Midterm Project Report

NAME: Anuveer Rayudu
UCID: ar2698
Email Address: ar2698@njit.edu

Date: 10/13/2024
Professor: Yasser Abduallah
Course: CS 634101 Data Mining

Implementation and Code Usage

Abstract:
In this project, I explored and evaluated three different algorithms for association rule mining: Brute Force, Apriori, and FP-Growth. These algorithms were applied to various transactional datasets to extract frequent item sets and generate association rules. The comparison focused on assessing their computational efficiency, scalability, and accuracy in identifying meaningful patterns. By analyzing the results from each algorithm, I was able to draw insights into their strengths and weaknesses, providing a comprehensive understanding of their practical applications in data mining tasks.

Introduction:

Data mining is important for finding useful patterns and insights from large datasets. One key technique in data mining is association rule mining, which helps uncover relationships between items in transactional data. This technique is often used in market basket analysis and recommendation systems. In this project, I worked with three different algorithms—Brute Force, Apriori, and FP-Growth—to find frequent itemsets and create association rules.

- **Brute Force** checks all possible item combinations to find patterns, but it takes a lot of time and computing power.
- **Apriori** improves this by using the fact that all subsets of a frequent itemset must also be frequent, reducing the number of combinations to check.
- **FP-Growth** is the most efficient because it builds an FP-Tree structure to find frequent itemsets without having to generate combinations.

I applied these algorithms to custom datasets from retailers like K-mart, Seven11, Costco, target, and Shoprite. My goal was to compare the algorithms based on how long they take to run, how accurate they are, and how well they handle large datasets.

Here's what I did step by step:

1. **Creating Datasets**: I created five custom datasets using data from different websites and additional data generated using ChatGPT, making sure each dataset had over 1,000+ transactions.
2. **Data Preparation**: I loaded these datasets into a Jupyter Notebook, where each row represented a transaction and each column an item.
3. **User Input**: During the process, I allowed user input to set support and confidence thresholds, which control which patterns get considered as frequent.
4. **Running the Algorithms**: I manually implemented the Brute Force method to generate frequent itemsets and association rules. Then, I applied the Apriori and FP-Growth algorithms to see how they compared in terms of speed and accuracy.
5. **Comparing Results**: After running all three algorithms, I compared their performance. The Brute Force method was the slowest and required the most computing power. Apriori was faster, but the FP-Growth method was the most efficient and scalable, especially for larger datasets.

Overall, this project helped me understand the strengths and weaknesses of different algorithms used in association rule mining and how to choose the right one based on the size and complexity of the dataset

Project Workflow:

This project follows a clear, step-by-step process using the Brute Force, Apriori, and FP-Growth algorithms.

We begin by loading transaction data from retailers such as K-mart, seven11, and Costco. Each transaction includes a list of items purchased by customers. To clean the data for analysis, we remove duplicate items and organize them in a specific order for consistency.

The user sets minimum values for support and confidence. These values help focus the analysis on frequent itemsets and strong associations, filtering out less significant patterns.

With the Apriori Algorithm, we create combinations of items, starting with individual items and then moving on to pairs, triplets, and more. Only item combinations that meet the minimum support level are considered, helping to save time and computing resources.

For each candidate itemset, we check how many transactions include that combination. If an itemset meets the minimum support threshold, we keep it; otherwise, it's discarded.

Once frequent itemsets are identified, we calculate how strong the relationships between items are. Confidence is measured by checking how often one item is purchased when another is also bought.

Finally, we create association rules from the frequent itemsets that meet both the support and confidence levels. These rules help identify which items are often bought together, offering useful insights for improving marketing strategies and product placements

Conclusion:

To sum up, the brute force approach proves to be impractical for handling large datasets due to its high computational cost. On the other hand, both Apriori and FP-Growth algorithms perform much better with large datasets, with FP-Growth standing out as the most efficient thanks to its FP-tree structure. This project showcased how both Apriori and FP-Growth can effectively identify frequent itemsets and generate association rules in a more efficient and scalable manner.

Screenshots :

Transactions CSV file images:

costco_data_new

Transaction ID	Filtered Transaction
TranC1	Garden Supplies, Canned Foods
TranC2	Tires, Optical Items, Packaged Foods
TranC3	Canned Foods, Cleaning Supplies
TranC4	Jewelry, Canned Foods, Packaged Foods, Optical Items
TranC5	Jewelry, Garden Supplies, Optical Items, Tires
TranC6	Garden Supplies
TranC7	Packaged Foods, Cleaning Supplies, Bakery Items
TranC8	Optical Items, Bakery Items, Garden Supplies, Optical Items
TranC9	Optical Items
TranC10	Canned Foods, Optical Items, Bulk Fruits, Garden Supplies
TranC11	Garden Supplies, Canned Foods, Bakery Items
TranC12	Bulk Fruits
TranC13	Packaged Foods, Cleaning Supplies, Tires, Bulk Fruits, Jewelry
TranC14	Bakery Items, Tires, Cleaning Supplies
TranC15	Garden Supplies
TranC16	Garden Supplies, Canned Foods, Tires, Bulk Fruits, Optical Items
TranC17	Packaged Foods, Bulk Fruits, Bakery Items, Tires, Canned Foods
TranC18	Tires
TranC19	Bakery Items
TranC20	Tires, Optical Items, Bulk Fruits, Jewelry
TranC21	Bulk Fruits, Optical Items, Tires, Garden Supplies, Packaged Foods
TranC22	Jewelry, Bulk Fruits, Optical Items
TranC23	Optical Items
TranC24	Optical Items, Packaged Foods, Jewelry, Cleaning Supplies, Bakery Items
TranC25	Jewelry, Bulk Fruits, Optical Items, Cleaning Supplies

kmart_data

Transaction ID	Filtered Transaction
TranK1	Beauty Products, Sports Equipment, Home Appliances, Footwear
TranK2	Home Appliances, Beauty Products, Pet Supplies
TranK3	Pet Supplies
TranK4	Toys, Home Appliances, Sports Equipment, Furniture, Clothing
TranK5	Beauty Products, Home Appliances, Pet Supplies, Furniture
TranK6	Home Appliances, Clothing, Footwear, Electronics, Beauty Products
TranK7	Pet Supplies
TranK8	Beauty Products, Footwear, Jewelry, Toys, Pet Supplies
TranK9	Sports Equipment, Beauty Products, Clothing, Electronics, Toys
TranK10	Jewelry
TranK11	Toys, Footwear, Jewelry, Furniture
TranK12	Furniture, Beauty Products, Jewelry
TranK13	Footwear
TranK14	Clothing, Home Appliances
TranK15	Home Appliances
TranK16	Toys, Clothing, Furniture, Electronics, Home Appliances
TranK17	Jewelry, Toys, Pet Supplies, Home Appliances, Electronics
TranK18	Clothing, Electronics, Footwear, Jewelry
TranK19	Electronics
TranK20	Home Appliances
TranK21	Toys, Beauty Products, Clothing
TranK22	Jewelry, Furniture, Sports Equipment, Clothing, Toys
TranK23	Electronics, Sports Equipment
TranK24	Toys, Jewelry, Footwear, Furniture
TranK25	Jewelry, Beauty Products, Electronics
TranK26	Sports Equipment, Pet Supplies, Jewelry, Footwear, Beauty Products

