Student Name	**Swanand Garge**
PRN No	2280030433
Roll No	39
Program	Computer Engg.
Year	Third Year
Division	D (D2)
Subject	Data Warehouse and Data Mining
Assignment No	7

Association Mining Techniques

Objective: The purpose of this lab assignment is to apply association mining techniques to uncover hidden patterns and relationships within a dataset. This hands-on exercise will help you understand the implementation and analysis of association rules.

CODE :-

```
import pandas as pd
from mlxtend.frequent_patterns import apriori
from mlxtend.preprocessing import TransactionEncoder
import warnings
from itertools import combinations

# Suppress warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.filterwarnings("ignore", category=FutureWarning)

# Step 1: Load and clean the dataset
file_path = 'D://PROGRAMMING//PYTHON//apriori//retail_dataset.csv'
try:
    df = pd.read_csv(file_path, encoding='utf-8-sig')
except FileNotFoundError:
    print(f"File not found: {file_path}")
    exit(1)

# Function to clean and convert to numeric
def clean_numeric(x):
    try:
        return pd.to_numeric(x)
    except:
        return pd.np.nan

# Clean numeric columns
numeric_columns = ['Quantity', 'UnitPrice']
for col in numeric_columns:
    if col in df.columns:
        df[col] = df[col].apply(clean_numeric)

# Remove rows with NaN values
df = df.dropna()

# Ensure 'InvoiceNo' is string type
if 'InvoiceNo' in df.columns:
    df['InvoiceNo'] = df['InvoiceNo'].astype(str)

# Print DataFrame info
print("DataFrame Info after cleaning:")
print(df.info())

print("\nColumn Names:")
```

```
print(df.columns)

print("\nFirst few rows after cleaning:")
print(df.head())

# Ensure correct columns exist
invoice_col = 'InvoiceNo' if 'InvoiceNo' in df.columns else df.columns[0]
description_col = 'Description' if 'Description' in df.columns else df.columns[2]

print(f"\nUsing '{invoice_col}' as the invoice number column")
print(f"Using '{description_col}' as the product description column")

# Step 2: Preprocess the data
transactions = df.groupby(invoice_col)[description_col].apply(list).reset_index()

# Convert transactions to one-hot encoded DataFrame
te = TransactionEncoder()
te_ary =
te.fit(transactions[description_col]).transform(transactions[description_col])
df_encoded = pd.DataFrame(te_ary, columns=te.columns_)

# Step 3: Apply Apriori algorithm
min_support = 0.01
frequent_itemsets = apriori(df_encoded, min_support=min_support,
use_colnames=True)

# Step 4: Generate association rules manually
def generate_rules(frequent_itemsets, min_confidence=0.5):
    rules = []
    for _, row in frequent_itemsets.iterrows():
        items = list(row['itemsets'])
        support = row['support']

        if len(items) < 2:  # Skip itemsets with less than 2 items
            continue

        # Generate all possible combinations for antecedents
        for i in range(1, len(items)):
            for antecedent in combinations(items, i):
                antecedent = frozenset(antecedent)
                consequent = frozenset(items) - antecedent

                # Calculate antecedent support
                antecedent_support =
df_encoded[list(antecedent)].all(axis=1).mean()

                # Calculate confidence
                confidence = support / antecedent_support if antecedent_support >
0 else 0

                # Calculate consequent support
                consequent_support =
df_encoded[list(consequent)].all(axis=1).mean()
```

```python
                # Calculate lift
                lift = confidence / consequent_support if consequent_support > 0
else 0

                if confidence >= min_confidence:
                    rules.append({
                        'antecedents': set(antecedent),
                        'consequents': set(consequent),
                        'support': support,
                        'confidence': confidence,
                        'lift': lift
                    })

    return pd.DataFrame(rules)

# Generate rules
min_confidence = 0.5
rules = generate_rules(frequent_itemsets, min_confidence)

# Display results
print("\nTop 20 Frequent Itemsets (by support):")
print(frequent_itemsets.sort_values(by='support', ascending=False).head(20))

print("\nTop 20 Association Rules (by confidence):")
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
if not rules.empty:
    print(rules[['antecedents', 'consequents', 'support', 'confidence', 'lift']]
          .sort_values(by='confidence', ascending=False)
          .head(20)
          .to_string(index=False))
else:
    print("No rules found with the current support and confidence thresholds")

print("\nAnalysis complete.")
print(f"Minimum support used: {min_support}")
print(f"Minimum confidence used: {min_confidence}")

# Optionally, save results to CSV files
frequent_itemsets.to_csv('frequent_itemsets.csv', index=False)
if not rules.empty:
    rules.to_csv('association_rules.csv', index=False)
```

