

## NAAN MUDHALVAN PROJECT (IBM)

**IBM AI 101 ARTIFICIAL INTELLIGENCE-GROUP 1**

## Title: Market Basket Analysis

|  |  |
| --- | --- |
| **Team name**: unknown |  |
| **Team members:** Mohammed Hanifa M | 113321106052 |
| Naveen D | 113321106060 |
| Mohammed Tawfiq P | 113321106053 |
| Dharshan G | 113321106022 |
| Agathiyan G | 113321106002 |
| Jebastin D | 113321106039 |

**Problem Statement:**

Unveiling Customer Behaviour through Association Analysis: Utilize market basket analysis on the provided dataset to uncover hidden patterns and associations between products, aiming to understand customer purchasing behaviour and identify potential cross-selling opportunities for the retail business

Challenges:

Large dataset Market Basket Analysis can be computationally demanding when dealing with large datasets.

Efficient algorithms and scalable techniques are required to handle such data.

Sparse data problem When transactional data is sparse, meaning there are many items but few frequent item sets, Market Basket Analysis becomes more challenging.

Techniques like FP-growth algorithm can be utilized to address this problem. Interpretation of results Understanding and interpreting the generated association rules require domain knowledge and careful analysis.

False positives and irrelevant rules can mislead decision-making if not properly evaluated.

# **METHODS FOR MARKET BASKET ANALYSIS:**

### Association rule mining

Association rule mining is a popular technique used in Market Basket Analysis. It uncovers relationships and dependencies between items purchased together, enabling businesses to identify rules such as "If A is purchased, then B is likely to be purchased as well."

### Apriori algorithm

The Apriori algorithm is a widely used method for Market Basket Analysis. It scans the transactional data to discover frequent itemsets and generate association rules. It is efficient in handling large datasets and is highly interpretable.

### FP-growth algorithm

The FP-growth algorithm is an alternative approach to Association Rule Mining. It constructs a compact data structure called the FP-tree, which allows for faster pattern mining. It is particularly useful for handling sparse datasets.

**Data Mining**:

In this method we were concentrating on the customers and the retail shop owners they were bought the product depend upon the area need then the retail shop owners collect the data from the customers what they have bought daily basis in this daily needs are different but some company in the market selling some good products to the people in essential manner so depend upon this data the company which were selling good product increasing the product depend upon the area of the need this market basket analyzis is about we have done on the data mining.

**GOALS:**

1. Association Rule Discovery: Implement the Apriori algorithm to identify frequent item sets and generate meaningful association rules based on support, confidence, and lift.
2. Insightful Interpretation: Interpret the generated association rules to understand the relationships between products and their significance, translating them into actionable business strategies.
3. Business Recommendations: Provide clear and actionable recommendations to businesses based on the derived insights, empowering them to make informed decisions for marketing campaigns, inventory management, and customer experience enhancement.
4. Enhanced Sales and Customer Satisfaction: Ultimately, the goal is to enhance sales by optimizing product offerings and improving customer satisfaction through personalized recommendations and strategic business decisions.

**Design Thinking:**

In this project we have design the application model for this market basket analysis. This application is fully based on user friendly to the customers where they have bought the products on the nearby retail shop

# **The Customer Journey:**

#### Exploration

Customers embark on a journey to explore various products available in the retail store.

#### Selection

After exploring, customers carefully select the items they need for their basket.

#### Purchase

Customers proceed to the checkout and finalize their purchase.

# The Power of Data

#### Insights

By analyzing customer purchase data, we can gain valuable insights into their shopping habits.

#### Trends

Identifying trends helps us anticipate customer needs and tailor our offerings accordingly.

#### Personalization

With data-driven approaches, we can provide personalized recommendations to enhance the customer experience.

# **Streamlined Process**

#### Research

Gather data about customer preferences and shopping patterns.

#### Analyze

Apply advanced algorithms to identify purchasing patterns and trends.

#### Implement

Develop and integrate the market basket analysis application into retail shops.

# **Benefits for Retailers**

#### Optimized Inventory

Market basket analysis allows retailers to optimize their inventory based on customer demand.



#### Increased Sales

By offering personalized recommendations, retailers can increase sales and customer satisfaction.

#### Effective Marketing

With insights into customer preferences, retailers can develop targeted marketing strategies.



# **Finding Success through Data**

#### Customer-Centric Approach

By understanding customer needs, retailers can create personalized experiences and build brand loyalty.