seven_eleven_data

Transaction ID	Filtered Transaction
TranS71	Cigarettes, Snacks, Gum, Energy Drinks, Batteries
TranS72	Snacks, Magazine
TranS73	Cigarettes, Batteries, Candy, Energy Drinks
TranS74	Magazine
TranS75	Candy, Snacks, Cigarettes, Batteries, Gum
TranS76	Milk, Batteries, Candy
TranS77	Snacks, Magazine, Energy Drinks
TranS78	Magazine
TranS79	Magazine
TranS710	Batteries, Milk, Toiletries, Gum
TranS711	Milk, Batteries, Soft Drinks, Candy
TranS712	Snacks, Gum, Candy, Toiletries
TranS713	Soft Drinks, Batteries, Toiletries, Gum
TranS714	Cigarettes, Snacks
TranS715	Milk
TranS716	Toiletries
TranS717	Milk, Gum, Energy Drinks, Candy, Snacks
TranS718	Milk, Magazine, Batteries
TranS719	Energy Drinks, Cigarettes, Gum, Snacks
TranS720	Batteries, Magazine, Energy Drinks, Gum
TranS721	Snacks, Milk, Gum, Batteries, Cigarettes
TranS722	Energy Drinks, Soft Drinks
TranS723	Gum
TranS724	Candy, Energy Drinks, Toiletries, Snacks, Magazine

shoprite_data

Transaction ID	Filtered Transaction
TranS1	Household Cleaning Products, Personal Care Products
TranS2	Household Cleaning Products, Meat, Personal Care Products
TranS3	Dairy, Snacks, Vegetables, Personal Care Products, Bread
TranS4	Snacks, Personal Care Products
TranS5	Personal Care Products, Frozen Foods, Vegetables
TranS6	Canned Goods, Frozen Foods, Dairy, Meat, Bakery Items
TranS7	Personal Care Products, Vegetables, Household Cleaning Products, Bread
TranS8	Vegetables, Household Cleaning Products, Personal Care Products
TranS9	Personal Care Products, Meat, Vegetables, Bakery Items
TranS10	Frozen Foods, Household Cleaning Products, Canned Goods, Snacks, Dairy
TranS11	Snacks, Dairy, Personal Care Products, Meat
TranS12	Bakery Items, Household Cleaning Products
TranS13	Bread, Dairy, Snacks, Household Cleaning Products, Frozen Foods
TranS14	Dairy, Canned Goods, Bakery Items
TranS15	Canned Goods
TranS16	Frozen Foods, Meat, Vegetables, Canned Goods
TranS17	Vegetables
TranS18	Bakery Items, Bread, Snacks, Household Cleaning Products
TranS19	Dairy, Vegetables, Snacks, Meat
TranS20	Frozen Foods, Meat, Canned Goods, Dairy, Bread
TranS21	Personal Care Products, Bread, Vegetables, Household Cleaning Products
TranS22	Meat, Frozen Foods
TranS23	Meat, Vegetables, Frozen Foods, Snacks
TranS24	Vegetables, Meat
TranS25	Vegetables, Frozen Foods, Bakery Items, Dairy, Snacks
TranS26	Snacks, Canned Goods, Dairy, Personal Care Products, Meat

target_data_new

Transaction ID	Filtered Transaction
TranT1	Beauty Products, Bedding, Gardening Supplies, Electronics
TranT2	Groceries, Gardening Supplies, Cleaning Supplies, Bedding
TranT3	Furniture, Electronics, Groceries, Gardening Supplies, Bedding
TranT4	Furniture, Cleaning Supplies, Clothes
TranT5	Furniture, Clothes, Bedding, Groceries, Electronics
TranT6	Clothes, Gardening Supplies, Stationery, Toys
TranT7	Cleaning Supplies, Toys
TranT8	Stationery, Cleaning Supplies, Electronics, Gardening Supplies, Clothes
TranT9	Stationery, Groceries
TranT10	Groceries, Electronics, Gardening Supplies, Furniture, Beauty Products
TranT11	Gardening Supplies, Furniture, Electronics, Stationery, Groceries
TranT12	Electronics, Stationery, Cleaning Supplies, Toys, Bedding
TranT13	Gardening Supplies, Beauty Products, Toys
TranT14	Toys
TranT15	Groceries, Toys, Cleaning Supplies, Stationery
TranT16	Furniture, Electronics, Toys, Beauty Products
TranT17	Clothes, Cleaning Supplies, Toys
TranT18	Groceries, Gardening Supplies, Electronics
TranT19	Beauty Products, Electronics, Toys
TranT20	Furniture, Groceries, Beauty Products, Stationery
TranT21	Furniture
TranT22	Cleaning Supplies, Electronics, Furniture, Groceries, Beauty Products
TranT23	Bedding, Groceries, Toys, Electronics, Stationery
TranT24	Electronics
TranT25	Furniture, Cleaning Supplies, Gardening Supplies, Electronics
TranT26	Bedding

Jupyter File ScreenShots:

In [5]:
```
import pandas as pd
import itertools
import time

# Load transactions from a CSV file, processing the items column into a list of sets
def load_transactions(file_path):
    df = pd.read_csv(file_path, usecols=[1], skiprows=1, names=['items'])
    return [set(item.split(", ")) for item in df['items']]

# Function to calculate support of an itemset within the transactions
def calculate_support(itemset, transactions):
    return sum(1 for transaction in transactions if itemset.issubset(transaction))

# Generate frequent itemsets based on minimum support
def find_frequent_itemsets(transactions, min_support, max_length=3):
    items_pool = {item for transaction in transactions for item in transaction}
    frequent_itemsets = []

    for size in range(1, max_length + 1):
        for combination in itertools.combinations(items_pool, size):
            support_count = calculate_support(set(combination), transactions)
            if support_count >= min_support:
                frequent_itemsets.append((set(combination), support_count))

    return frequent_itemsets

# Generate association rules based on frequent itemsets
def generate_rules(frequent_itemsets, transactions, min_confidence):
    rules = []

    for itemset, support_count in frequent_itemsets:
        for subset_size in range(1, len(itemset)):
            for subset in itertools.combinations(itemset, subset_size):
                subset = set(subset)
                remainder = itemset - subset
                if remainder:
                    subset_support = calculate_support(subset, transactions)
                    confidence = support_count / subset_support if subset_support > 0 else 0
                    if confidence >= min_confidence:
                        rules.append((subset, remainder, confidence))

    return rules

# Main function to process a CSV file, find itemsets, and generate rules
def process_transactions(file_path, min_support, min_confidence):
    transactions = load_transactions(file_path)
    min_support_count = int(min_support * len(transactions))  # Convert to absolute support count
    frequent_itemsets = find_frequent_itemsets(transactions, min_support_count)
    association_rules = generate_rules(frequent_itemsets, transactions, min_confidence)

    return frequent_itemsets, association_rules

# Start execution timer
start_time = time.time()

# Provided datasets
datasets = {
    'Target': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\target_data_new.csv',
    'Costco': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\costco_data_new.csv',
    '7-Eleven': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\seven_eleven_data.csv',
    'ShopRite': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\shoprite_data.csv',
    'K-Mart': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\kmart_data.csv'
}
```

```
    'K-Mart': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\kmart_data.csv'
}

# Minimum support and confidence
min_support = 0.05
min_confidence = 0.3

# Analyze each dataset
for store, file_path in datasets.items():
    itemsets, rules = process_transactions(file_path, min_support, min_confidence)
    print(f"Results for {store}:")
    print("Frequent Itemsets:", itemsets)
    print("Association Rules:", rules)
    print("\n")

# Total execution time
print(f"Total Execution Time: {time.time() - start_time:.2f} seconds")
```