OUTPUT :-

```
DataFrame Info after cleaning:
<class 'pandas.core.frame.DataFrame'>
Index: 406829 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    406829 non-null  object
 1   StockCode    406829 non-null  object
 2   Description  406829 non-null  object
 3   Quantity     406829 non-null  int64
 4   InvoiceDate  406829 non-null  object
 5   UnitPrice    406829 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      406829 non-null  object
dtypes: float64(2), int64(1), object(5)
memory usage: 27.9+ MB
None

Column Names:
Index(['InvoiceNo', 'StockCode', 'Description', 'Quantity', 'InvoiceDate',
       'UnitPrice', 'CustomerID', 'Country'],
      dtype='object')
```

```
Column Names:
Index(['InvoiceNo', 'StockCode', 'Description', 'Quantity', 'InvoiceDate',
       'UnitPrice', 'CustomerID', 'Country'],
      dtype='object')

First few rows after cleaning:
  InvoiceNo StockCode                          Description  Quantity      InvoiceDate  UnitPrice  CustomerID         Country
0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6  01-12-2010 8.26       2.55     17850.0  United Kingdom
1    536365     71053                  WHITE METAL LANTERN         6  01-12-2010 8.26       3.39     17850.0  United Kingdom
2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8  01-12-2010 8.26       2.75     17850.0  United Kingdom
3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6  01-12-2010 8.26       3.39     17850.0  United Kingdom
4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6  01-12-2010 8.26       3.39     17850.0  United Kingdom
```

```
Using 'InvoiceNo' as the invoice number column
Using 'Description' as the product description column

Top 20 Frequent Itemsets (by support):
      support                              itemsets
487  0.090717  (WHITE HANGING HEART T-LIGHT HOLDER)
349  0.084903           (REGENCY CAKESTAND 3 TIER)
202  0.074042            (JUMBO BAG RED RETROSPOT)
283  0.063046                      (PARTY BUNTING)
30   0.062416      (ASSORTED COLOUR BIRD ORNAMENT)
234  0.059892            (LUNCH BAG RED RETROSPOT)
395  0.054890   (SET OF 3 CAKE TINS PANTRY DESIGN )
313  0.053808                            (POSTAGE)
226  0.048355            (LUNCH BAG  BLACK SKULL.)
269  0.046913     (PACK OF 72 RETROSPOT CAKE CASES)
443  0.045741                      (SPOTTY BUNTING)
235  0.045110          (LUNCH BAG SPACEBOY DESIGN )
277  0.044615      (PAPER CHAIN KIT 50'S CHRISTMAS )
229  0.044570                 (LUNCH BAG CARS BLUE)
253  0.044344     (NATURAL SLATE HEART CHALKBOARD )
172  0.043804               (HEART OF WICKER SMALL)
188  0.043488             (JAM MAKING SET WITH JARS)
233  0.042857            (LUNCH BAG PINK POLKADOT)
236  0.041280              (LUNCH BAG SUKI DESIGN )
26   0.040874            (ALARM CLOCK BAKELIKE RED )
```

```
Top 20 Association Rules (by confidence):
antecedents consequents support confidence lift
