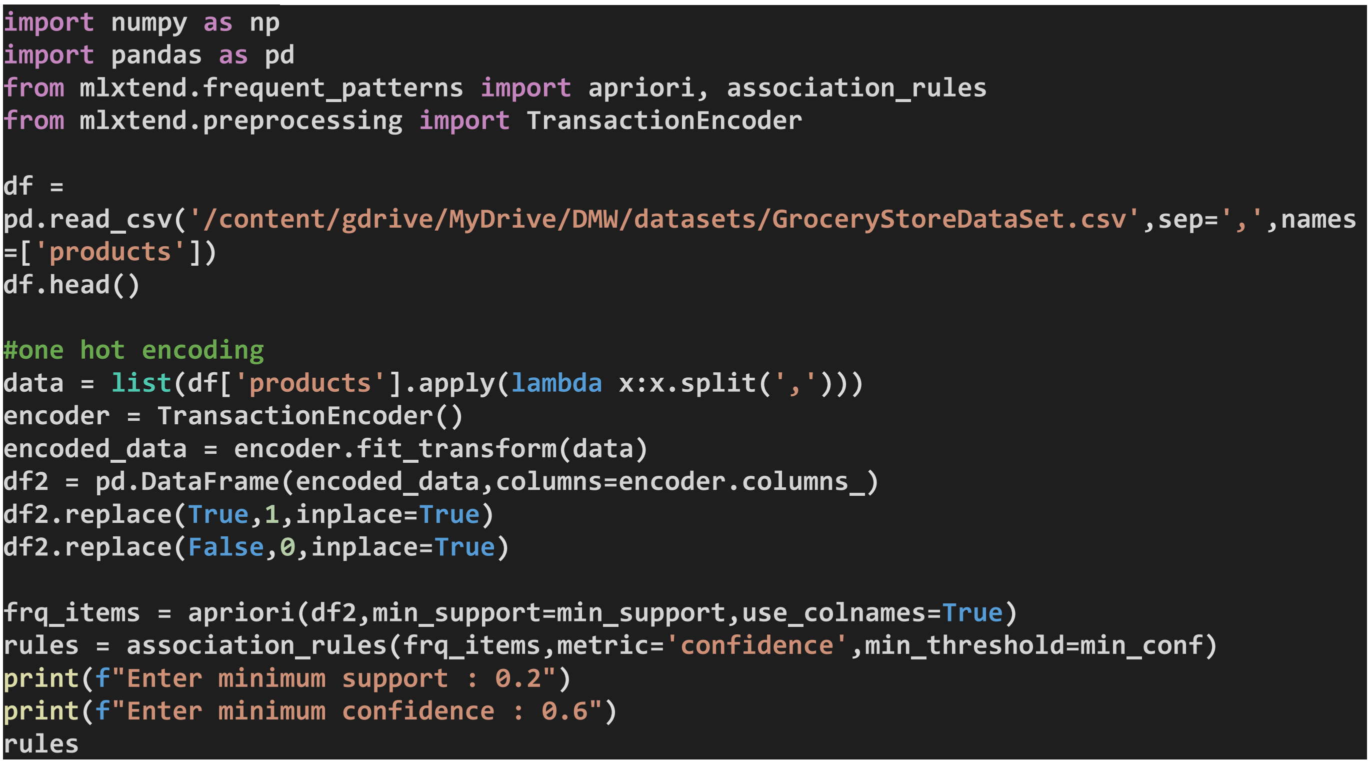
Part A:

Read min_support and confidence from the user

Program Apriori algorithm using inbuilt functions.