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

# Function to load and filter transactions from the dataset
def load_and_filter_transactions(file_path, min_items=2):
    df = pd.read_csv(file_path)

    # Use 'Filtered Transaction' as the correct column for transaction data
    transactions = df['Filtered Transaction'].apply(lambda x: x.split(', '))

    # Filtering transactions based on the minimum number of items
    filtered_transactions = [tx for tx in transactions if len(tx) >= min_items]
    return filtered_transactions

# Function to analyze transactions and generate frequent itemsets and association rules
def perform_transaction_analysis(file_path, min_support, min_confidence):
    start_time = time()  # Start timer for each analysis

    # Load and filter transactions
    transactions = load_and_filter_transactions(file_path)

    # Encoding transactions into a format suitable for Apriori
    encoder = TransactionEncoder()
    transaction_array = encoder.fit(transactions).transform(transactions)
    transaction_df = pd.DataFrame(transaction_array, columns=encoder.columns_)

    # Finding frequent itemsets using Apriori algorithm
    frequent_itemsets = apriori(transaction_df, min_support=min_support, use_colnames=True)

    # Generating association rules based on the frequent itemsets
    rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=min_confidence)
```

```
    # Print filtered transactions
    print(f"Filtered Transactions from {file_path.split('/')[-1]}:")
    for transaction in transactions:
        print(transaction)

    # Print the generated association rules
    print("\nGenerated Association Rules:")
    print(rules[['antecedents', 'consequents', 'support', 'confidence']])

    # Print the time taken for each dataset analysis
    print(f"\nTotal Execution Time: {time() - start_time:.2f} seconds\n")

# Main function to handle the execution of transaction analysis
def execute_analysis():
    datasets = {
    'Target': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\target_data_new.csv',
    'Costco': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\costco_data_new.csv',
    '7-Eleven': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\seven_eleven_data.csv',
    'ShopRite': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\shoprite_data.csv',
    'K-Mart': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\kmart_data.csv'
}


    # Display the available datasets to the user
    print("Please select dataset(s):")
    for key, value in datasets.items():
        print(f"{key} - {value.split('/')[-1]}")

    # Accept user input for selected datasets and analysis parameters
    selected_datasets = input("Enter your choices (e.g., Target, 7-Eleven): ").split(',')
    min_support = float(input("Enter minimum support value (e.g., 0.05): "))
    min_confidence = float(input("Enter minimum confidence value (e.g., 0.5): "))
```

```
    # Accept user input for selected datasets and analysis parameters
    selected_datasets = input("Enter your choices (e.g., Target, 7-Eleven): ").split(',')
    min_support = float(input("Enter minimum support value (e.g., 0.05): "))
    min_confidence = float(input("Enter minimum confidence value (e.g., 0.5): "))

    # Perform analysis on selected datasets
    for dataset in selected_datasets:
        dataset = dataset.strip()  # Remove any leading/trailing spaces
        if dataset in datasets:
            perform_transaction_analysis(datasets[dataset], min_support, min_confidence)
        else:
            print(f"Dataset {dataset} not found. Please check your input.")

# Execute the analysis if the script is run directly
if __name__ == "__main__":
    execute_analysis()
```

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

# Function to load transactions from the dataset
def load_transactions(file_path):
    # Read CSV and split the transactions by commas
    df = pd.read_csv(file_path)
    transactions = df['Filtered Transaction'].str.split(', ').tolist()
    return transactions

# Function to perform frequent itemset and association rule analysis
def perform_analysis(file_path, min_support, min_confidence):
    start_time = time.time()  # Start timing the analysis for this dataset

    # Load the transaction data
    transactions = load_transactions(file_path)

    # Encode the transactions into a format suitable for fpgrowth
    encoder = TransactionEncoder()
    transformed_data = encoder.fit(transactions).transform(transactions)
    transactions_df = pd.DataFrame(transformed_data, columns=encoder.columns_)

    # Apply the fpgrowth algorithm to find frequent itemsets
    frequent_itemsets = fpgrowth(transactions_df, min_support=min_support, use_colnames=True)

    # Derive association rules from the frequent itemsets
    rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=min_confidence)

    # Print the results
    print(f"\nFrequent Itemsets from {file_path.split('/')[-1]}:")
    print(frequent_itemsets)
    print("\nGenerated Association Rules:")
```

```
    # Print execution time for this dataset in the required format
    print(f"Total Execution Time: {time.time() - start_time:.2f} seconds\n")

# Main function to execute the analysis based on user input
def execute_analysis():
    # Dictionary mapping dataset numbers to file paths
    datasets = {
    'Target': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\target_data_new.csv',
    'Costco': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\costco_data_new.csv',
    '7-Eleven': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\seven_eleven_data.csv',
    'ShopRite': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\shoprite_data.csv',
    'K-Mart': 'C:\\Users\\anuve\\Downloads\\dataset2\\dataset2\\kmart_data.csv'
}


    # Display options to the user
    print("Choose dataset(s) for analysis:")
    for index, name in enumerate(['Target', 'Costco', '7-Eleven', 'ShopRite', 'K-Mart'], start=1):
        print(f"{index} - {name}")

    # Accept user input for datasets and parameters
    selected_datasets = input("Enter dataset numbers (space-separated, e.g., 1 3): ").split()
    min_support_value = float(input("Enter minimum support (e.g., 0.05): "))
    min_confidence_value = float(input("Enter minimum confidence (e.g., 0.5): "))

    # Perform analysis for each selected dataset
    for dataset_key in selected_datasets:
        if dataset_key in datasets:
            perform_analysis(datasets[dataset_key], min_support_value, min_confidence_value)
        else:
            print(f"Invalid choice: {dataset_key}")
```

```
    # Accept user input for datasets and parameters
    selected_datasets = input("Enter dataset numbers (space-separated, e.g., 1 3): ").split()
    min_support_value = float(input("Enter minimum support (e.g., 0.05): "))
    min_confidence_value = float(input("Enter minimum confidence (e.g., 0.5): "))

    # Perform analysis for each selected dataset
    for dataset_key in selected_datasets:
        if dataset_key in datasets:
            perform_analysis(datasets[dataset_key], min_support_value, min_confidence_value)
        else:
            print(f"Invalid choice: {dataset_key}")

# Run the analysis when the script is executed
if __name__ == "__main__":
    execute_analysis()
```

