dataset=[['COFFEE','BISCUIT','MAGGI','CORNFLAKS'],  
 ['BISCUIT','CPRNFLAKS','TEA','BISCUIT','BREAD'],  
 ['BISCUIT','TEA','MAGGI','BREAD'],  
 ['BISCUIT','MAGGE','CORNFLAKS']]  
  
import pandas as pd  
  
from mlxtend.preprocessing import TransactionEncoder  
te = TransactionEncoder()  
te\_ary = te.fit(dataset).transform(dataset)  
df = pd.DataFrame(te\_ary, columns=te.columns\_)  
print(df)

o/p:

BISCUIT BREAD COFFEE CORNFLAKS CPRNFLAKS MAGGE MAGGI TEA

0 True False True True False False True False

1 True True False False True False False True

2 True True False False False False True True

3 True False False True False True False False

from mlxtend.frequent\_patterns import apriori  
print(apriori(df, min\_support=0.5,use\_colnames=True))

o/p:

support itemsets

0 1.0 (BISCUIT)

1 0.5 (BREAD)

2 0.5 (CORNFLAKS)

3 0.5 (MAGGI)

4 0.5 (TEA)

5 0.5 (BREAD, BISCUIT)

6 0.5 (CORNFLAKS, BISCUIT)

7 0.5 (MAGGI, BISCUIT)

8 0.5 (TEA, BISCUIT)

9 0.5 (BREAD, TEA)

10 0.5 (BREAD, TEA, BISCUIT)

frequent\_itemsets = apriori(df, min\_support=0.5, use\_colnames=True)  
frequent\_itemsets['length'] = frequent\_itemsets['itemsets'].apply(lambda x: len(x))  
print(frequent\_itemsets)

o/p:

support itemsets length

0 1.0 (BISCUIT) 1

1 0.5 (BREAD) 1

2 0.5 (CORNFLAKS) 1

3 0.5 (MAGGI) 1

4 0.5 (TEA) 1

5 0.5 (BREAD, BISCUIT) 2

6 0.5 (CORNFLAKS, BISCUIT) 2

7 0.5 (MAGGI, BISCUIT) 2

8 0.5 (TEA, BISCUIT) 2

9 0.5 (BREAD, TEA) 2

10 0.5 (BREAD, TEA, BISCUIT) 3

print(frequent\_itemsets[ (frequent\_itemsets['length'] == 3) &  
 (frequent\_itemsets['support'] >= 0.5) ])  
print(frequent\_itemsets[ (frequent\_itemsets['length'] == 2) &  
(frequent\_itemsets['support'] >= 0.5) ])

support itemsets length

10 0.5 (BREAD, TEA, BISCUIT) 3

support itemsets length

5 0.5 (BREAD, BISCUIT) 2

6 0.5 (CORNFLAKS, BISCUIT) 2

7 0.5 (MAGGI, BISCUIT) 2

8 0.5 (TEA, BISCUIT) 2

9 0.5 (BREAD, TEA) 2

from mlxtend.frequent\_patterns import association\_rules  
rules = association\_rules(frequent\_itemsets, metric="confidence", min\_threshold=0.7)  
print(rules)

o/p:

antecedents consequents ... leverage conviction

0 (BREAD) (BISCUIT) ... 0.00 inf

1 (CORNFLAKS) (BISCUIT) ... 0.00 inf

2 (MAGGI) (BISCUIT) ... 0.00 inf

3 (TEA) (BISCUIT) ... 0.00 inf

4 (BREAD) (TEA) ... 0.25 inf

5 (TEA) (BREAD) ... 0.25 inf

6 (BREAD, TEA) (BISCUIT) ... 0.00 inf

7 (BREAD, BISCUIT) (TEA) ... 0.25 inf

8 (TEA, BISCUIT) (BREAD) ... 0.25 inf

9 (BREAD) (TEA, BISCUIT) ... 0.25 inf

10 (TEA) (BREAD, BISCUIT) ... 0.25 inf

[11 rows x 9 columns]

rules['antecedant\_len'] = rules['antecedents'].apply(lambda x: len(x))  
print(rules)

o/p:

antecedents consequents ... conviction antecedant\_len

0 (BREAD) (BISCUIT) ... inf 1

1 (CORNFLAKS) (BISCUIT) ... inf 1

2 (MAGGI) (BISCUIT) ... inf 1

3 (TEA) (BISCUIT) ... inf 1

4 (BREAD) (TEA) ... inf 1

5 (TEA) (BREAD) ... inf 1

6 (BREAD, TEA) (BISCUIT) ... inf 2

7 (BREAD, BISCUIT) (TEA) ... inf 2

8 (TEA, BISCUIT) (BREAD) ... inf 2

9 (BREAD) (TEA, BISCUIT) ... inf 1

10 (TEA) (BREAD, BISCUIT) ... inf 1

[11 rows x 10 columns]

rules=rules[ (rules['antecedant\_len'] >= 2) &  
(rules['confidence'] > 0.75) &  
(rules['lift'] >1.75) ]  
print(rules)

o/p:

antecedents consequents ... conviction antecedant\_len

7 (BREAD, BISCUIT) (TEA) ... inf 2

8 (TEA, BISCUIT) (BREAD) ... inf 2

[2 rows x 10 columns]