DS634 MID TERM PROJECT REPORT

GHIYAS NIZAMUDDEN SHAIK

INSTRUCTOR: YASSER ABDUALLAH

Table of Contents

Introduction

This project explores Apriori Algorithm, a data mining technique, to identify associations within retail transaction data. The algorithm was implemented using various data mining concepts, principles, and methods and tested for effectiveness and efficiency. I have designed and developed a custom model for mining valuable insights from the data.

Apriori Algorithm

Data Mining is a process of discovering hidden patterns, trends, and correlations from large datasets. This project focuses on Apriori Algorithm, a popular technique used for association rule mining. The key data mining concepts and principles used in this project are highlighted below.

Core Concepts and Principles

Frequent Itemset

The primary goal of the Apriori algorithm is to find the set of items that are frequently bought together. The items provide information about customer purchase patterns and preferences.

Support and Confidence

The two key metrics of this algorithm are support and confidence. Support is the measure of how frequently an item, or a combination of items appear in the dataset, whereas confidence measures the likelihood of two itemsets being purchased together.

Association Rules

Finding strong association rules helps us understand the patterns in customer transactions, particularly the relationship between items. This information helps businesses to make informed decisions for optimizing sale strategies and creating targeted promotions.

Project Workflow

1. Prompt the user for the database, minimum support, and minimum confidence thresholds
2. Load the transaction dataset from the CSV file
3. Preprocessing the dataset and extracting a list of unique itemsets
4. Iteratively generate all possible combinations of itemsets of sizes = 1,2,3, and so on
5. Calculate support for each itemset and update frequent itemsets based on the threshold
6. Create association rules that satisfy the confidence requirements from frequent itemsets

Results and Evaluation

The project’s effectiveness and efficiency are assessed using performance metrics like support, confidence, and the resulting association rules. We also compare the custom Apriori Algorithm implementation with the Apriori library to measure its reliability.

Conclusion

In conclusion, this project successfully implements the Apriori Algorithm to uncover meaningful associations within the data. By applying key data mining concepts, we demonstrated how association rule mining can reveal valuable insights into customer spending habits. The results highlight the significance of data mining for making optimized decision-making in the retail industry.

Program Requirements

Required Software and Packages

Please install the following packages in the terminal using the following commands.

```
python 3.11.7
pandas 2.2.3–pip install pandas
mlxtend 0.23.4–pip install mlxtend
```

Running Instructions

Please follow the steps to run the program

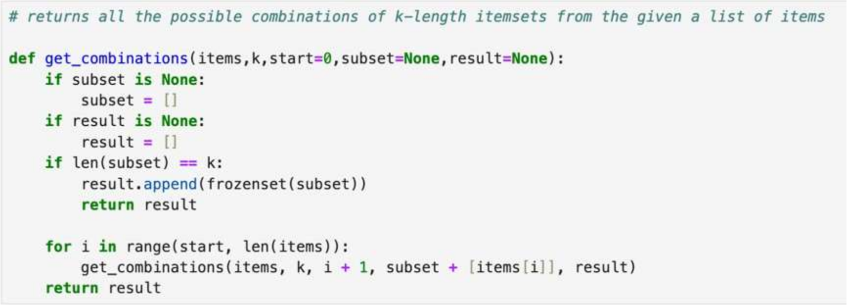
1. Navigate to the directory where the code is located.
2. Create a folder named **`data`** within the same directory.
3. Place all the provided datasets into the **`data`** folder.
4. Open the terminal and navigate to the directory containing the code.
5. Run the python file using the command

```
python ds634-midsem.py
```