Ouput ScreenShots:

```
Results for Target:
Frequent Itemsets: [({'Toys'}, 278), ({'Gardening Supplies'}, 287), ({'Clothes'}, 287), ({'Bedding'}, 291), ({'Furniture'}, 329), ({'Beauty Products'}, 292), ({'Station
ery'}, 320), ({'Groceries'}, 286), ({'Electronics'}, 309), ({'Cleaning Supplies'}, 295), ({'Toys', 'Gardening Supplies'}, 71), ({'Toys', 'Clothes'}, 77), ({'Toys', 'Bed
ding'}, 73), ({'Furniture', 'Toys'}, 82), ({'Beauty Products', 'Toys'}, 77), ({'Stationery', 'Toys'}, 87), ({'Groceries', 'Toys'}, 82), ({'Toys', 'Electronics'}, 91),
({'Toys', 'Cleaning Supplies'}, 74), ({'Clothes', 'Gardening Supplies'}, 89), ({'Gardening Supplies', 'Bedding'}, 85), ({'Furniture', 'Gardening Supplies'}, 97), ({'Bea
uty Products', 'Gardening Supplies'}, 82), ({'Stationery', 'Gardening Supplies'}, 95), ({'Groceries', 'Gardening Supplies'}, 83), ({'Electronics', 'Gardening Supplie
s'}, 87), ({'Gardening Supplies', 'Cleaning Supplies'}, 89), ({'Clothes', 'Bedding'}, 80), ({'Furniture', 'Clothes'}, 89), ({'Beauty Products', 'Clothes'}, 77), ({'Stat
ionery', 'Clothes'}, 93), ({'Groceries', 'Clothes'}, 73), ({'Clothes', 'Electronics'}, 81), ({'Clothes', 'Cleaning Supplies'}, 87), ({'Furniture', 'Bedding'}, 90), ({'B
eauty Products', 'Bedding'}, 89), ({'Stationery', 'Bedding'}, 97), ({'Groceries', 'Bedding'}, 83), ({'Electronics', 'Bedding'}, 92), ({'Cleaning Supplies', 'Bedding'},
81), ({'Furniture', 'Beauty Products'}, 94), ({'Stationery', 'Furniture'}, 101), ({'Furniture', 'Groceries'}, 100), ({'Furniture', 'Electronics'}, 98), ({'Furniture',
'Cleaning Supplies'}, 101), ({'Stationery', 'Beauty Products'}, 97), ({'Groceries', 'Beauty Products'}, 79), ({'Beauty Products', 'Electronics'}, 92), ({'Beauty Product
s', 'Cleaning Supplies'}, 83), ({'Stationery', 'Groceries'}, 87), ({'Stationery', 'Electronics'}, 99), ({'Stationery', 'Cleaning Supplies'}, 97), ({'Groceries', 'Electr
onics'}, 97), ({'Groceries', 'Cleaning Supplies'}, 91), ({'Electronics', 'Cleaning Supplies'}, 97)]
Association Rules: [({'Toys'}, {'Stationery'}, 0.3129496402877698), ({'Toys'}, {'Electronics'}, 0.3273381294964029), ({'Clothes'}, {'Gardening Supplies'}, 0.31010452961
672474), ({'Gardening Supplies'}, {'Clothes'}, 0.31010452961672474), ({'Gardening Supplies'}, {'Furniture'}, 0.33797909407665505), ({'Gardening Supplies'}, {'Stationer
y'}, 0.3310104529616725), ({'Gardening Supplies'}, {'Electronics'}, 0.30313588850174217), ({'Gardening Supplies'}, {'Cleaning Supplies'}, 0.31010452961672474), ({'Clean
ing Supplies'}, {'Gardening Supplies'}, 0.3016949152542373), ({'Clothes'}, {'Furniture'}, 0.31010452961672474), ({'Clothes'}, {'Stationery'}, 0.32404181184668899), ({'Cl
othes'}, {'Cleaning Supplies'}, 0.30313588850174217), ({'Bedding'}, {'Furniture'}, 0.30927835051546393), ({'Beauty Products'}, {'Bedding'}, 0.3047945205479452), ({'Bedd
ing'}, {'Beauty Products'}, 0.30584192439862545), ({'Stationery'}, {'Bedding'}, 0.303125), ({'Bedding'}, {'Stationery'}, 0.3333333333333333), ({'Bedding'}, {'Electronic
s'}, 0.3161512027491409), ({'Beauty Products'}, {'Furniture'}, 0.3219178082191781), ({'Stationery'}, {'Furniture'}, 0.315625), ({'Furniture'}, {'Stationery'}, 0.3069908
814589666), ({'Furniture'}, {'Groceries'}, 0.303951367781155), ({'Groceries'}, {'Furniture'}, 0.34965034965034963), ({'Electronics'}, {'Furniture'}, 0.3171521035598705
6), ({'Furniture'}, {'Cleaning Supplies'}, 0.3069908814589666), ({'Cleaning Supplies'}, {'Furniture'}, 0.3423728813559322), ({'Stationery'}, {'Beauty Products'}, 0.3031
25), ({'Beauty Products'}, {'Stationery'}, 0.3321917808219178), ({'Beauty Products'}, {'Electronics'}, 0.3150684931506849), ({'Groceries'}, {'Stationery'}, 0.3041958041
958042), ({'Stationery'}, {'Electronics'}, 0.309375), ({'Electronics'}, {'Stationery'}, 0.32038834951456313), ({'Stationery'}, {'Cleaning Supplies'}, 0.303125), ({'Clea
ning Supplies'}, {'Stationery'}, 0.3288135593220339), ({'Groceries'}, {'Electronics'}, 0.33916083916083917), ({'Electronics'}, {'Groceries'}, 0.313915857605178), ({'Gro
ceries'}, {'Cleaning Supplies'}, 0.3181818181818182), ({'Cleaning Supplies'}, {'Groceries'}, 0.30847457627118646), ({'Electronics'}, {'Cleaning Supplies'}, 0.3139158576
05178), ({'Cleaning Supplies'}, {'Electronics'}, 0.3288135593220339)]


Results for Costco:
Frequent Itemsets: [({'Optical Items'}, 513), ({'Packaged Foods'}, 285), ({'Bakery Items'}, 321), ({'Tires'}, 329), ({'Bulk Fruits'}, 307), ({'Garden Supplies'}, 300),
({'Jewelry'}, 301), ({'Canned Foods'}, 277), ({'Cleaning Supplies'}, 295), ({'Packaged Foods', 'Optical Items'}, 143), ({'Bakery Items', 'Optical Items'}, 163), ({'Tire
s', 'Optical Items'}, 172), ({'Bulk Fruits', 'Optical Items'}, 167), ({'Optical Items', 'Garden Supplies'}, 152), ({'Optical Items', 'Jewelry'}, 157), ({'Canned Foods',
'Optical Items'}, 153), ({'Optical Items', 'Cleaning Supplies'}, 158), ({'Packaged Foods', 'Bakery Items'}, 87), ({'Packaged Foods', 'Tires'}, 95), ({'Packaged Foods',
'Bulk Fruits'}, 74), ({'Packaged Foods', 'Garden Supplies'}, 82), ({'Packaged Foods', 'Jewelry'}, 87), ({'Packaged Foods', 'Canned Foods'}, 77), ({'Packaged Foods', 'Cl
eaning Supplies'}, 83), ({'Bakery Items', 'Tires'}, 102), ({'Bakery Items', 'Bulk Fruits'}, 91), ({'Bakery Items', 'Garden Supplies'}, 97), ({'Bakery Items', 'Jewelr
y'}, 96), ({'Bakery Items', 'Canned Foods'}, 93), ({'Bakery Items', 'Cleaning Supplies'}, 100), ({'Bulk Fruits', 'Tires'}, 91), ({'Tires', 'Garden Supplies'}, 106),
({'Tires', 'Jewelry'}, 101), ({'Canned Foods', 'Tires'}, 82), ({'Tires', 'Cleaning Supplies'}, 93), ({'Bulk Fruits', 'Garden Supplies'}, 87), ({'Bulk Fruits', 'Jewelr
y'}, 93), ({'Canned Foods', 'Bulk Fruits'}, 80), ({'Bulk Fruits', 'Cleaning Supplies'}, 105), ({'Garden Supplies', 'Jewelry'}, 100), ({'Canned Foods', 'Garden Supplie
s'}, 84), ({'Garden Supplies', 'Cleaning Supplies'}, 79), ({'Canned Foods', 'Jewelry'}, 68), ({'Cleaning Supplies', 'Jewelry'}, 92), ({'Canned Foods', 'Cleaning Supplie
s'}, 77), ({'Bakery Items', 'Tires', 'Optical Items'}, 52), ({'Bulk Fruits', 'Bakery Items', 'Optical Items'}, 53), ({'Bakery Items', 'Optical Items', 'Cleaning Supplie
s'}, 56), ({'Tires', 'Optical Items', 'Garden Supplies'}, 56), ({'Tires', 'Optical Items', 'Jewelry'}, 56), ({'Bulk Fruits', 'Optical Items', 'Jewelry'}, 52), ({'Bulk F
ruits', 'Optical Items', 'Cleaning Supplies'}, 55), ({'Optical Items', 'Garden Supplies', 'Jewelry'}, 52)]
Association Rules: [({'Packaged Foods'}, {'Optical Items'}, 0.5017543859649123), ({'Bakery Items'}, {'Optical Items'}, 0.5077881619937694), ({'Optical Items'}, {'Bakery
Items'}, 0.31773879142300193), ({'Tires'}, {'Optical Items'}, 0.5227963525835866), ({'Optical Items'}, {'Tires'}, 0.33528265107212474), ({'Bulk Fruits'}, {'Optical Item
s'}, 0.5439739413680782), ({'Optical Items'}, {'Bulk Fruits'}, 0.3255360623781676), ({'Garden Supplies'}, {'Optical Items'}, 0.5066666666666667), ({'Optical Items'},
{'Jewelry'}, 0.3060428849902534), ({'Jewelry'}, {'Optical Items'}, 0.521594684385382), ({'Canned Foods'}, {'Optical Items'}, 0.5523465703971119), ({'Optical Items'},
```

```
Association Rules: [({'Bread'}, {'Household Cleaning Products'}, 0.3519163763066202), ({'Household Cleaning Products'}, {'Bread'}, 0.3126934984520124), ({'Bread'}, {'Sn
acks'}, 0.3310104529616725), ({'Snacks'}, {'Bread'}, 0.3125), ({'Bread'}, {'Canned Goods'}, 0.30313588850174217), ({'Bread'}, {'Frozen Foods'}, 0.3170731707317073),
({'Bread'}, {'Vegetables'}, 0.3310104529616725), ({'Household Cleaning Products'}, {'Snacks'}, 0.33436532507739936), ({'Snacks'}, {'Household Cleaning Products'}, 0.355
26315789473684), ({'Household Cleaning Products'}, {'Meat'}, 0.30030959752321984), ({'Meat'}, {'Household Cleaning Products'}, 0.3201320132013201), ({'Bakery Items'},
