****

**A**

**Assesment Report**

on

**“Market Basket Analysis”**

submitted as partial fulfillment for the award of

**BACHELOR OF TECHNOLOGY**

**DEGREE**

SESSION 2024-25

in

**Computer Science Engineering (Artificial Intelligence)**

By

Utkarsh Gupta (202401100300270)

**Under the supervision of**

“Mr. Abhishek Shukla”

**KIET Group of Institutions, Ghaziabad**

Affiliated to

**Dr. A.P.J. Abdul Kalam Technical University, Lucknow**

(Formerly UPTU)

**May, 2025**

**Market Basket Analysis Report**

**Introduction**

Market Basket Analysis (MBA) is a data mining technique used to uncover associations between products purchased by customers. It helps businesses understand purchasing patterns, optimize product placement, and design targeted marketing strategies.

**Problem Statement:**  
Given a dataset of grocery aisles, we aim to identify frequent item associations using Association Rule Mining to generate insights for targeted marketing.

**Key Objectives:**

* Identify frequently co-purchased items
* Generate association rules with metrics (support, confidence, lift)
* Provide actionable marketing strategies

**Methodology:**

We used **Association Rule Mining** with the **Apriori algorithm** to analyze synthetic transaction data. The steps included:

1. **Data Preparation:**
   * Loaded the aisle dataset
   * Generated synthetic transactions with realistic shopping patterns
2. **Frequent Itemset Mining:**
   * Applied the Apriori algorithm to find itemsets with min\_support = 0.03
3. **Rule Generation:**
   * Extracted association rules with min\_lift > 1 and confidence ≥ 0.5
4. **Visualization & Insights:**
   * Created bar plots and scatter plots for rule analysis
   * Derived marketing recommendations

**Code:**

# Market Basket Analysis with Association Rule Mining

# Using already uploaded files in Colab environment

# Import necessary libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import os

from mlxtend.frequent\_patterns import apriori, association\_rules

from mlxtend.preprocessing import TransactionEncoder

# ----------------------------

# STEP 1: Locate Uploaded Files

# ----------------------------

print("="\*50)

print("STEP 1: LOCATING UPLOADED FILES")

print("="\*50)

# List files in /content directory

content\_files = os.listdir("/content")

print("\nFiles available in /content directory:")

for file in content\_files:

    print(f"- {file}")

# Find CSV files

csv\_files = [f for f in content\_files if f.lower().endswith('.csv')]

if not csv\_files:

    raise FileNotFoundError("No CSV files found in /content directory")

# Use the first CSV found (modify if you need a specific file)

file\_name = csv\_files[0]

print(f"\nUsing CSV file: {file\_name}")

# ----------------------------

# STEP 2: Data Loading

# ----------------------------

print("\n" + "="\*50)

print("STEP 2: DATA LOADING")

print("="\*50)

# Load the aisle data

aisles = pd.read\_csv(f"/content/{file\_name}")

print("\nAisle data loaded successfully. First 5 rows:")

print(aisles.head())

# ----------------------------

# STEP 3: Generate Synthetic Transactions

# ----------------------------

print("\n" + "="\*50)

print("STEP 3: GENERATING SYNTHETIC TRANSACTIONS")

print("="\*50)

np.random.seed(42)

num\_transactions = 1000

transactions = []

# Common shopping patterns (based on typical consumer behavior)

common\_patterns = [

    ['fresh fruits', 'fresh vegetables'],

    ['milk', 'eggs', 'bread'],

    ['pasta', 'pasta sauce'],

    ['coffee', 'cream'],

    ['cereal', 'milk'],

    ['chips', 'soda'],

    ['beer', 'snacks'],

    ['diapers', 'baby food formula'],

    ['shampoo', 'conditioner'],

    ['toothpaste', 'toothbrush']

]

print("\nGenerating transactions with realistic shopping patterns...")

for \_ in range(num\_transactions):

    # 50% chance to include a common pattern

    if np.random.random() < 0.5:

        pattern = common\_patterns[np.random.randint(0, len(common\_patterns))]

        remaining\_items = np.random.choice(

            aisles[~aisles['aisle'].isin(pattern)]['aisle'],

            np.random.randint(1, 6),

            replace=False

        )

        transaction = list(pattern) + list(remaining\_items)

    else:

        # Random transaction

        transaction = list(np.random.choice(aisles['aisle'], np.random.randint(3, 8), replace=False))

    transactions.append(transaction)

print(f"\nGenerated {len(transactions)} synthetic transactions with:")

print(f"- {len(common\_patterns)} built-in shopping patterns")

print(f"- Random transactions to simulate variety")

# ----------------------------

# STEP 4: Market Basket Analysis

# ----------------------------

print("\n" + "="\*50)

print("STEP 4: MARKET BASKET ANALYSIS")

print("="\*50)

# Convert transactions to one-hot encoded format

print("\nPreparing data for analysis...")

te = TransactionEncoder()

te\_ary = te.fit(transactions).transform(transactions)

df = pd.DataFrame(te\_ary, columns=te.columns\_)

# Find frequent itemsets

print("\nFinding frequent itemsets with min\_support=0.03...")

frequent\_itemsets = apriori(df, min\_support=0.03, use\_colnames=True)

if not frequent\_itemsets.empty:

    print("\nTop 10 frequent itemsets:")

    print(frequent\_itemsets.sort\_values('support', ascending=False).head(10))

    # Generate association rules

    print("\nGenerating association rules...")

    rules = association\_rules(frequent\_itemsets, metric="lift", min\_threshold=1)

    if not rules.empty:

        # Filter strong rules

        strong\_rules = rules[(rules['confidence'] >= 0.5) & (rules['lift'] > 1)]

