```
import pandas as pd
from mlxtend.frequent_patterns import fpgrowth, association_rules
from mlxtend.preprocessing import TransactionEncoder

# Given transaction set
transactions = [
    [1, 3, 4],
    [2, 3, 5],
    [1, 2, 3, 5],
    [2, 5]
]

# Step 1: One-hot encode the transaction data
te = TransactionEncoder()
te_ary = te.fit(transactions).transform(transactions)
df = pd.DataFrame(te_ary, columns=te.columns_)

# Step 2: Apply FP-Growth algorithm
frequent_itemsets = fpgrowth(df, min_support=0.5, use_colnames=True)

# Step 3: Generate Association Rules
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.75)

# Step 4: Print results
print("📌 Frequent Itemsets:")
print(frequent_itemsets)

print("\n📌 Association Rules:")
print(rules[['antecedents', 'consequents', 'support', 'confidence', 'lift']])
```

```
📌 Frequent Itemsets:
   support   itemsets
0     0.75        (3)
1     0.50        (1)
2     0.75        (5)
3     0.75        (2)
4     0.50     (3, 5)
5     0.50     (1, 3)
6     0.75     (2, 5)
7     0.50     (2, 3)
8     0.50  (2, 3, 5)

📌 Association Rules:
  antecedents consequents  support  confidence      lift
0         (1)         (3)     0.50         1.0  1.333333
1         (2)         (5)     0.75         1.0  1.333333
2         (5)         (2)     0.75         1.0  1.333333
3      (2, 3)         (5)     0.50         1.0  1.333333
4      (3, 5)         (2)     0.50         1.0  1.333333
```

```
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules


# Define transactions as a list of lists
transactions = [
    [1, 3, 4],
    [2, 3, 5],
    [1, 2, 3, 5],
    [2, 5]
]


# Convert to a one-hot encoded DataFrame
df = pd.DataFrame([[item in transaction for item in range(1, 6)] for transaction in transactions],
                  columns=[1, 2, 3, 4, 5])


# Step 1: Apply Apriori algorithm to find frequent itemsets
frequent_itemsets = apriori(df, min_support=2/4, use_colnames=True)  # Minimum support = 2 (out of 4 transactions)


# Step 2: Generate association rules with minimum confidence of 75%
```

```
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.75)


# Step 3: Drop unwanted columns (lift, leverage, conviction)
rules = rules.drop(columns=['lift', 'leverage', 'conviction', 'antecedent support',  'consequent support',  'support'])


# Step 3: Display the results
print("Frequent Itemsets:")
print(frequent_itemsets)


print("\nAssociation Rules:")
print(rules)
```

```
Frequent Itemsets:
   support   itemsets
0     0.50        (1)
1     0.75        (2)
2     0.75        (3)
3     0.75        (5)
4     0.50     (1, 3)
5     0.50     (2, 3)
6     0.75     (2, 5)
7     0.50     (3, 5)
8     0.50  (2, 3, 5)

Association Rules:
  antecedents consequents  confidence  representativity  zhangs_metric  \
0         (1)         (3)         1.0               1.0            0.5
1         (2)         (5)         1.0               1.0            1.0
2         (5)         (2)         1.0               1.0            1.0
3      (2, 3)         (5)         1.0               1.0            0.5
4      (3, 5)         (2)         1.0               1.0            0.5

    jaccard  certainty  kulczynski
0  0.666667        1.0    0.833333
1  1.000000        1.0    1.000000
2  1.000000        1.0    1.000000
3  0.666667        1.0    0.833333
4  0.666667        1.0    0.833333
```