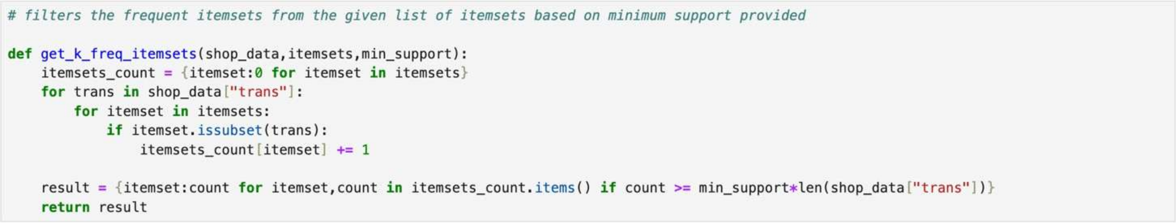
After executing the command, please follow the instructions provided by the program. Provide the store number (1-5), minimum support (1-100), and minimum confidence (1-100).

Code

This section showcases key parts of the code, especially the brute force implementation.



```python
# returns all the possible combinations of k-length itemsets from the given a list of items

def get_combinations(items,k,start=0,subset=None,result=None):
    if subset is None:
        subset = []
    if result is None:
        result = []
    if len(subset) == k:
        result.append(frozenset(subset))
        return result

    for i in range(start, len(items)):
        get_combinations(items, k, i + 1, subset + [items[i]], result)
    return result
```



```python
# filters the frequent itemsets from the given list of itemsets based on minimum support provided

def get_k_freq_itemsets(shop_data,itemsets,min_support):
    itemsets_count = {itemset:0 for itemset in itemsets}
    for trans in shop_data["trans"]:
        for itemset in itemsets:
            if itemset.issubset(trans):
                itemsets_count[itemset] += 1

    result = {itemset:count for itemset,count in itemsets_count.items() if count >= min_support*len(shop_data["trans"])}
    return result
```

```
# iterative approach to generate k-length itemsets and update the frequent itemsets list until there when are no more frequent sets
# being generated from the provide possible itemsets

def get_freq_itemsets(shop_data,min_support):
    freq_itemsets = {}
    k = 1

    while True:
        # generates all possible k-itemsets
        k_itemsets = [subset for subset in get_combinations(shop_data["items"],k)]
        k_itemsets = list(frozenset(k_itemsets))
        print(f"{k}-itemsets generated: {len(k_itemsets)}")

        # checks to see which itemsets are frequent and stops if an empty list is provided
        freq_k_itemsets = get_k_freq_itemsets(shop_data,k_itemsets,min_support)
        if not freq_k_itemsets:  # stopping condition
            break

        freq_itemsets.update(freq_k_itemsets)

        prev_itemsets = frozenset(freq_k_itemsets.keys())
        k += 1

    return freq_itemsets
```