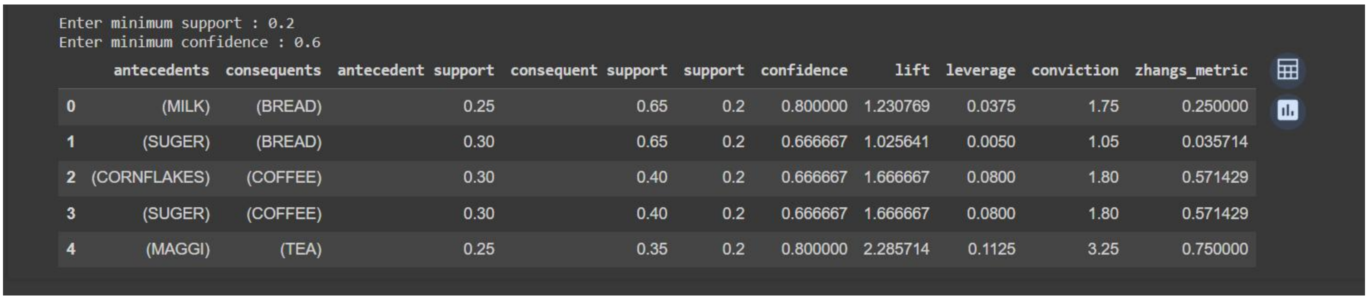
Print the association rules

Code:



```python
import numpy as np
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules
from mlxtend.preprocessing import TransactionEncoder

df =
pd.read_csv('/content/gdrive/MyDrive/DMW/datasets/GroceryStoreDataSet.csv',sep=',',names
=['products'])
df.head()

#one hot encoding
data = list(df['products'].apply(lambda x:x.split(',')))
encoder = TransactionEncoder()
encoded_data = encoder.fit_transform(data)
df2 = pd.DataFrame(encoded_data,columns=encoder.columns_)
df2.replace(True,1,inplace=True)
df2.replace(False,0,inplace=True)

frq_items = apriori(df2,min_support=min_support,use_colnames=True)
rules = association_rules(frq_items,metric='confidence',min_threshold=min_conf)
print(f"Enter minimum support : 0.2")
print(f"Enter minimum confidence : 0.6")
rules
```

Output:

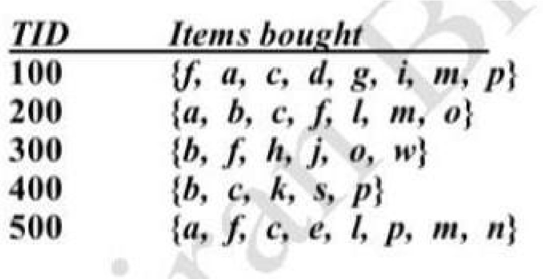


```
Enter minimum support : 0.2
Enter minimum confidence : 0.6
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(MILK)	(BREAD)	0.25	0.65	0.2	0.800000	1.230769	0.0375	1.75	0.250000
1	(SUGER)	(BREAD)	0.30	0.65	0.2	0.666667	1.025641	0.0050	1.05	0.035714
2	(CORNFLAKES)	(COFFEE)	0.30	0.40	0.2	0.666667	1.666667	0.0800	1.80	0.571429
3	(SUGER)	(COFFEE)	0.30	0.40	0.2	0.666667	1.666667	0.0800	1.80	0.571429
4	(MAGGI)	(TEA)	0.25	0.35	0.2	0.800000	2.285714	0.1125	3.25	0.750000

Part B:

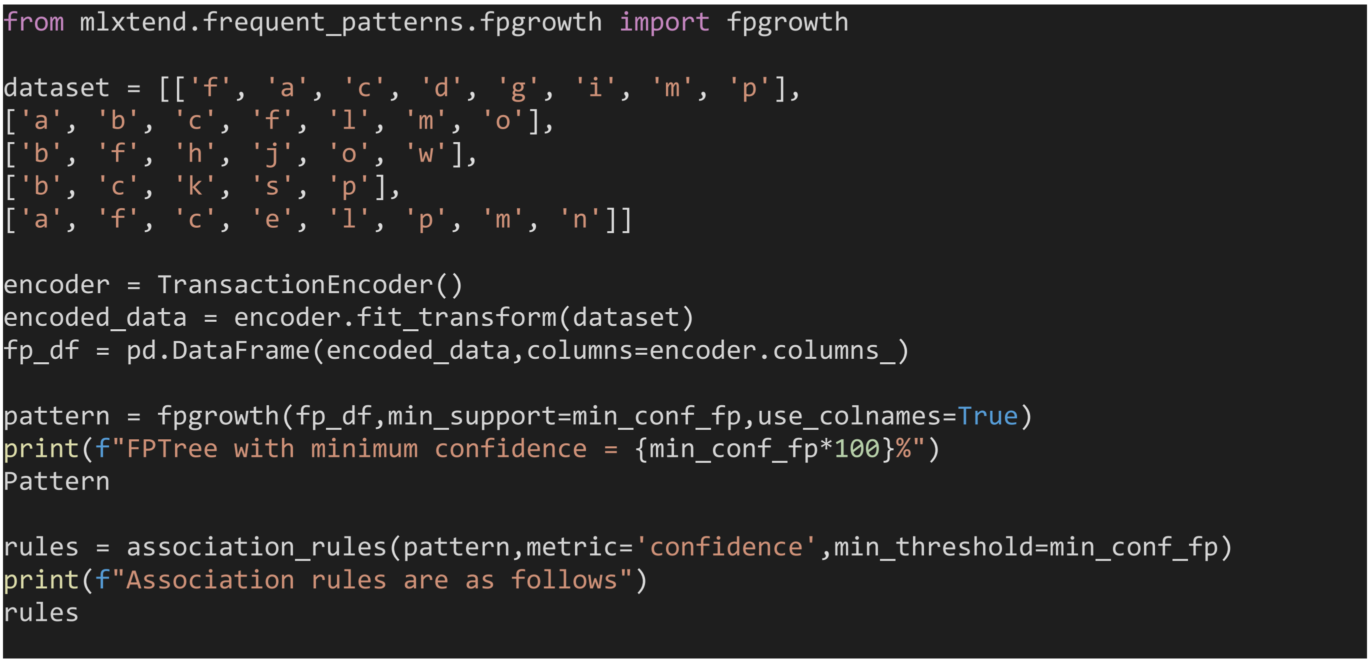
Program FP tree using inbuilt functions for the following dataset



TID	Items bought
100	{f, a, c, d, g, i, m, p}
200	{a, b, c, f, l, m, o}
300	{b, f, h, j, o, w}
400	{b, c, k, s, p}
500	{a, f, c, e, l, p, m, n}

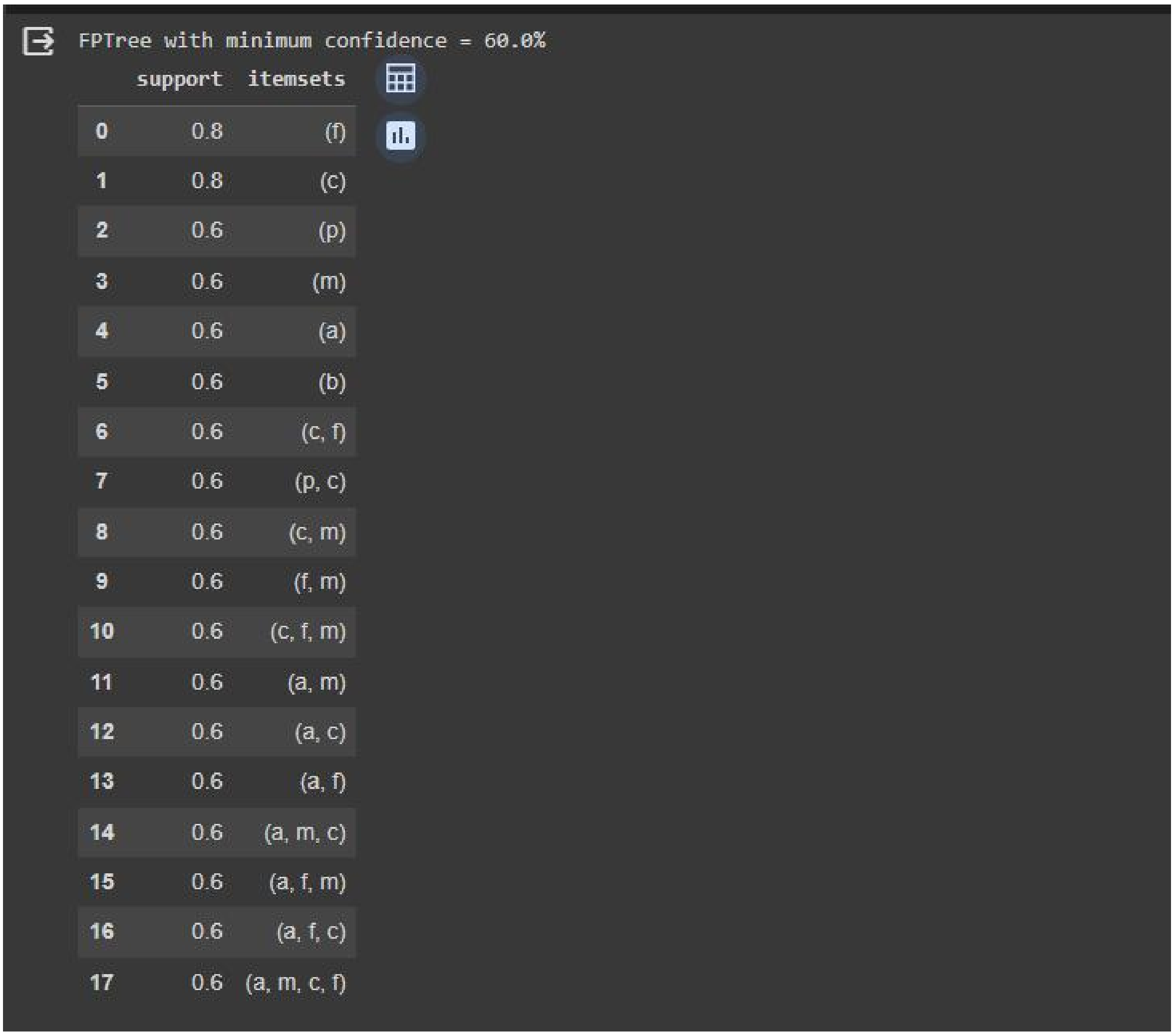
Print the frequent patterns generated.

Code:



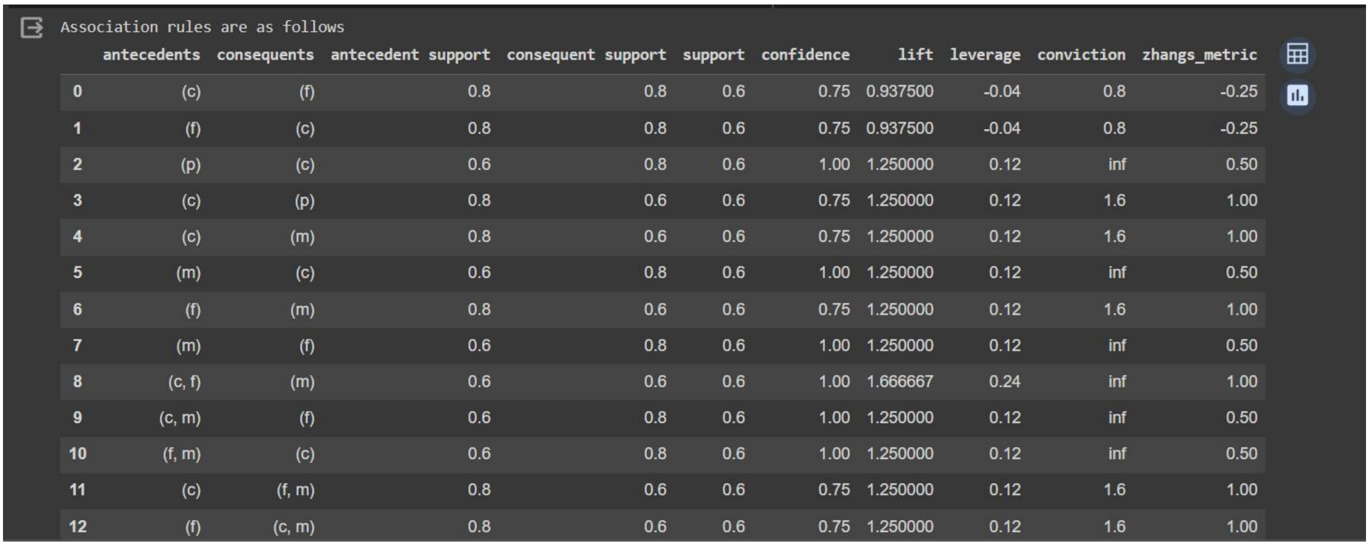
```
from mlxtend.frequent_patterns.fpgrowth import fpgrowth