{'Household Cleaning Products'}, 0.35018050541516244), ({'Household Cleaning Products'}, {'Bakery Items'}, 0.300309597523219840), ({'Personal Care Products'}, {'Household
d Cleaning Products'}, 0.3148148148148148), ({'Household Cleaning Products'}, {'Frozen Foods'}, 0.309597523219814260), ({'Frozen Foods'}, {'Household Cleaning Product
s'}, 0.3184713375796178), ({'Dairy'}, {'Snacks'}, 0.3356401384083045), ({'Snacks'}, {'Dairy'}, 0.31907894736842111), ({'Bakery Items'}, {'Snacks'}, 0.30324909747292417),
({'Personal Care Products'}, {'Snacks'}, 0.3), ({'Frozen Foods'}, {'Snacks'}, 0.3184713375796178), ({'Snacks'}, {'Frozen Foods'}, 0.32894736842105265), ({'Dairy'}, {'Me
at'}, 0.30103806228373703), ({'Meat'}, {'Canned Goods'}, 0.30033003300330036), ({'Personal Care Products'}, {'Meat'}, 0.3074074074074074), ({'Frozen Foods'}, {'Meat'},
0.31847133757961783), ({'Meat'}, {'Frozen Foods'}, 0.33003300330033003), ({'Meat'}, {'Vegetables'}, 0.3201320132013201), ({'Dairy'}, {'Canned Goods'}, 0.31141868512110726), ({'Dairy'}, {'Bakery Items'}, 0.30103806228373703), ({'Bakery Items'}, {'Dairy'}, 0.3140794223826715), ({'Dairy'}, {'Frozen Foods'}, 0.356401384083045), ({'Frozen Foods'}, {'Dairy'}, 0.32802547770700635), ({'Dairy'}, {'Vegetables'}, 0.3460207612456747), ({'Vegetables'}, {'Dairy'}, 0.3067484662576687), ({'Bakery Items'}, {'Canned Goods'}, 0.30685920577617326), ({'Personal Care Products'}, {'Canned Goods'}, 0.337037037037037), ({'Frozen Foods'}, {'Canned Goods'}, 0.34076433121019106), ({'Canned Goods'}, {'Frozen Foods'}, 0.33229813664596275), ({'Vegetables'}, {'Canned Goods'}, 0.30368098159509205), ({'Canned Goods'}, {'Vegetables'}, 0.30745341614906835), ({'Bakery Items'}, {'Vegetables'}, 0.3176895306859206), ({'Personal Care Products'}, {'Frozen Foods'}, 0.3037037037037037), ({'Personal Care Products'}, {'Vegetables'}, 0.3111111111111111), ({'Vegetables'}, {'Frozen Foods'}, 0.3220858895705521), ({'Frozen Foods'}, {'Vegetables'}, 0.3343949044585987)]


Results for K-Mart:
Frequent Itemsets: [({'Sports Equipment'}, 277), ({'Clothing'}, 306), ({'Toys'}, 289), ({'Pet Supplies'}, 303), ({'Furniture'}, 286), ({'Beauty Products'}, 297), ({'Jew
elry'}, 327), ({'Home Appliances'}, 301), ({'Electronics'}, 284), ({'Footwear'}, 328), ({'Sports Equipment', 'Clothing'}, 85), ({'Sports Equipment', 'Toys'}, 83), ({'Sp
orts Equipment', 'Pet Supplies'}, 87), ({'Sports Equipment', 'Furniture'}, 75), ({'Sports Equipment', 'Beauty Products'}, 95), ({'Sports Equipment', 'Jewelry'}, 97),
({'Sports Equipment', 'Home Appliances'}, 85), ({'Sports Equipment', 'Electronics'}, 68), ({'Sports Equipment', 'Footwear'}, 97), ({'Clothing', 'Toys'}, 86), ({'Pet Sup
plies', 'Clothing'}, 98), ({'Furniture', 'Clothing'}, 91), ({'Clothing', 'Beauty Products'}, 87), ({'Clothing', 'Jewelry'}, 100), ({'Home Appliances', 'Clothing'}, 90),
({'Clothing', 'Electronics'}, 85), ({'Clothing', 'Footwear'}, 91), ({'Pet Supplies', 'Toys'}, 84), ({'Furniture', 'Toys'}, 92), ({'Beauty Products', 'Toys'}, 93), ({'To
ys', 'Jewelry'}, 88), ({'Home Appliances', 'Toys'}, 81), ({'Toys', 'Electronics'}, 87), ({'Toys', 'Footwear'}, 89), ({'Pet Supplies', 'Furniture'}, 96), ({'Pet Supplie
s', 'Beauty Products'}, 93), ({'Pet Supplies', 'Jewelry'}, 102), ({'Pet Supplies', 'Home Appliances'}, 81), ({'Pet Supplies', 'Electronics'}, 76), ({'Pet Supplies', 'Fo
otwear'}, 89), ({'Furniture', 'Beauty Products'}, 85), ({'Furniture', 'Jewelry'}, 97), ({'Furniture', 'Home Appliances'}, 78), ({'Furniture', 'Electronics'}, 80), ({'Fu
rniture', 'Footwear'}, 89), ({'Beauty Products', 'Jewelry'}, 80), ({'Home Appliances', 'Beauty Products'}, 94), ({'Beauty Products', 'Electronics'}, 91), ({'Beauty Prod
ucts', 'Footwear'}, 88), ({'Home Appliances', 'Jewelry'}, 84), ({'Electronics', 'Jewelry'}, 96), ({'Footwear', 'Jewelry'}, 98), ({'Home Appliances', 'Electronics'}, 8
4), ({'Home Appliances', 'Footwear'}, 96), ({'Electronics', 'Footwear'}, 92)]
Association Rules: [({'Sports Equipment'}, {'Clothing'}, 0.30685920577617326), ({'Sports Equipment'}, {'Pet Supplies'}, 0.3140794223826715), ({'Sports Equipment'}, {'Be
auty Products'}, 0.34296028880866425), ({'Beauty Products'}, {'Sports Equipment'}, 0.31986531986531985), ({'Sports Equipment'}, {'Jewelry'}, 0.35018050541516244), ({'Sp
orts Equipment'}, {'Home Appliances'}, 0.30685920577617326), ({'Sports Equipment'}, {'Footwear'}, 0.35018050541516244), ({'Pet Supplies'}, {'Clothing'}, 0.3234323432343
234), ({'Clothing'}, {'Pet Supplies'}, 0.3202614379084967), ({'Furniture'}, {'Clothing'}, 0.3181818181818182), ({'Clothing'}, {'Jewelry'}, 0.32679738562091504), ({'Jewe
lry'}, {'Clothing'}, 0.3058103975535168), ({'Furniture'}, {'Toys'}, 0.32167832167832167), ({'Toys'}, {'Furniture'}, 0.31833910034602075), ({'Beauty Products'}, {'Toy
s'}, 0.31313131313131315), ({'Toys'}, {'Beauty Products'}, 0.3217993079584775), ({'Toys'}, {'Jewelry'}, 0.3044982698961938), ({'Toys'}, {'Electronics'}, 0.30103806228373703), ({'Electronics'}, {'Toys'}, 0.30633802816901406), ({'Toys'}, {'Footwear'}, 0.3079584775086505), ({'Pet Supplies'}, {'Furniture'}, 0.31683168316831684), ({'Furniture', {'Pet Supplies'}, 0.3356643356643357), ({'Pet Supplies'}, {'Beauty Products'}, 0.3069306930693069), ({'Beauty Products'}, {'Pet Supplies'}, 0.31313131313131315),
({'Pet Supplies'}, {'Jewelry'}, 0.33663366336633666), ({'Jewelry'}, {'Pet Supplies'}, 0.3119266055045872), ({'Furniture'}, {'Jewelry'}, 0.33916083916083917), ({'Furnitu
re'}, {'Footwear'}, 0.3111888111888112), ({'Home Appliances'}, {'Beauty Products'}, 0.3122923588039867), ({'Beauty Products'}, {'Home Appliances'}, 0.3164983164983165),
({'Beauty Products'}, {'Electronics'}, 0.3063973063973064), ({'Electronics'}, {'Beauty Products'}, 0.3204225352112676), ({'Electronics'}, {'Jewelry'}, 0.338028169014084
5), ({'Home Appliances'}, {'Footwear'}, 0.31893687707641194), ({'Electronics'}, {'Footwear'}, 0.323943661971831)]


Total Execution Time: 0.08 seconds
```

```
Please select dataset(s):
Target - target_data_new.csv
Costco - costco_data_new.csv
7-Eleven - seven_eleven_data.csv
ShopRite - shoprite_data.csv
K-Mart - kmart_data.csv
Enter your choices (e.g., Target, 7-Eleven): Target,Shoprite
Enter minimum support value (e.g., 0.05): 0.05
Enter minimum confidence value (e.g., 0.5): 0.3
Filtered Transactions from target_data_new.csv:
['Beauty Products', 'Bedding', 'Gardening Supplies', 'Electronics']
['Groceries', 'Gardening Supplies', 'Cleaning Supplies', 'Bedding']
['Furniture', 'Electronics', 'Groceries', 'Gardening Supplies', 'Bedding']
['Furniture', 'Cleaning Supplies', 'Clothes']
['Furniture', 'Clothes', 'Bedding', 'Groceries', 'Electronics']
['Clothes', 'Gardening Supplies', 'Stationery', 'Toys']
['Cleaning Supplies', 'Toys']
['Stationery', 'Cleaning Supplies', 'Electronics', 'Gardening Supplies', 'Clothes']
['Stationery', 'Groceries']
['Groceries', 'Electronics', 'Gardening Supplies', 'Furniture', 'Beauty Products']
['Gardening Supplies', 'Furniture', 'Electronics', 'Stationery', 'Groceries']
['Electronics', 'Stationery', 'Cleaning Supplies', 'Toys', 'Bedding']
['Gardening Supplies', 'Beauty Products', 'Toys']
['Groceries', 'Toys', 'Cleaning Supplies', 'Stationery']
['Furniture', 'Electronics', 'Toys', 'Beauty Products']
['Clothes', 'Cleaning Supplies', 'Toys']
['Groceries', 'Gardening Supplies', 'Electronics']
['Beauty Products', 'Electronics', 'Toys']
['Furniture', 'Groceries', 'Beauty Products', 'Stationery']
['Cleaning Supplies', 'Electronics', 'Furniture', 'Groceries', 'Beauty Products']
['Bedding', 'Groceries', 'Toys', 'Electronics', 'Stationery']
['Furniture', 'Cleaning Supplies', 'Gardening Supplies', 'Electronics']
['Cleaning Supplies', 'Stationery', 'Electronics']
['Gardening Supplies', 'Toys', 'Clothes', 'Bedding']