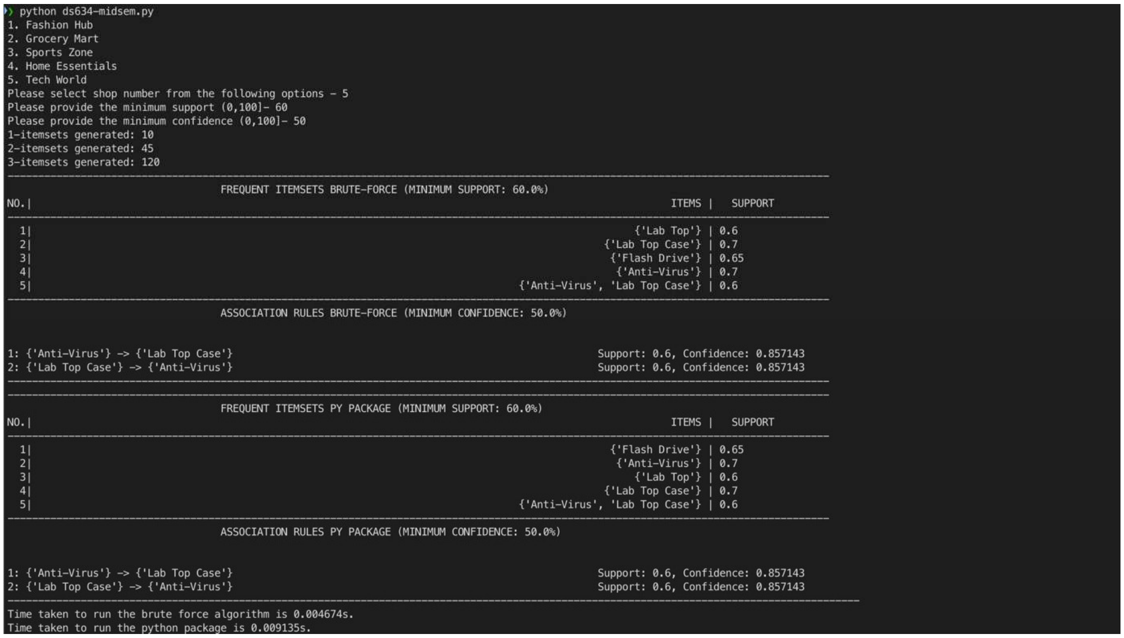
```
# generate association rules from the given frequent itemsets based on the minimum confidence provided

def get_rules(freq_itemsets,shop_data,min_conf):
    assoc_rules = {}

    for itemset,support in freq_itemsets.items():
        if len(itemset) <= 1:
            continue
        subsets = [subset for k in range(1,len(itemset)) for subset in get_combinations(list(itemset),k)]
        for lhs in subsets:
            rhs = itemset - lhs
            lhs_support = freq_itemsets[lhs]
            conf = support/lhs_support
            if conf >= min_conf:
                assoc_rules[(lhs,rhs)] = (support,conf)

    return assoc_rules
```

Program Testing

Below are test inputs used for program testing along with the corresponding result screenshots

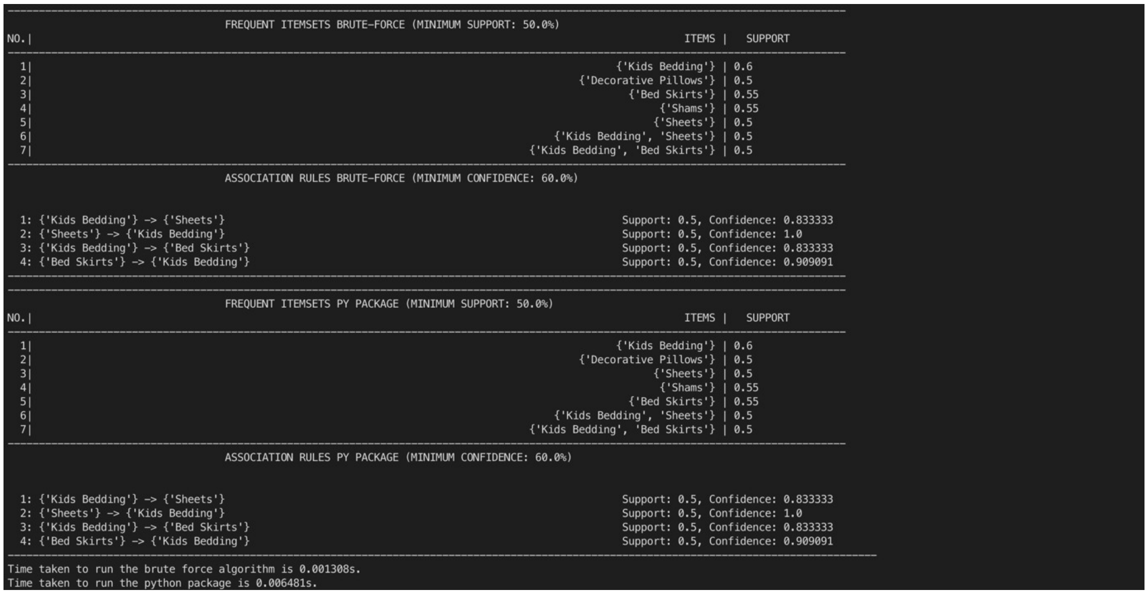
Input format: `(store no., minimum confidence, minimum support)`