dataset = [['f', 'a', 'c', 'd', 'g', 'i', 'm', 'p'],
['a', 'b', 'c', 'f', 'l', 'm', 'o'],
['b', 'f', 'h', 'j', 'o', 'w'],
['b', 'c', 'k', 's', 'p'],
['a', 'f', 'c', 'e', 'l', 'p', 'm', 'n']]

encoder = TransactionEncoder()
encoded_data = encoder.fit_transform(dataset)
fp_df = pd.DataFrame(encoded_data,columns=encoder.columns_)

pattern = fpgrowth(fp_df,min_support=min_conf_fp,use_colnames=True)
print(f"FPTree with minimum confidence = {min_conf_fp*100}%")
Pattern

rules = association_rules(pattern,metric='confidence',min_threshold=min_conf_fp)
print(f"Association rules are as follows")
rules
```

Output:



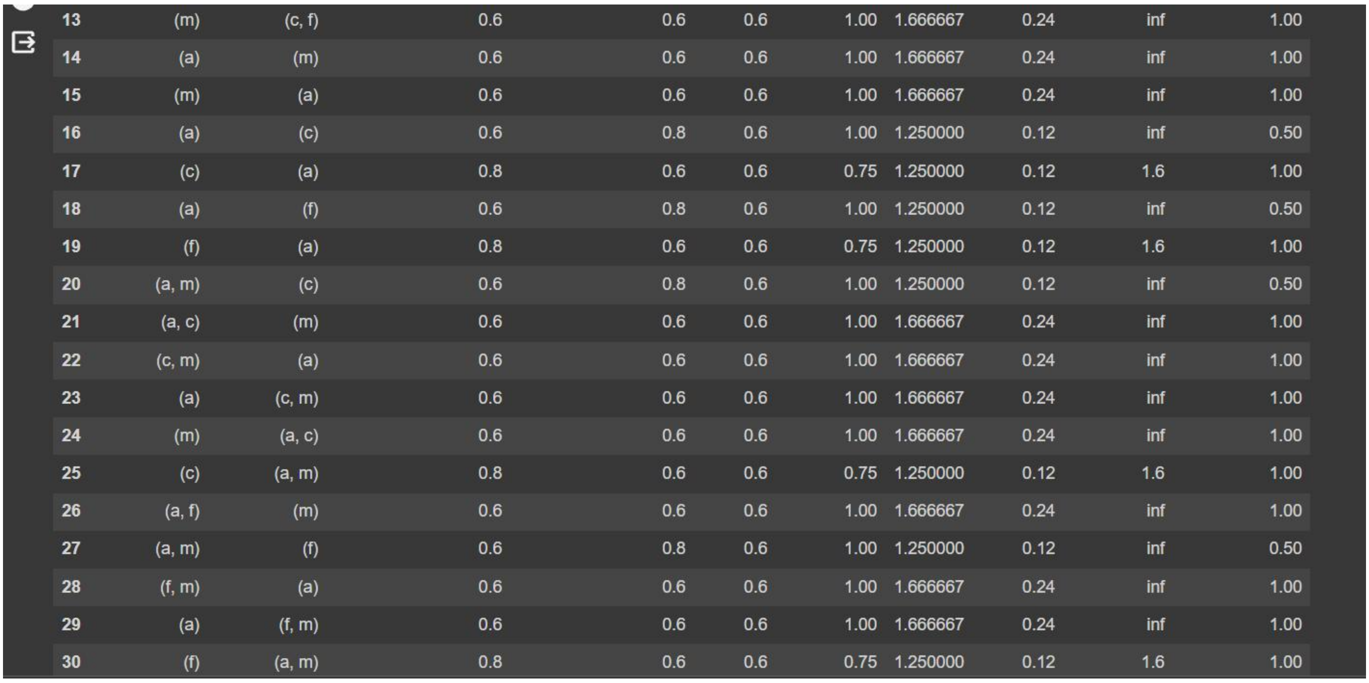
FPTree with minimum confidence = 60.0%

	support	itemsets
0	0.8	(f)
1	0.8	(c)
2	0.6	(p)
3	0.6	(m)
4	0.6	(a)
5	0.6	(b)
6	0.6	(c, f)
7	0.6	(p, c)
8	0.6	(c, m)
9	0.6	(f, m)
10	0.6	(c, f, m)
11	0.6	(a, m)
12	0.6	(a, c)
13	0.6	(a, f)
14	0.6	(a, m, c)
15	0.6	(a, f, m)
16	0.6	(a, f, c)
17	0.6	(a, m, c, f)



Association rules are as follows

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(c)	(f)	0.8	0.8	0.6	0.75	0.937500	-0.04	0.8	-0.25
1	(f)	(c)	0.8	0.8	0.6	0.75	0.937500	-0.04	0.8	-0.25
2	(p)	(c)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
3	(c)	(p)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00
4	(c)	(m)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00
5	(m)	(c)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
6	(f)	(m)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00
7	(m)	(f)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
8	(c, f)	(m)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
9	(c, m)	(f)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
10	(f, m)	(c)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
11	(c)	(f, m)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00
12	(f)	(c, m)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00



13	(m)	(c, f)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
14	(a)	(m)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
15	(m)	(a)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
16	(a)	(c)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
17	(c)	(a)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00
18	(a)	(f)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
19	(f)	(a)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00
20	(a, m)	(c)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
21	(a, c)	(m)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
22	(c, m)	(a)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
23	(a)	(c, m)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
24	(m)	(a, c)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
25	(c)	(a, m)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00
26	(a, f)	(m)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
27	(a, m)	(f)	0.6	0.8	0.6	1.00	1.250000	0.12	inf	0.50
28	(f, m)	(a)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
29	(a)	(f, m)	0.6	0.6	0.6	1.00	1.666667	0.24	inf	1.00
30	(f)	(a, m)	0.8	0.6	0.6	0.75	1.250000	0.12	1.6	1.00

Conclusion:

Implemented Apriori and algorithm for a market basket analysis dataset and made and FP Tree for the given dataset. Apriori is a Join-Based algorithm and FP-Growth is Tree-Based algorithm for frequent itemset mining or frequent pattern mining for market basket analysis.