In [ ]:
```
pip install mlxtend
```

```
Requirement already satisfied: mlxtend in c:\users\ac_001 lenovo.id\appdata\local
\programs\python\python311\lib\site-packages (0.23.1)
Requirement already satisfied: scipy>=1.2.1 in c:\users\ac_001 lenovo.id\appdata
\local\programs\python\python311\lib\site-packages (from mlxtend) (1.11.2)
Requirement already satisfied: numpy>=1.16.2 in c:\users\ac_001 lenovo.id\appdata
\local\programs\python\python311\lib\site-packages (from mlxtend) (1.24.3)
Requirement already satisfied: pandas>=0.24.2 in c:\users\ac_001 lenovo.id\appdat
a\local\programs\python\python311\lib\site-packages (from mlxtend) (2.1.0)
Requirement already satisfied: scikit-learn>=1.0.2 in c:\users\ac_001 lenovo.id\a
ppdata\local\programs\python\python311\lib\site-packages (from mlxtend) (1.4.1.po
st1)
Requirement already satisfied: matplotlib>=3.0.0 in c:\users\ac_001 lenovo.id\app
data\local\programs\python\python311\lib\site-packages (from mlxtend) (3.7.2)
Requirement already satisfied: joblib>=0.13.2 in c:\users\ac_001 lenovo.id\appdat
a\local\programs\python\python311\lib\site-packages (from mlxtend) (1.3.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\ac_001 lenovo.id\appd
ata\local\programs\python\python311\lib\site-packages (from matplotlib>=3.0.0->ml
xtend) (1.1.0)
Requirement already satisfied: cycler>=0.10 in c:\users\ac_001 lenovo.id\appdata
\local\programs\python\python311\lib\site-packages (from matplotlib>=3.0.0->mlxte
nd) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\ac_001 lenovo.id\app
data\local\programs\python\python311\lib\site-packages (from matplotlib>=3.0.0->m
lxtend) (4.42.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\ac_001 lenovo.id\app
data\local\programs\python\python311\lib\site-packages (from matplotlib>=3.0.0->m
lxtend) (1.4.5)
Requirement already satisfied: packaging>=20.0 in c:\users\ac_001 lenovo.id\appda
ta\local\programs\python\python311\lib\site-packages (from matplotlib>=3.0.0->mlx
tend) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\ac_001 lenovo.id\appdata
\local\programs\python\python311\lib\site-packages (from matplotlib>=3.0.0->mlxte
nd) (10.0.0)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\users\ac_001 lenovo.id
\appdata\local\programs\python\python311\lib\site-packages (from matplotlib>=3.0.
0->mlxtend) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\ac_001 lenovo.id
\appdata\local\programs\python\python311\lib\site-packages (from matplotlib>=3.0.
0->mlxtend) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\ac_001 lenovo.id\appdata
\local\programs\python\python311\lib\site-packages (from pandas>=0.24.2->mlxtend)
(2023.3)
Requirement already satisfied: tzdata>=2022.1 in c:\users\ac_001 lenovo.id\appdat
a\local\programs\python\python311\lib\site-packages (from pandas>=0.24.2->mlxten
d) (2023.3)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\ac_001 lenovo.id
\appdata\local\programs\python\python311\lib\site-packages (from scikit-learn>=1.
0.2->mlxtend) (3.4.0)
Requirement already satisfied: six>=1.5 in c:\users\ac_001 lenovo.id\appdata\loca
l\programs\python\python311\lib\site-packages (from python-dateutil>=2.7->matplot
lib>=3.0.0->mlxtend) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
[notice] A new release of pip is available: 23.1.2 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [ ]:
```python
# Mengimpor library yang diperlukan
from mlxtend.frequent_patterns import apriori, association_rules
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import fpgrowth
import pandas as pd
```

In [ ]:
```python
dataset = [
    ['MILK', 'BREAD', 'BISCUIT'],
    ['BREAD', 'MILK', 'BISCUIT', 'CORNFLAKES'],
    ['BREAD', 'TEA', 'BOURNVITA'],
    ['JAM', 'MAGGI', 'BREAD', 'MILK'],
    ['MAGGI', 'TEA', 'BISCUIT'],
    ['BREAD', 'TEA', 'BOURNVITA'],
    ['MAGGI', 'TEA', 'CORNFLAKES'],
    ['MAGGI', 'BREAD', 'TEA', 'BISCUIT'],
    ['JAM', 'MAGGI', 'BREAD', 'TEA'],
    ['BREAD', 'MILK'],
    ['COFFEE', 'COCK', 'BISCUIT', 'CORNFLAKES'],
    ['COFFEE', 'COCK', 'BISCUIT', 'CORNFLAKES'],
    ['COFFEE', 'SUGER', 'BOURNVITA'],
    ['BREAD', 'COFFEE', 'COCK'],
    ['BREAD', 'SUGER', 'BISCUIT'],
    ['COFFEE', 'SUGER', 'CORNFLAKES'],
    ['BREAD', 'SUGER', 'BOURNVITA'],
    ['BREAD', 'COFFEE', 'SUGER'],
    ['BREAD', 'COFFEE', 'SUGER'],
    ['TEA', 'MILK', 'COFFEE', 'CORNFLAKES']
]
```

In [ ]:
```python
# Menggunakan TransactionEncoder
te = TransactionEncoder()
te_ary = te.fit(dataset).transform(dataset)
df = pd.DataFrame(te_ary, columns=te.columns_)