1. `(5,60,50)`



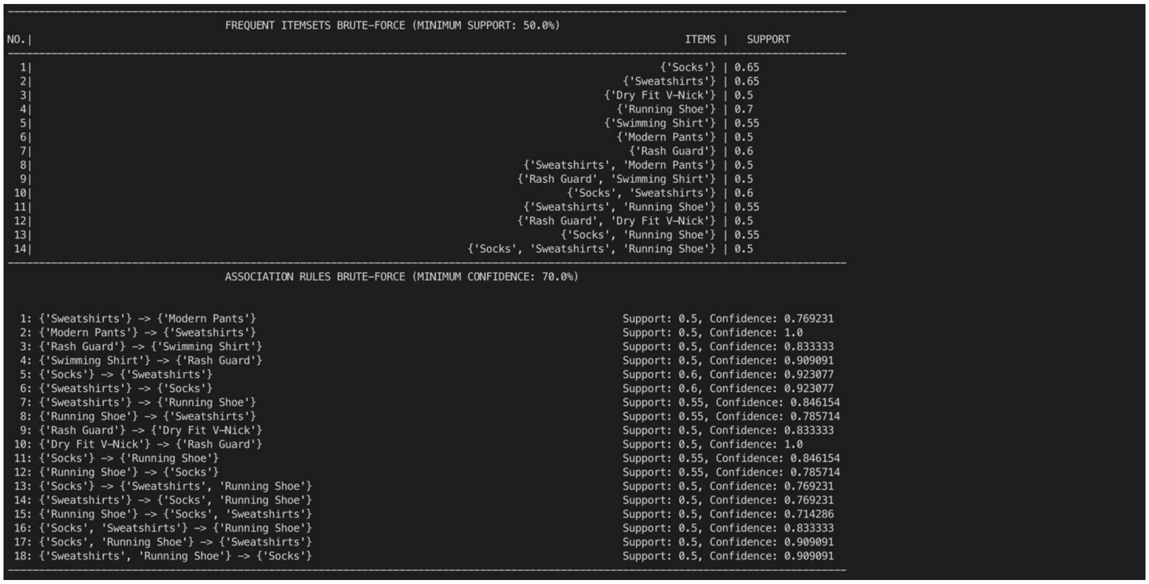
```
❯ python ds634-midsem.py
1. Fashion Hub
2. Grocery Mart
3. Sports Zone
4. Home Essentials
5. Tech World
Please select shop number from the following options - 5
Please provide the minimum support (0,100]- 60
Please provide the minimum confidence (0,100]- 50
1-itemsets generated: 10
2-itemsets generated: 45
3-itemsets generated: 120
---------------------------------------------------------------------------------------------------------------------------------------
                                   FREQUENT ITEMSETS BRUTE-FORCE (MINIMUM SUPPORT: 60.0%)
NO.|                                                                                                         ITEMS |   SUPPORT
---------------------------------------------------------------------------------------------------------------------------------------
  1|                                                                                                   {'Lab Top'} | 0.6
  2|                                                                                              {'Lab Top Case'} | 0.7
  3|                                                                                               {'Flash Drive'} | 0.65
  4|                                                                                                {'Anti-Virus'} | 0.7
  5|                                                                                {'Anti-Virus', 'Lab Top Case'} | 0.6
---------------------------------------------------------------------------------------------------------------------------------------
                                   ASSOCIATION RULES BRUTE-FORCE (MINIMUM CONFIDENCE: 50.0%)


1: {'Anti-Virus'} -> {'Lab Top Case'}                                                            Support: 0.6, Confidence: 0.857143
2: {'Lab Top Case'} -> {'Anti-Virus'}                                                            Support: 0.6, Confidence: 0.857143
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
                                   FREQUENT ITEMSETS PY PACKAGE (MINIMUM SUPPORT: 60.0%)
NO.|                                                                                                         ITEMS |   SUPPORT
---------------------------------------------------------------------------------------------------------------------------------------
  1|                                                                                               {'Flash Drive'} | 0.65
  2|                                                                                                {'Anti-Virus'} | 0.7
  3|                                                                                                   {'Lab Top'} | 0.6
  4|                                                                                              {'Lab Top Case'} | 0.7
  5|                                                                                {'Anti-Virus', 'Lab Top Case'} | 0.6
---------------------------------------------------------------------------------------------------------------------------------------
                                   ASSOCIATION RULES PY PACKAGE (MINIMUM CONFIDENCE: 50.0%)


1: {'Anti-Virus'} -> {'Lab Top Case'}                                                            Support: 0.6, Confidence: 0.857143
2: {'Lab Top Case'} -> {'Anti-Virus'}                                                            Support: 0.6, Confidence: 0.857143
--------------------------------------------------------------------------------------------------------------------------------------------
Time taken to run the brute force algorithm is 0.004674s.
Time taken to run the python package is 0.009135s.
```

```
2. (4,50,60)
```



```
----------------------------------------------------------------------------------------------------