#### Continuous Improvement

Regularly analyzing market basket data allows retailers to adapt and improve their offerings over time.

#### Staying Ahead

Using data-driven insights helps retailers stay ahead of competition and drive innovation in the industry

Market Analysis Model 1:

Market basket analysis is a technique used in retail and e-commerce to analyze customer purchase patterns and understand relationships between products that are frequently bought together. This analysis helps businesses optimize product placement, promotional strategies, and cross-selling efforts. One popular algorithm for market basket analysis is the Apriori algorithm, which generates association rules between items based on their co-occurrence in transactions.

import pandas as pd

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

pd.set\_option('display.max\_columns', None)

pd.set\_option('display.max\_rows', None)

pd.set\_option('display.width', 500)

df = pd.read\_excel("/kaggle/input/market-basket-analysis/Assignment-1\_Data.xlsx")

def outlier\_thresholds(dataframe, variable):

quartile1 = dataframe[variable].quantile(0.01)

quartile3 = dataframe[variable].quantile(0.99)

interquantile\_range = quartile3 - quartile1

up\_limit = quartile3 + 1.5 \* interquantile\_range

low\_limit = quartile1 - 1.5 \* interquantile\_range

return low\_limit, up\_limit

def replace\_with\_thresholds(dataframe, variable):

low\_limit, up\_limit = outlier\_thresholds(dataframe, variable)

dataframe.loc[(dataframe[variable] < low\_limit), variable] = low\_limit

dataframe.loc[(dataframe[variable] > up\_limit), variable] = up\_limit

def retail\_data\_prep(dataframe):

dataframe = dataframe[dataframe["Quantity"] > 0]

dataframe = dataframe[dataframe["Price"] > 0]

replace\_with\_thresholds(dataframe, "Quantity")

replace\_with\_thresholds(dataframe, "Price")

return dataframe

df = retail\_data\_prep(df)

df.describe().T

Out[1]:

|  | count | mean | min | 25% | 50% | 75% | max | std |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Quantity | 519551.0 | 9.39742 | 1.0 | 1.0 | 3.0 | 10.0 | 248.5 | 21.281261 |
| Date | 519551 | 2011-07-04 16:03:31.051080704 | 2010-12-01 08:26:00 | 2011-03-28 10:52:00 | 2011-07-20 11:55:00 | 2011-10-19 15:08:00 | 2011-12-09 12:50:00 | NaN |
| Price | 519551.0 | 3.32647 | 0.001 | 1.25 | 2.08 | 4.13 | 41.94 | 3.87738 |
| CustomerID | 387985.0 | 15317.042994 | 12346.0 | 13950.0 | 15265.0 | 16837.0 | 18287.0 | 1721.813298 |

In [2]:

df\_fr = df[df['Country'] == "France"]

df\_fr.groupby(['BillNo', 'Itemname']).agg({"Quantity": "sum"}).unstack().fillna(0).iloc[0:5, 0:5]

Out[2]:

|  | Quantity | | | | |
| --- | --- | --- | --- | --- | --- |
| Itemname | 10 COLOUR SPACEBOY PEN | 12 COLOURED PARTY BALLOONS | 12 EGG HOUSE PAINTED WOOD | 12 MESSAGE CARDS WITH ENVELOPES | 12 PENCIL SMALL TUBE WOODLAND |
| BillNo |  |  |  |  |  |
| 536370 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 536852 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 536974 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 537065 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 537463 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

In [3]:

fr\_inv\_pro\_df=df\_fr.groupby(['BillNo', 'Itemname']). \

agg({"Quantity": "sum"}). \

unstack(). \

fillna(0). \

applymap(lambda x: 1 if x > 0 else 0)

frequent\_itemsets = apriori(fr\_inv\_pro\_df.astype("bool"),

min\_support=0.01,

use\_colnames=True)

frequent\_itemsets.sort\_values("support", ascending=False).head()

Out[3]:

|  | support | itemsets |
| --- | --- | --- |
| 330 | 0.765306 | ((Quantity, POSTAGE)) |
| 332 | 0.188776 | ((Quantity, RABBIT NIGHT LIGHT)) |
| 371 | 0.181122 | ((Quantity, RED TOADSTOOL LED NIGHT LIGHT)) |
| 320 | 0.170918 | ((Quantity, PLASTERS IN TIN WOODLAND ANIMALS)) |
| 315 | 0.168367 | ((Quantity, PLASTERS IN TIN CIRCUS PARADE)) |