['Bedding', 'Stationery', 'Groceries', 'Clothes', 'Furniture']
['Electronics', 'Clothes', 'Toys', 'Groceries']
['Cleaning Supplies', 'Electronics', 'Stationery', 'Gardening Supplies']
['Toys', 'Furniture', 'Cleaning Supplies']
['Cleaning Supplies', 'Stationery']
['Electronics', 'Beauty Products']
['Bedding', 'Toys', 'Electronics', 'Groceries']
['Stationery', 'Clothes']
['Beauty Products', 'Clothes']
['Beauty Products', 'Gardening Supplies', 'Cleaning Supplies', 'Furniture', 'Stationery']
['Toys', 'Cleaning Supplies', 'Beauty Products', 'Stationery', 'Electronics']
['Stationery', 'Electronics', 'Cleaning Supplies', 'Furniture']
['Groceries', 'Toys', 'Cleaning Supplies', 'Gardening Supplies']
['Electronics', 'Groceries']
['Beauty Products', 'Toys']
['Beauty Products', 'Stationery', 'Furniture']
['Electronics', 'Groceries', 'Cleaning Supplies', 'Toys']
['Gardening Supplies', 'Clothes']
['Furniture', 'Gardening Supplies', 'Bedding']
['Groceries', 'Toys']
['Beauty Products', 'Furniture']
['Gardening Supplies', 'Cleaning Supplies', 'Furniture', 'Groceries', 'Toys']
['Toys', 'Bedding']
['Groceries', 'Clothes', 'Beauty Products', 'Stationery']
['Gardening Supplies', 'Cleaning Supplies', 'Stationery', 'Clothes']
['Electronics', 'Bedding', 'Furniture', 'Groceries']
['Electronics', 'Furniture']
['Toys', 'Clothes', 'Groceries', 'Electronics', 'Gardening Supplies']
['Furniture', 'Electronics', 'Beauty Products', 'Groceries']
['Cleaning Supplies', 'Groceries', 'Gardening Supplies', 'Stationery']
['Bedding', 'Beauty Products', 'Furniture', 'Electronics', 'Cleaning Supplies']
['Beauty Products', 'Cleaning Supplies', 'Groceries']
['Cleaning Supplies', 'Furniture', 'Toys', 'Gardening Supplies', 'Bedding']
['Gardening Supplies', 'Beauty Products', 'Bedding', 'Furniture', 'Stationery']
['Toys', 'Bedding']
['Furniture', 'Toys']
['Electronics', 'Bedding', 'Groceries', 'Gardening Supplies', 'Stationery']
['Toys', 'Clothes', 'Groceries']
['Toys', 'Electronics']
['Furniture', 'Gardening Supplies', 'Clothes']
['Clothes', 'Groceries', 'Cleaning Supplies', 'Bedding', 'Beauty Products']

Generated Association Rules:
             antecedents          consequents   support  confidence
0      (Beauty Products)            (Bedding)  0.112944    0.328413
1              (Bedding)    (Beauty Products)  0.112944    0.328413
2      (Beauty Products)  (Cleaning Supplies)  0.105330    0.306273
3    (Cleaning Supplies)    (Beauty Products)  0.105330    0.300725
4      (Beauty Products)        (Electronics)  0.116751    0.339483
..                   ...                  ...       ...         ...
61  (Gardening Supplies)         (Stationery)  0.120558    0.358491
62           (Groceries)         (Stationery)  0.110406    0.332061
63           (Groceries)               (Toys)  0.104061    0.312977
64                (Toys)          (Groceries)  0.104061    0.319066
65                (Toys)         (Stationery)  0.110406    0.338521

[66 rows x 4 columns]

Total Execution Time: 0.03 seconds
```

```
Choose dataset(s) for analysis:
1 - Target
2 - Costco
3 - 7-Eleven
4 - ShopRite
5 - K-Mart
Enter dataset numbers (space-separated, e.g., 1 3): 1 3
Enter minimum support (e.g., 0.05): 0.05
Enter minimum confidence (e.g., 0.5): 0.3

Frequent Itemsets from target_data_new.csv:
    support                                itemsets
0     0.309                           (Electronics)
1     0.292                        (Beauty Products)
2     0.291                               (Bedding)
3     0.287                     (Gardening Supplies)
4     0.295                      (Cleaning Supplies)
5     0.286                               (Groceries)
6     0.329                               (Furniture)
7     0.287                                 (Clothes)
8     0.320                              (Stationery)
9     0.278                                   (Toys)
10    0.098                 (Furniture, Electronics)
11    0.099               (Stationery, Electronics)
12    0.092          (Beauty Products, Electronics)
13    0.094            (Furniture, Beauty Products)
14    0.097           (Stationery, Beauty Products)
15    0.083    (Beauty Products, Cleaning Supplies)
16    0.092                 (Electronics, Bedding)
17    0.089            (Beauty Products, Bedding)
18    0.081           (Cleaning Supplies, Bedding)
19    0.090                 (Furniture, Bedding)
20    0.097                (Stationery, Bedding)
21    0.087       (Electronics, Gardening Supplies)
22    0.085           (Gardening Supplies, Bedding)
23    0.082   (Beauty Products, Gardening Supplies)
24    0.089  (Gardening Supplies, Cleaning Supplies)
25    0.097          (Furniture, Gardening Supplies)
26    0.095         (Stationery, Gardening Supplies)
27    0.101           (Furniture, Cleaning Supplies)
28    0.097         (Electronics, Cleaning Supplies)
29    0.097          (Stationery, Cleaning Supplies)
30    0.091           (Groceries, Cleaning Supplies)
31    0.083          (Groceries, Gardening Supplies)
32    0.083                 (Groceries, Bedding)
33    0.100               (Furniture, Groceries)
34    0.097              (Groceries, Electronics)
```

```
29    0.097          (Stationery, Cleaning Supplies)
30    0.091           (Groceries, Cleaning Supplies)
31    0.083          (Groceries, Gardening Supplies)
32    0.083                 (Groceries, Bedding)
33    0.100               (Furniture, Groceries)
34    0.097              (Groceries, Electronics)
35    0.073                 (Groceries, Clothes)
36    0.087              (Stationery, Groceries)
37    0.079           (Groceries, Beauty Products)
38    0.089                 (Furniture, Clothes)
39    0.087           (Clothes, Cleaning Supplies)
40    0.081                (Clothes, Electronics)
41    0.080                   (Clothes, Bedding)
42    0.093                (Stationery, Clothes)
43    0.089          (Clothes, Gardening Supplies)
44    0.077             (Beauty Products, Clothes)
45    0.101             (Stationery, Furniture)
46    0.087                  (Stationery, Toys)
47    0.077                     (Toys, Clothes)
48    0.071             (Toys, Gardening Supplies)
49    0.074              (Toys, Cleaning Supplies)
50    0.091                    (Toys, Electronics)
51    0.073                       (Toys, Bedding)
52    0.077             (Toys, Beauty Products)
53    0.082                  (Groceries, Toys)
54    0.082                  (Furniture, Toys)

Generated Association Rules:
            antecedents           consequents  support  confidence
0          (Electronics)           (Furniture)    0.098    0.317152
1           (Stationery)         (Electronics)    0.099    0.309375
2          (Electronics)           (Stationery)    0.099    0.320388
3     (Beauty Products)         (Electronics)    0.092    0.315068
4     (Beauty Products)           (Furniture)    0.094    0.321918
5           (Stationery)     (Beauty Products)    0.097    0.303125
6     (Beauty Products)          (Stationery)    0.097    0.332192
7               (Bedding)         (Electronics)    0.092    0.316151
8     (Beauty Products)              (Bedding)    0.089    0.304795
9               (Bedding)     (Beauty Products)    0.089    0.305842
10              (Bedding)           (Furniture)    0.090    0.309278
11          (Stationery)              (Bedding)    0.097    0.303125
12              (Bedding)          (Stationery)    0.097    0.333333
13  (Gardening Supplies)         (Electronics)    0.087    0.303136
14  (Gardening Supplies)   (Cleaning Supplies)    0.089    0.310105
15   (Cleaning Supplies)  (Gardening Supplies)    0.089    0.301695
16  (Gardening Supplies)           (Furniture)    0.097    0.337979
17  (Gardening Supplies)          (Stationery)    0.095    0.331010
```