                                        FREQUENT ITEMSETS BRUTE-FORCE (MINIMUM SUPPORT: 50.0%)
NO.|                                                                           ITEMS |   SUPPORT
----------------------------------------------------------------------------------------------------
  1|                                                            {'Kids Bedding'} | 0.6
  2|                                                      {'Decorative Pillows'} | 0.5
  3|                                                              {'Bed Skirts'} | 0.55
  4|                                                                   {'Shams'} | 0.55
  5|                                                                  {'Sheets'} | 0.5
  6|                                                  {'Kids Bedding', 'Sheets'} | 0.5
  7|                                              {'Kids Bedding', 'Bed Skirts'} | 0.5
----------------------------------------------------------------------------------------------------
                                        ASSOCIATION RULES BRUTE-FORCE (MINIMUM CONFIDENCE: 60.0%)


  1: {'Kids Bedding'} -> {'Sheets'}                                                 Support: 0.5, Confidence: 0.833333
  2: {'Sheets'} -> {'Kids Bedding'}                                                 Support: 0.5, Confidence: 1.0
  3: {'Kids Bedding'} -> {'Bed Skirts'}                                             Support: 0.5, Confidence: 0.833333
  4: {'Bed Skirts'} -> {'Kids Bedding'}                                             Support: 0.5, Confidence: 0.909091
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
                                        FREQUENT ITEMSETS PY PACKAGE (MINIMUM SUPPORT: 50.0%)
NO.|                                                                           ITEMS |   SUPPORT
----------------------------------------------------------------------------------------------------
  1|                                                            {'Kids Bedding'} | 0.6
  2|                                                      {'Decorative Pillows'} | 0.5
  3|                                                                  {'Sheets'} | 0.5
  4|                                                                   {'Shams'} | 0.55
  5|                                                              {'Bed Skirts'} | 0.55
  6|                                                  {'Kids Bedding', 'Sheets'} | 0.5
  7|                                              {'Kids Bedding', 'Bed Skirts'} | 0.5
----------------------------------------------------------------------------------------------------
                                        ASSOCIATION RULES PY PACKAGE (MINIMUM CONFIDENCE: 60.0%)


  1: {'Kids Bedding'} -> {'Sheets'}                                                 Support: 0.5, Confidence: 0.833333
  2: {'Sheets'} -> {'Kids Bedding'}                                                 Support: 0.5, Confidence: 1.0
  3: {'Kids Bedding'} -> {'Bed Skirts'}                                             Support: 0.5, Confidence: 0.833333
  4: {'Bed Skirts'} -> {'Kids Bedding'}                                             Support: 0.5, Confidence: 0.909091
--------------------------------------------------------------------------------------------------------
Time taken to run the brute force algorithm is 0.001308s.
Time taken to run the python package is 0.006481s.
```

```
3. (3,50,70)
```



```
----------------------------------------------------------------------------------------------------
                                        FREQUENT ITEMSETS BRUTE-FORCE (MINIMUM SUPPORT: 50.0%)
NO.|                                                                           ITEMS |   SUPPORT
----------------------------------------------------------------------------------------------------