Market Analysis Model 2:

Market basket analysis is a technique used in retail and e-commerce to discover associations and relationships between items that are frequently purchased together. The Apriori algorithm is a popular choice for this type of analysis. In this introduction, I'll provide you with Python code to perform basket analysis using the Apriori algorithm. We'll use the apyori library, which is a simple and lightweight library for Apriori

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from mlxtend.preprocessing import TransactionEncoder

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

from pandas.plotting import parallel\_coordinates

import warnings

warnings.filterwarnings('ignore')

In [2]:

df = pd.read\_csv('../input/datasets-for-appiori/basket\_analysis.csv')

In [3]:

print(df)

Unnamed: 0 Apple Bread Butter Cheese Corn Dill Eggs Ice cream \

0 0 False True False False True True False True

1 1 False False False False False False False False

2 2 True False True False False True False True

3 3 False False True True False True False False

4 4 True True False False False False False False

.. ... ... ... ... ... ... ... ... ...

994 994 False True False False False False True False

995 995 True False False False True False False False

996 996 True False False False True True False False

997 997 False False True True True False True True

998 998 False False False False False False False False

Kidney Beans Milk Nutmeg Onion Sugar Unicorn Yogurt chocolate

0 False False False False True False True True

1 False True False False False False False False

2 False True False False False False True True

3 False True True True False False False False

4 False False False False False False False False

.. ... ... ... ... ... ... ... ...

994 False False False False False True False True

995 True True True False False False True False

996 False False False False True False False True

997 True False True False True False True True

998 False True False False False False False True

[999 rows x 17 columns]

In [4]:

df.drop(df.columns[0],axis=1,inplace=True)

In [5]:

print(df)

Apple Bread Butter Cheese Corn Dill Eggs Ice cream \

0 False True False False True True False True

1 False False False False False False False False

2 True False True False False True False True

3 False False True True False True False False

4 True True False False False False False False

.. ... ... ... ... ... ... ... ...

994 False True False False False False True False

995 True False False False True False False False

996 True False False False True True False False

997 False False True True True False True True

998 False False False False False False False False

Kidney Beans Milk Nutmeg Onion Sugar Unicorn Yogurt chocolate

0 False False False False True False True True

1 False True False False False False False False

2 False True False False False False True True

3 False True True True False False False False

4 False False False False False False False False

.. ... ... ... ... ... ... ... ...

994 False False False False False True False True

995 True True True False False False True False

996 False False False False True False False True

997 True False True False True False True True

998 False True False False False False False True

[999 rows x 16 columns]

In [6]:

df.shape

Out[6]:

(999, 16)

We have 999 basket for us to compute the recommendation for each item that sold in the store. There are 16 items that sold in the shop.

In [7]:

df.mean()

Out[7]:

Apple 0.383383

Bread 0.384384

Butter 0.420420

Cheese 0.404404

Corn 0.407407

Dill 0.398398

Eggs 0.384384

Ice cream 0.410410

Kidney Beans 0.408408

Milk 0.405405

Nutmeg 0.401401

Onion 0.403403

Sugar 0.409409

Unicorn 0.389389

Yogurt 0.420420

chocolate 0.421421

dtype: float64

The sale transaction or count for each unique item approximately for this sample. We will dive into and see whether there is any difference or correlation between the baskets. Since the dataframe is already tabulated one hot data frame, we will straight away and use the dataset to be analyzed with apriori

## Apriori Algorithm:

Little bit background introduction for Apriori Algorithm. The algorithm assumes that any subset of a frequent itemset must be frequent. Say in our cases, where {apple, unicorn, yoghurt} is frequent then {apple,yoghurt} is frequent. Whereas {apple,unicorn} is not frequent, then {apple,unicorn,yoghurt} is not frequent.

**SUPPORT** = A simple way to control complexity is to place a constraint that such rules must apply to some minimum percentage of the data  
**CONFIDENCE** = The probability that B occurs when A; it is p(B|A), which in association mining.  
**LIFT** = the co-occurrence of A and B is the probability that we actually see the two together, compared to the probability that we would see the two together if they were unrelated to (independent of) each other.  
**LEVERAGE** = alternative is to look at the difference between these quantities rather than their ratio.  
**CONVICTION** = measure to ascertain the direction of the rule. Unlike lift, conviction is sensitive to the rule direction.