# Menampilkan hasil konversi
print(te_ary)
```

```
[[ True False  True False False False False False  True False False]
 [ True False  True False False  True False False  True False False]
 [False  True  True False False False False False False False  True]
 [False False  True False False False  True  True  True False False]
 [ True False False False False False False  True False False  True]
 [False  True  True False False False False False False False  True]
 [False False False False False  True False  True False False  True]
 [ True False  True False False False False  True False False  True]
 [False False  True False False False  True  True False False  True]
 [False False  True False False False False False  True False False]
 [ True False False  True  True  True False False False False False]
 [ True False False  True  True  True False False False False False]
 [False  True False False  True False False False False  True False]
 [False False  True  True  True False False False False False False]
 [ True False  True False False False False False False  True False]
 [False False False False  True  True False False False  True False]
 [False  True  True False False False False False False  True False]
 [False False  True False  True False False False False  True False]
 [False False  True False  True False False False False  True False]
 [False False False False  True  True False False  True False  True]]
```

In [ ]:
```python
# 1. min_support = 70%, min_confidence = 70%
min_support = 0.7
min_confidence = 0.7

# Apriori
print("Apriori")

# Aplikasikan algoritma aprior
frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True)
print(frequent_itemsets)

# Tampilkan aturan asosiasi dengan min_confidence yang diberikan
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=
print(rules[['antecedents', 'consequents', 'support', 'confidence']])
```

```
Apriori
Empty DataFrame
Columns: [support, itemsets]
Index: []
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[32], line 13
     10 print(frequent_itemsets)
     12 # Tampilkan aturan asosiasi dengan min_confidence yang diberikan
---> 13 rules = association_rules(frequent_itemsets, metric="confidence", min_thr
eshold=min_confidence)
     14 print(rules[['antecedents', 'consequents', 'support', 'confidence']])

File c:\Users\AC_001 LENOVO.iD\AppData\Local\Programs\Python\Python311\Lib\site-p
ackages\mlxtend\frequent_patterns\association_rules.py:83, in association_rules(d
f, metric, min_threshold, support_only)
     18 """Generates a DataFrame of association rules including the
     19 metrics 'score', 'confidence', and 'lift'
     20
   (...)
     80
     81 """
     82 if not df.shape[0]:
---> 83     raise ValueError(
     84         "The input DataFrame `df` containing " "the frequent itemsets is
empty."
     85     )
     87 # check for mandatory columns
     88 if not all(col in df.columns for col in ["support", "itemsets"]):

ValueError: The input DataFrame `df` containing the frequent itemsets is empty.
```

In [ ]:
```python
# 2. min_support = 20%, min_confidence = 40%
min_support = 0.2
min_confidence = 0.4

# Apriori
print("Apriori")

# Aplikasikan algoritma aprior
frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True)
print(frequent_itemsets)

# Tampilkan aturan asosiasi dengan min_confidence yang diberikan
```

```
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=
print(rules[['antecedents', 'consequents', 'support', 'confidence']])
```