Frequent Itemsets from seven_eleven_data.csv:

	support	itemsets
0	0.303	(Gum)
1	0.297	(Batteries)
2	0.286	(Cigarettes)
3	0.284	(Energy Drinks)
4	0.271	(Snacks)
5	0.318	(Magazine)
6	0.321	(Candy)
7	0.314	(Milk)
8	0.280	(Toiletries)
9	0.274	(Soft Drinks)
10	0.095	(Gum, Candy)
11	0.093	(Gum, Milk)
12	0.109	(Magazine, Gum)
13	0.093	(Gum, Batteries)
14	0.108	(Candy, Batteries)
15	0.100	(Batteries, Milk)
16	0.091	(Magazine, Batteries)
17	0.093	(Cigarettes, Batteries)
18	0.088	(Gum, Cigarettes)
19	0.085	(Cigarettes, Candy)
20	0.092	(Cigarettes, Milk)
21	0.088	(Magazine, Cigarettes)
22	0.099	(Energy Drinks, Gum)
23	0.095	(Energy Drinks, Cigarettes)
24	0.064	(Energy Drinks, Batteries)
25	0.089	(Energy Drinks, Candy)
26	0.077	(Magazine, Energy Drinks)
27	0.082	(Energy Drinks, Milk)
28	0.082	(Gum, Snacks)
29	0.075	(Energy Drinks, Snacks)
30	0.071	(Cigarettes, Snacks)
31	0.070	(Batteries, Snacks)
32	0.075	(Magazine, Snacks)
33	0.084	(Candy, Snacks)
34	0.068	(Toiletries, Snacks)
35	0.081	(Milk, Snacks)
36	0.078	(Snacks, Soft Drinks)
37	0.099	(Magazine, Candy)
38	0.101	(Candy, Milk)
39	0.098	(Magazine, Milk)
40	0.101	(Toiletries, Milk)
41	0.085	(Gum, Toiletries)
42	0.079	(Toiletries, Batteries)
43	0.099	(Toiletries, Candy)

