```
pip install apyori
```

```
Collecting apyori
  Downloading apyori-1.1.2.tar.gz (8.6 kB)
Building wheels for collected packages: apyori
  Building wheel for apyori (setup.py) ... done
  Created wheel for apyori: filename=apyori-1.1.2-py3-none-any.whl size=5974 sha256=1d776f67605f1f7878b82f6
  Stored in directory: /root/.cache/pip/wheels/cb/f6/e1/57973c631d27efd1a2f375bd6a83b2a616c4021f24aab84080
Successfully built apyori
Installing collected packages: apyori
Successfully installed apyori-1.1.2
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from apyori import apriori
```

```
store_data = pd.read_csv("/content/Market Basket_Small dataset.csv", header=None)
display(store_data.head(15))
print(store_data.shape)
```

	0	1	2	3	4	5
0	Wine	Chips	Bread	Butter	Milk	Apple
1	Wine	Chips	Bread	Butter	Milk	Apple
2	Wine	Chips	Bread	Butter	Milk	NaN
3	Wine	Chips	NaN	Butter	Milk	NaN
4	Wine	NaN	Bread	NaN	NaN	Apple
5	NaN	NaN	NaN	Butter	Milk	NaN
6	NaN	Chips	Bread	NaN	NaN	Apple
7	Wine	Chips	NaN	Butter	Milk	NaN
8	Wine	NaN	Bread	NaN	NaN	Apple
9	Wine	NaN	Bread	NaN	Milk	NaN
10	NaN	Chips	Bread	Butter	NaN	Apple
11	Wine	NaN	NaN	Butter	Milk	Apple
12	Wine	Chips	Bread	Butter	Milk	NaN
13	Wine	NaN	Bread	NaN	Milk	Apple
14	Wine	NaN	Bread	Butter	Milk	Apple

```
(22, 6)
```

```
transactions = []
for i in range(0, len(store_data)):
    transactions.append([str(store_data.values[i,j]) for j in range(0, len(store_data.columns))])
```

```
association_rules = apriori(transactions, min_support=0.5, min_confidence=0.7, min_lift=1.2, min_length=2)
association_results = list(association_rules)
```

```
print(len(association_results))
```

```
print(len(association_results ))
```

```
3
```

```
print(association_results )
```

```
[RelationRecord(items=frozenset({'Butter', 'Milk '}), support=0.6363636363636364, ordered_statistics=[Order
```

```
print("There are {} Relation derived.".format(len(association_results)))
```

```
There are 3 Relation derived.
```

```
for i in range(0, len(association_results)):
    print(association_results[i][0])
```

```
frozenset({'Butter', 'Milk '})
frozenset({'Bread', 'Milk ', 'Wine '})
frozenset({'Butter', 'Milk ', 'Wine '})
```

```
# Import the transaction encoder function from mlxtend
from mlxtend.preprocessing import TransactionEncoder

# Instantiate transaction encoder and identify unique items
encoder = TransactionEncoder().fit(transactions)

# One-hot encode transactions
onehot = encoder.transform(transactions)

# Convert one-hot encoded data to DataFrame
onehot = pd.DataFrame(onehot, columns = encoder.columns_).drop('nan', axis=1)

# Print the one
onehot.head()
```

	Apple	Bread	Butter	Chips	Milk	Wine
0	True	True	True	True	True	True
1	True	True	True	True	True	True
2	False	True	True	True	True	True
3	False	False	True	True	True	True
4	True	True	False	False	False	True

```
# Import the association rules function
from mlxtend.frequent_patterns import apriori, association_rules

# Compute frequent itemsets using the Apriori algorithm
frequent_itemsets = apriori(onehot, min_support = 0.5,
                            max_len = 2, use_colnames = True)

# Compute all association rules using confidence
rules = association_rules(frequent_itemsets,
                            metric = "confidence",
                            min_threshold = 0.7)

# Print association rules
rules.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16 entries, 0 to 15
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   antecedents         16 non-null     object 
 1   consequents         16 non-null     object 
 2   antecedent support  16 non-null     float64
 3   consequent support  16 non-null     float64
 4   support             16 non-null     float64
 5   confidence          16 non-null     float64
 6   lift                16 non-null     float64
 7   leverage            16 non-null     float64
 8   conviction          16 non-null     float64
dtypes: float64(7), object(2)
memory usage: 1.2+ KB
```

```
rules.head()
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Apple)	(Bread)	0.681818	0.727273	0.590909	0.866667	1.191667	0.095041	2.045455
1	(Bread)	(Apple)	0.727273	0.681818	0.590909	0.812500	1.191667	0.095041	1.696970
2	(Apple)	(Milk)	0.681818	0.772727	0.500000	0.733333	0.949020	-0.026860	0.852273
3	(Apple)	(Wine)	0.681818	0.727273	0.500000	0.733333	1.008333	0.004132	1.022727
4	(Bread)	(Milk)	0.727273	0.772727	0.545455	0.750000	0.970588	-0.016529	0.909091