```
Apriori
    support              itemsets
0      0.35             (BISCUIT)
1      0.20           (BOURNVITA)
2      0.65               (BREAD)
3      0.40              (COFFEE)
4      0.30          (CORNFLAKES)
5      0.25               (MAGGI)
6      0.25                (MILK)
7      0.30               (SUGER)
8      0.35                 (TEA)
9      0.20      (BISCUIT, BREAD)
10     0.20         (BREAD, MILK)
11     0.20        (BREAD, SUGER)
12     0.20          (TEA, BREAD)
13     0.20  (CORNFLAKES, COFFEE)
14     0.20       (COFFEE, SUGER)
15     0.20          (MAGGI, TEA)
    antecedents   consequents  support  confidence
0     (BISCUIT)       (BREAD)      0.2    0.571429
1        (MILK)       (BREAD)      0.2    0.800000
2       (SUGER)       (BREAD)      0.2    0.666667
3         (TEA)       (BREAD)      0.2    0.571429
4  (CORNFLAKES)      (COFFEE)      0.2    0.666667
5      (COFFEE)  (CORNFLAKES)      0.2    0.500000
6      (COFFEE)       (SUGER)      0.2    0.500000
7       (SUGER)      (COFFEE)      0.2    0.666667
8       (MAGGI)         (TEA)      0.2    0.800000
9         (TEA)       (MAGGI)      0.2    0.571429
```

In [ ]:
```
# 3. min_support = 30%, min_confidence = 70%
min_support = 0.3
min_confidence = 0.7

# Apriori
print("Apriori")

# Aplikasikan algoritma aprior
frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True)
print(frequent_itemsets)

# Tampilkan aturan asosiasi dengan min_confidence yang diberikan
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=
print(rules[['antecedents', 'consequents', 'support', 'confidence']])
```

```
Apriori
   support      itemsets
0     0.35     (BISCUIT)
1     0.65       (BREAD)
2     0.40      (COFFEE)
3     0.30  (CORNFLAKES)
4     0.30       (SUGER)
5     0.35         (TEA)
Empty DataFrame
Columns: [antecedents, consequents, support, confidence]
Index: []
```

In [ ]:
```
# 4. min_support = 30%, min_confidence = 60%
min_support = 0.3
min_confidence = 0.6

# Apriori
print("Apriori")

# Aplikasikan algoritma aprior
frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True)
print(frequent_itemsets)

# Tampilkan aturan asosiasi dengan min_confidence yang diberikan
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=
print(rules[['antecedents', 'consequents', 'support', 'confidence']])
```

```
Apriori
   support      itemsets
0     0.35     (BISCUIT)
1     0.65       (BREAD)
2     0.40      (COFFEE)
3     0.30  (CORNFLAKES)
4     0.30       (SUGER)
5     0.35         (TEA)
Empty DataFrame
Columns: [antecedents, consequents, support, confidence]
Index: []
```

In [ ]:
```
# 5. min_support = 50%, min_confidence = 50%
min_support = 0.5
min_confidence = 0.5

# Apriori
print("Apriori")

# Aplikasikan algoritma aprior
frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True)
print(frequent_itemsets)

# Tampilkan aturan asosiasi dengan min_confidence yang diberikan
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=
print(rules[['antecedents', 'consequents', 'support', 'confidence']])
```

```
Apriori
   support itemsets
0     0.65  (BREAD)
Empty DataFrame
Columns: [antecedents, consequents, support, confidence]
Index: []
```

In [ ]:
```
# 6. min_support = 30%, min_confidence = 50%
min_support = 0.3
min_confidence = 0.5

# Apriori
print("Apriori")

# Aplikasikan algoritma aprior
frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True)
print(frequent_itemsets)
```

```python
# Tampilkan aturan asosiasi dengan min_confidence yang diberikan
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=
print(rules[['antecedents', 'consequents', 'support', 'confidence']])
```

```
Apriori
   support      itemsets
0     0.35     (BISCUIT)
1     0.65       (BREAD)
2     0.40      (COFFEE)
3     0.30  (CORNFLAKES)
4     0.30       (SUGER)
5     0.35         (TEA)
Empty DataFrame
Columns: [antecedents, consequents, support, confidence]
Index: []
```