Generated Association Rules:

	antecedents	consequents	support	confidence
0	(Gum)	(Candy)	0.095	0.313531
1	(Gum)	(Milk)	0.093	0.306931
2	(Magazine)	(Gum)	0.109	0.342767
3	(Gum)	(Magazine)	0.109	0.359736
4	(Gum)	(Batteries)	0.093	0.306931
5	(Batteries)	(Gum)	0.093	0.313131
6	(Candy)	(Batteries)	0.108	0.336449
7	(Batteries)	(Candy)	0.108	0.363636
8	(Batteries)	(Milk)	0.100	0.336700
9	(Milk)	(Batteries)	0.100	0.318471
10	(Batteries)	(Magazine)	0.091	0.306397
11	(Cigarettes)	(Batteries)	0.093	0.325175
12	(Batteries)	(Cigarettes)	0.093	0.313131
13	(Cigarettes)	(Gum)	0.088	0.307692
14	(Cigarettes)	(Milk)	0.092	0.321678
15	(Cigarettes)	(Magazine)	0.088	0.307692
16	(Energy Drinks)	(Gum)	0.099	0.348592
17	(Gum)	(Energy Drinks)	0.099	0.326733
18	(Energy Drinks)	(Cigarettes)	0.095	0.334507
19	(Cigarettes)	(Energy Drinks)	0.095	0.332168
20	(Energy Drinks)	(Candy)	0.089	0.313380
21	(Snacks)	(Gum)	0.082	0.302583
22	(Snacks)	(Candy)	0.084	0.309963
23	(Magazine)	(Candy)	0.099	0.311321
24	(Candy)	(Magazine)	0.099	0.308411
25	(Candy)	(Milk)	0.101	0.314642
26	(Milk)	(Candy)	0.101	0.321656
27	(Magazine)	(Milk)	0.098	0.308176
28	(Milk)	(Magazine)	0.098	0.312102
29	(Toiletries)	(Milk)	0.101	0.360714
30	(Milk)	(Toiletries)	0.101	0.321656
31	(Toiletries)	(Gum)	0.085	0.303571
32	(Toiletries)	(Candy)	0.099	0.353571
33	(Candy)	(Toiletries)	0.099	0.308411
34	(Energy Drinks)	(Toiletries)	0.086	0.302817
35	(Toiletries)	(Energy Drinks)	0.086	0.307143
36	(Toiletries)	(Magazine)	0.085	0.303571
37	(Soft Drinks)	(Candy)	0.090	0.328467
38	(Soft Drinks)	(Milk)	0.084	0.306569
39	(Soft Drinks)	(Magazine)	0.086	0.313869

Total Execution Time: 0.02 seconds

GitHub link:

https://github.com/AnuveerRayudu/Anuveer_DataMining