Just Support and Confidence as a parameter might be misleading for items that are too common/ popular in the basket. It is more likely that popular items are part of the same basket just because they are popular rather than anything else.

We set the mininum support as 0.06, maximum number that being analysed in the basket is 3. We are doing first pruning and see what we get from the result

In [8]:

*# Compute frequent itemsets using the Apriori algorithm*

frequent\_itemsets = apriori(df,

min\_support = .006,

max\_len = 3,

use\_colnames = True)

In [9]:

frequent\_itemsets

Out[9]:

|  | support | itemsets |
| --- | --- | --- |
| 0 | 0.383383 | (Apple) |
| 1 | 0.384384 | (Bread) |
| 2 | 0.420420 | (Butter) |
| 3 | 0.404404 | (Cheese) |
| 4 | 0.407407 | (Corn) |
| ... | ... | ... |
| 691 | 0.098098 | (Onion, Yogurt, chocolate) |
| 692 | 0.087087 | (Yogurt, Sugar, Unicorn) |
| 693 | 0.090090 | (Sugar, chocolate, Unicorn) |
| 694 | 0.095095 | (Yogurt, Sugar, chocolate) |
| 695 | 0.086086 | (Yogurt, chocolate, Unicorn) |

In [10]:

*# Compute all association rules for frequent\_itemsets*

rules = association\_rules(frequent\_itemsets,

metric = 'support',

min\_threshold=0.1)

In [11]:

rules

Out[11]:

|  | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | (Apple) | (Bread) | 0.383383 | 0.384384 | 0.154154 | 0.402089 | 1.046059 | 0.006788 | 1.029610 |
| 1 | (Bread) | (Apple) | 0.384384 | 0.383383 | 0.154154 | 0.401042 | 1.046059 | 0.006788 | 1.029482 |
| 2 | (Butter) | (Apple) | 0.420420 | 0.383383 | 0.188188 | 0.447619 | 1.167549 | 0.027006 | 1.116289 |
| 3 | (Apple) | (Butter) | 0.383383 | 0.420420 | 0.188188 | 0.490862 | 1.167549 | 0.027006 | 1.138354 |
| 4 | (Cheese) | (Apple) | 0.404404 | 0.383383 | 0.162162 | 0.400990 | 1.045925 | 0.007120 | 1.029393 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 433 | (Yogurt, Milk) | (chocolate) | 0.190190 | 0.421421 | 0.104104 | 0.547368 | 1.298862 | 0.023954 | 1.278255 |
| 434 | (chocolate, Milk) | (Yogurt) | 0.211211 | 0.420420 | 0.104104 | 0.492891 | 1.172376 | 0.015307 | 1.142909 |
| 435 | (Yogurt) | (chocolate, Milk) | 0.420420 | 0.211211 | 0.104104 | 0.247619 | 1.172376 | 0.015307 | 1.048390 |
| 436 | (chocolate) | (Yogurt, Milk) | 0.421421 | 0.190190 | 0.104104 | 0.247031 | 1.298862 | 0.023954 | 1.075489 |
| 437 | (Milk) | (Yogurt, chocolate) | 0.405405 | 0.198198 | 0.104104 | 0.256790 | 1.295623 | 0.023753 | 1.078836 |

In [12]:

*#*

filtered\_rules = rules[(rules['antecedent support'] > 0.02)&

(rules['consequent support'] >0.01) &

(rules['confidence'] > 0.2) &

(rules['lift'] > 1.0)]

In [13]:

filtered\_rules

Out[13]:

|  | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | (Apple) | (Bread) | 0.383383 | 0.384384 | 0.154154 | 0.402089 | 1.046059 | 0.006788 | 1.029610 |
| 1 | (Bread) | (Apple) | 0.384384 | 0.383383 | 0.154154 | 0.401042 | 1.046059 | 0.006788 | 1.029482 |
| 2 | (Butter) | (Apple) | 0.420420 | 0.383383 | 0.188188 | 0.447619 | 1.167549 | 0.027006 | 1.116289 |
| 3 | (Apple) | (Butter) | 0.383383 | 0.420420 | 0.188188 | 0.490862 | 1.167549 | 0.027006 | 1.138354 |
| 4 | (Cheese) | (Apple) | 0.404404 | 0.383383 | 0.162162 | 0.400990 | 1.045925 | 0.007120 | 1.029393 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 433 | (Yogurt, Milk) | (chocolate) | 0.190190 | 0.421421 | 0.104104 | 0.547368 | 1.298862 | 0.023954 | 1.278255 |
| 434 | (chocolate, Milk) | (Yogurt) | 0.211211 | 0.420420 | 0.104104 | 0.492891 | 1.172376 | 0.015307 | 1.142909 |
| 435 | (Yogurt) | (chocolate, Milk) | 0.420420 | 0.211211 | 0.104104 | 0.247619 | 1.172376 | 0.015307 | 1.048390 |
| 436 | (chocolate) | (Yogurt, Milk) | 0.421421 | 0.190190 | 0.104104 | 0.247031 | 1.298862 | 0.023954 | 1.075489 |
| 437 | (Milk) | (Yogurt, chocolate) | 0.405405 | 0.198198 | 0.104104 | 0.256790 | 1.295623 | 0.023753 | 1.078836 |