{ROSES REGENCY TEACUP AND SAUCER , REGENCY CAKESTAND 3 TIER, PINK REGENCY TEACUP AND SAUCER} {GREEN REGENCY TEACUP AND SAUCER} 0.010861 0.889299 26.921613
{PINK REGENCY TEACUP AND SAUCER, ROSES REGENCY TEACUP AND SAUCER } {GREEN REGENCY TEACUP AND SAUCER} 0.017891 0.880266 26.648164
{GREEN REGENCY TEACUP AND SAUCER, REGENCY CAKESTAND 3 TIER, PINK REGENCY TEACUP AND SAUCER} {ROSES REGENCY TEACUP AND SAUCER } 0.010861 0.879562 23.346270
{REGENCY CAKESTAND 3 TIER, PINK REGENCY TEACUP AND SAUCER} {GREEN REGENCY TEACUP AND SAUCER} 0.012348 0.858934 26.002386
{REGENCY CAKESTAND 3 TIER, PINK REGENCY TEACUP AND SAUCER} {ROSES REGENCY TEACUP AND SAUCER } 0.012213 0.849530 22.549122
{REGENCY TEA PLATE GREEN } {REGENCY TEA PLATE ROSES } 0.010455 0.843636 55.059679
{GREEN REGENCY TEACUP AND SAUCER, PINK REGENCY TEACUP AND SAUCER} {ROSES REGENCY TEACUP AND SAUCER } 0.017891 0.842887 22.372815
{SET/6 RED SPOTTY PAPER CUPS} {SET/6 RED SPOTTY PAPER PLATES} 0.010635 0.828070 56.538084
{GREEN REGENCY TEACUP AND SAUCER, REGENCY CAKESTAND 3 TIER} {ROSES REGENCY TEACUP AND SAUCER } 0.014241 0.825065 21.899759
{WOODEN TREE CHRISTMAS SCANDINAVIAN} {WOODEN STAR CHRISTMAS SCANDINAVIAN} 0.010230 0.819495 41.707763
{POPPY'S PLAYHOUSE BEDROOM } {POPPY'S PLAYHOUSE KITCHEN} 0.011492 0.799373 50.825466
{PINK REGENCY TEACUP AND SAUCER} {GREEN REGENCY TEACUP AND SAUCER} 0.021226 0.796954 24.126079
{JUMBO BAG PINK POLKADOT, JUMBO BAG STRAWBERRY} {JUMBO BAG RED RETROSPOT} 0.010500 0.792517 10.703562
{JUMBO BAG PINK POLKADOT, JUMBO STORAGE BAG SUKI} {JUMBO BAG RED RETROSPOT} 0.010050 0.785211 10.604892
{SMALL MARSHMALLOWS PINK BOWL} {SMALL DOLLY MIX DESIGN ORANGE BOWL} 0.010185 0.782007 47.935728
{ALARM CLOCK BAKELIKE PINK, ALARM CLOCK BAKELIKE GREEN} {ALARM CLOCK BAKELIKE RED } 0.012078 0.779070 19.060152
{PINK REGENCY TEACUP AND SAUCER} {ROSES REGENCY TEACUP AND SAUCER } 0.020324 0.763113 20.255366
{GREEN REGENCY TEACUP AND SAUCER, REGENCY CAKESTAND 3 TIER, ROSES REGENCY TEACUP AND SAUCER } {PINK REGENCY TEACUP AND SAUCER} 0.010861 0.762658 28.635171
{GREEN REGENCY TEACUP AND SAUCER} {ROSES REGENCY TEACUP AND SAUCER } 0.025101 0.759891 20.169830
{REGENCY CAKESTAND 3 TIER, PINK REGENCY TEACUP AND SAUCER} {GREEN REGENCY TEACUP AND SAUCER, ROSES REGENCY TEACUP AND SAUCER } 0.010861 0.755486 30.097364

Analysis complete.
Minimum support used: 0.01
Minimum confidence used: 0.5
```

CONCLUSION:-

The Apriori algorithm is an efficient method for discovering frequent itemsets and generating association rules from a transaction dataset. By leveraging the Apriori property, the algorithm significantly reduces the search space, making it practical for large datasets. The generated association rules can provide valuable insights into customer purchasing patterns, aiding in recommendation systems and strategic decision-making. However, the efficiency of the algorithm can be further improved using techniques like Hash-based itemset counting and the FP-Growth algorithm for larger and more complex datasets.