A

Assesment Report

on

“Market Basket Analysis”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25 in

CSE(AIML)

By

Sudhakar Kumar (202401100400190)

Under the supervision of

Abhishek Shukla

KIET Group of Institutions, Ghaziabad

April, 2025

Introduction

Market Basket Analysis (MBA) is a technique used in data mining to uncover associations and patterns among a set of items in transactional data. In retail, this is often used to understand customer purchasing ， by finding out which products tend to be bought together. This report presents an implementation of the algorithm i, a popular algorithm for mining frequent datasets and discovering association rules in transactional data.

The main objective of this analysis is to simulate a retail dataset, apply the algorithm to identify frequent datasets, and generate association rules based on the aisle data. These rules can help identify relationships between products, such as which products are often bought together.

Methodology

Step 1: Data Collection

The first step involves uploading a CSV file that contains data on different products organized by aisles. The dataset is loaded using Python's pandas library. Each row of this dataset represents a product in a particular aisle of the store.

Step 2: Data Preparation

Once the dataset is loaded, the aisle column is extracted into a list. This represents the different aisles or categories of products. The data is then used to simulate customer transactions, where each transaction is a random sample of aisles. A total of 100 simulated transactions are generated, with each transaction containing between 1 and 5 aisles.

Step 3: Encoding the Transactions

For the algorithm to work, the transactions need to be encoded into a binary matrix, where each column corresponds to an aisle, and each row represents a transaction. The value in a cell is 1 if the aisle is part of the transaction and 0 otherwise. This is accomplished using the Transaction Encoder class from the .preprocessing module.

Step 4: Running the Algorithm

The algorithm is applied to the encoded data to find frequent datasets. Frequent datasets are sets of aisles that appear together in a minimum percentage of transactions, known as the **support** threshold. In this case, the minimum support threshold is set to 3%, meaning that an itemset must appear in at least 3% of all transactions to be considered frequent.

Step 5: Generating Association Rules

Once frequent datasets are identified, the next step is to generate association rules. Association rules express relationships between different items, such as "if item A is purchased, then item B is likely to be purchased." These rules are evaluated using the following metrics:

- **Support**: The fraction of transactions that contain the itemset.
- **Confidence**: The probability that the consequent of the rule appears in a transaction, given that the antecedent is already present.
- **Lift**: A measure of how much more likely the consequent is to appear with the antecedent, compared to random chance.

The association _rules function from the frequent_ patterns module is used to generate rules with a minimum **lift** of 1.0, which ensures that the rules are statistically meaningful.

Step 6: Output

If frequent datasets are found, the results are printed, showing both the frequent datasets and the association rules. The rules display the antecedents (left-hand side items), consequents (right-hand side items), support, confidence, and lift values.

Code

```
# Step 1: Upload CSV file in Colab
from google.colab import files
uploaded = files.upload()

# Step 2: Load uploaded file (auto-detect name)
import pandas as pd
import random
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules
import io

# Get the uploaded filename
filename = next(iter(uploaded))
df_aisles = pd.read_csv(io.BytesIO(uploaded[filename]))

# Step 3: Convert aisle column to a list
aisle_list = df_aisles['aisle'].tolist()

# Step 4: Simulate transactions
num_transactions = 100
transactions = [
    random.sample(aisle_list, random.randint(1, 5))
    for _ in range(num_transactions)
]

# Step 5: Encode transactions
te = TransactionEncoder()
te_data = te.fit(transactions).transform(transactions)
df_encoded = pd.DataFrame(te_data, columns=te.columns_)

# Step 6: Run Apriori
frequent_itemsets = apriori(df_encoded, min_support=0.03, use_colnames=True)
```

```
# Step 7: Association rules
if not frequent_itemsets.empty:
    rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1.0)
    print("\n📦 Frequent Itemsets:")
    print(frequent_itemsets)

    print("\n🔗 Association Rules:")
    print(rules[['antecedents', 'consequents', 'support', 'confidence', 'lift']])
else:
    print("⚠️ No frequent itemsets found. Try lowering the min_support value.")
```

Output

Choose Files 10. Market …Analysis.csv

- **10. Market Basket Analysis.csv**(text/csv) - 2603 bytes, last modified: 4/18/2025 - 100% done

```
Saving 10. Market Basket Analysis.csv to 10. Market Basket Analysis (1).csv
```

📦 Frequent Itemsets:

	support	itemsets
0	0.04	(air fresheners candles)
1	0.05	(baking supplies decor)
2	0.03	(beauty)
3	0.05	(beers coolers)
4	0.04	(breakfast bakery)
5	0.04	(breakfast bars pastries)
6	0.04	(bulk dried fruits vegetables)
7	0.03	(bulk grains rice dried goods)
8	0.04	(buns rolls)
9	0.03	(butter)
10	0.03	(candy chocolate)
11	0.03	(canned fruit applesauce)
12	0.03	(canned jarred vegetables)
13	0.03	(canned meat seafood)
14	0.03	(cereal)
15	0.03	(chips pretzels)
16	0.03	(cookies cakes)
17	0.04	(cream)
18	0.04	(dish detergents)
19	0.04	(dog food care)
20	0.03	(eggs)
21	0.04	(energy sports drinks)

22	0.03	(feminine care)
23	0.04	(food storage)
24	0.04	(fresh pasta)
25	0.05	(frozen breads doughs)
26	0.03	(frozen pizza)
27	0.04	(frozen vegan vegetarian)
28	0.07	(fruit vegetable snacks)
29	0.07	(grains rice dried goods)
30	0.04	(granola)
31	0.03	(ice cream toppings)
32	0.05	(indian foods)
33	0.03	(kosher foods)
34	0.04	(laundry)
35	0.03	(lunch meat)
36	0.04	(meat counter)
37	0.03	(milk)
38	0.04	(oils vinegars)
39	0.04	(other)
40	0.05	(packaged meat)
41	0.03	(packaged produce)
42	0.06	(packaged vegetables fruits)
43	0.03	(pasta sauce)
44	0.05	(popcorn jerky)
45	0.06	(prepared meals)
46	0.04	(preserved dips spreads)
47	0.04	(protein meal replacements)
48	0.04	(red wines)
49	0.04	(refrigerated)
50	0.04	(salad dressing toppings)
51	0.04	(spices seasonings)
52	0.03	(spirits)
53	0.03	(tea)
54	0.03	(tofu meat alternatives)
55	0.03	(water seltzer sparkling water)
56	0.03	(yogurt)

🔗 Association Rules:

Empty DataFrame

Columns: [antecedents, consequents, support, confidence, lift]

Index: []

References

1. Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, 487-499.
2. MLxtend Documentation. (2025). *mlxtend: Machine Learning Extensions*. Retrieved from https://rasbt.github.io/mlxtend/
3. Pandas Documentation. (2025). *pandas: Python Data Analysis Library*. Retrieved from https://pandas.pydata.org/pandas-docs/stable/

Credits

- **Python Libraries Used:**
  - pandas: For data manipulation and loading CSV files.
  - mlxtend: For performing the Apriori algorithm and generating association rules.
  - random: For generating simulated transaction data.
  - io: For handling the uploaded CSV file.