        # ----------------------------

        # STEP 5: Visualization

        # ----------------------------

        print("\n" + "="\*50)

        print("STEP 5: VISUALIZATION")

        print("="\*50)

        # Visualization 1: Top 20 rules by lift

        plt.figure(figsize=(12, 8))

        top\_rules = rules.sort\_values('lift', ascending=False).head(20)

        top\_rules['rule'] = top\_rules.apply(

            lambda x: f"{', '.join(list(x['antecedents']))} → {', '.join(list(x['consequents']))}", axis=1)

        sns.barplot(x='lift', y='rule', data=top\_rules, palette='viridis')

        plt.title("Top 20 Association Rules by Lift", fontsize=16)

        plt.xlabel("Lift Score", fontsize=12)

        plt.ylabel("Association Rule", fontsize=12)

        plt.tight\_layout()

        plt.show()

        # Visualization 2: Scatter plot

        plt.figure(figsize=(10, 6))

        scatter = sns.scatterplot(x="support", y="confidence", size="lift",

                                data=rules, hue="lift", palette="viridis")

        plt.title("Association Rules: Support vs Confidence", fontsize=16)

        plt.xlabel("Support", fontsize=12)

        plt.ylabel("Confidence", fontsize=12)

        plt.legend(bbox\_to\_anchor=(1.05, 1), loc='upper left')

        plt.tight\_layout()

        plt.show()

        # ----------------------------

        # STEP 6: Marketing Insights

        # ----------------------------

        print("\n" + "="\*50)

        print("STEP 6: MARKETING INSIGHTS")

        print("="\*50)

        if not strong\_rules.empty:

            print(f"\nFound {len(strong\_rules)} strong association rules:")

            for idx, rule in strong\_rules.sort\_values('lift', ascending=False).iterrows():

                print("\n" + "-"\*50)

                print(f"RULE: When customers buy {list(rule['antecedents'])}, they also buy {list(rule['consequents'])}")

                print("\nMETRICS:")

                print(f"- Support:    {rule['support']:.3f} (Frequency of this combination)")

                print(f"- Confidence: {rule['confidence']:.3f} (Probability of consequent given antecedent)")

                print(f"- Lift:       {rule['lift']:.3f} (Strength of association)")

                print("\nMARKETING RECOMMENDATIONS:")

                print("1. Product Placement: Place these items near each other")

                print("2. Bundling: Create special bundle offers")

                print("3. Promotions: Offer discounts on consequent when antecedent is purchased")

                print("4. Recommendations: Suggest these items together online")

        else:

            print("\nNo strong rules found. Try lowering the confidence threshold or min\_support.")

    else:

        print("\nNo association rules generated. Try lowering the min\_support parameter.")

else:

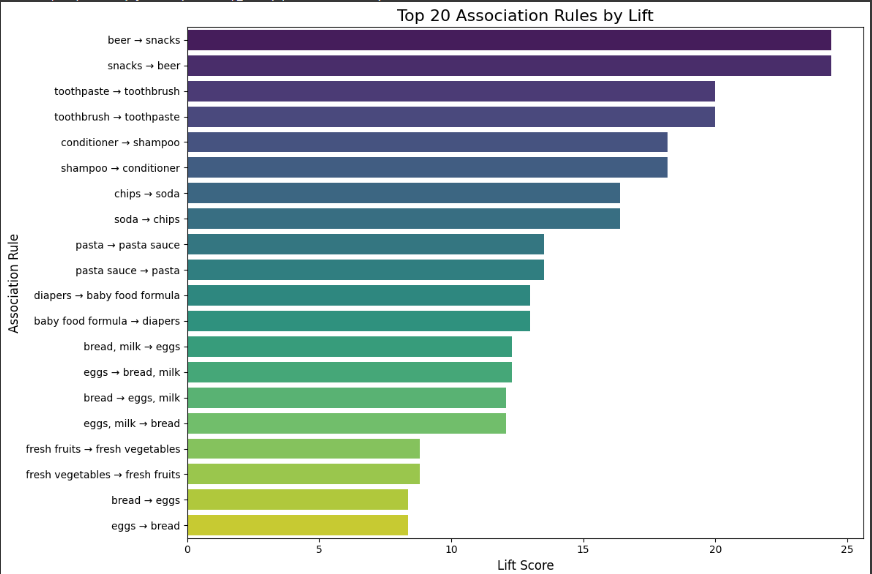
    print("\nNo frequent itemsets found. The min\_support parameter might be too high.")

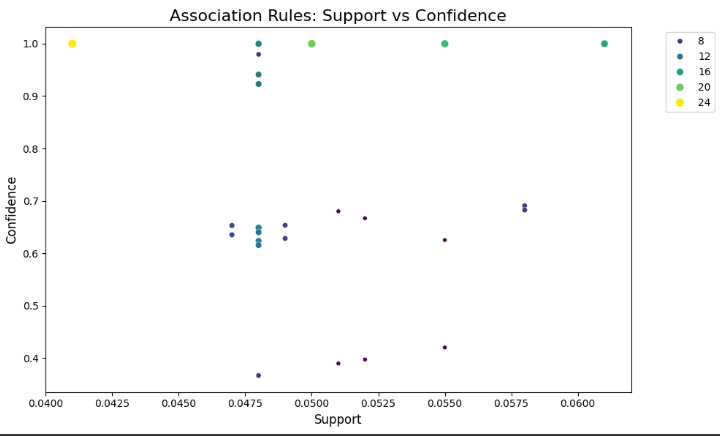
print("\n" + "="\*50)

print("ANALYSIS COMPLETE")

print("="\*50)

Output:





**References/Credits**

* Dataset: Instacart Market Basket Dataset (Kaggle)
* Libraries: mlxtend, pandas, seaborn
* Inspiration: "Introduction to Market Basket Analysis" by Analytics Vidhya