  1|                                                                   {'Socks'} | 0.65
  2|                                                             {'Sweatshirts'} | 0.65
  3|                                                          {'Dry Fit V-Nick'} | 0.5
  4|                                                            {'Running Shoe'} | 0.7
  5|                                                          {'Swimming Shirt'} | 0.55
  6|                                                             {'Modern Pants'} | 0.5
  7|                                                              {'Rash Guard'} | 0.6
  8|                                             {'Sweatshirts', 'Modern Pants'} | 0.5
  9|                                            {'Rash Guard', 'Swimming Shirt'} | 0.5
 10|                                                    {'Socks', 'Sweatshirts'} | 0.6
 11|                                             {'Sweatshirts', 'Running Shoe'} | 0.55
 12|                                            {'Rash Guard', 'Dry Fit V-Nick'} | 0.5
 13|                                                   {'Socks', 'Running Shoe'} | 0.55
 14|                                    {'Socks', 'Sweatshirts', 'Running Shoe'} | 0.5
----------------------------------------------------------------------------------------------------
                                        ASSOCIATION RULES BRUTE-FORCE (MINIMUM CONFIDENCE: 70.0%)


  1: {'Sweatshirts'} -> {'Modern Pants'}                                             Support: 0.5, Confidence: 0.769231
  2: {'Modern Pants'} -> {'Sweatshirts'}                                             Support: 0.5, Confidence: 1.0
  3: {'Rash Guard'} -> {'Swimming Shirt'}                                            Support: 0.5, Confidence: 0.833333
  4: {'Swimming Shirt'} -> {'Rash Guard'}                                            Support: 0.5, Confidence: 0.909091
  5: {'Socks'} -> {'Sweatshirts'}                                                    Support: 0.6, Confidence: 0.923077
  6: {'Sweatshirts'} -> {'Socks'}                                                    Support: 0.6, Confidence: 0.923077
  7: {'Sweatshirts'} -> {'Running Shoe'}                                             Support: 0.55, Confidence: 0.846154
  8: {'Running Shoe'} -> {'Sweatshirts'}                                             Support: 0.55, Confidence: 0.785714
  9: {'Rash Guard'} -> {'Dry Fit V-Nick'}                                            Support: 0.5, Confidence: 0.833333
 10: {'Dry Fit V-Nick'} -> {'Rash Guard'}                                            Support: 0.5, Confidence: 1.0
 11: {'Socks'} -> {'Running Shoe'}                                                   Support: 0.55, Confidence: 0.846154
 12: {'Running Shoe'} -> {'Socks'}                                                   Support: 0.55, Confidence: 0.785714
 13: {'Socks'} -> {'Sweatshirts', 'Running Shoe'}                                    Support: 0.5, Confidence: 0.769231
 14: {'Sweatshirts'} -> {'Socks', 'Running Shoe'}                                    Support: 0.5, Confidence: 0.769231
 15: {'Running Shoe'} -> {'Socks', 'Sweatshirts'}                                    Support: 0.5, Confidence: 0.714286
 16: {'Socks', 'Sweatshirts'} -> {'Running Shoe'}                                    Support: 0.5, Confidence: 0.833333
 17: {'Socks', 'Running Shoe'} -> {'Sweatshirts'}                                    Support: 0.5, Confidence: 0.909091
 18: {'Sweatshirts', 'Running Shoe'} -> {'Socks'}                                    Support: 0.5, Confidence: 0.909091
----------------------------------------------------------------------------------------------------
```

References

The source code and datasets used for this project are available in the following Git repository:

https://github.com/Ghiyas155/Shaik_GhiyasNizamudden_Midtermproject