In [14]:

filtered\_rules.sort\_values('confidence',ascending=False)

Out[14]:

|  | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 404 | (Dill, Unicorn) | (chocolate) | 0.168168 | 0.421421 | 0.101101 | 0.601190 | 1.426578 | 0.030231 | 1.450764 |
| 392 | (Milk, Dill) | (chocolate) | 0.190190 | 0.421421 | 0.114114 | 0.600000 | 1.423753 | 0.033964 | 1.446446 |
| 326 | (Cheese, Dill) | (Onion) | 0.177177 | 0.403403 | 0.102102 | 0.576271 | 1.428523 | 0.030628 | 1.407968 |
| 391 | (chocolate, Dill) | (Milk) | 0.199199 | 0.405405 | 0.114114 | 0.572864 | 1.413065 | 0.033358 | 1.392051 |
| 259 | (Kidney Beans, Ice cream) | (Butter) | 0.196196 | 0.420420 | 0.110110 | 0.561224 | 1.334913 | 0.027625 | 1.320902 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 286 | (Butter) | (Unicorn, Ice cream) | 0.420420 | 0.185185 | 0.100100 | 0.238095 | 1.285714 | 0.022244 | 1.069444 |
| 323 | (Butter) | (Nutmeg, Yogurt) | 0.420420 | 0.192192 | 0.100100 | 0.238095 | 1.238839 | 0.019299 | 1.060248 |
| 370 | (Yogurt) | (Kidney Beans, Corn) | 0.420420 | 0.195195 | 0.100100 | 0.238095 | 1.219780 | 0.018036 | 1.056306 |
| 243 | (Butter) | (Sugar, Apple) | 0.420420 | 0.182182 | 0.100100 | 0.238095 | 1.306907 | 0.023507 | 1.073386 |
| 376 | (chocolate) | (Kidney Beans, Corn) | 0.421421 | 0.195195 | 0.100100 | 0.237530 | 1.216883 | 0.017841 | 1.055523 |

In [15]:

*# Generate scatterplot confidence versus support*

sns.scatterplot(x = "support", y = "confidence", data = filtered\_rules)

plt.show()

With scatterplot, we can have quick glimpse, where the boundary should be and what metric should be set to filter out the frequent itemsets.

In [16]:

filtered\_rules = rules[(rules['antecedent support'] > 0.02)&

(rules['consequent support'] >0.01) &

(rules['confidence'] > 0.45) &

(rules['lift'] > 1.0)]

In [17]:

filtered\_rules

Out[17]:

|  | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | (Apple) | (Butter) | 0.383383 | 0.420420 | 0.188188 | 0.490862 | 1.167549 | 0.027006 | 1.138354 |
| 6 | (Corn) | (Apple) | 0.407407 | 0.383383 | 0.186186 | 0.457002 | 1.192025 | 0.029993 | 1.135579 |
| 7 | (Apple) | (Corn) | 0.383383 | 0.407407 | 0.186186 | 0.485640 | 1.192025 | 0.029993 | 1.152096 |
| 8 | (Apple) | (Dill) | 0.383383 | 0.398398 | 0.179179 | 0.467363 | 1.173104 | 0.026440 | 1.129478 |
| 15 | (Apple) | (Kidney Beans) | 0.383383 | 0.408408 | 0.176176 | 0.459530 | 1.125173 | 0.019599 | 1.094587 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 427 | (Kidney Beans, Yogurt) | (Nutmeg) | 0.194194 | 0.401401 | 0.101101 | 0.520619 | 1.297002 | 0.023151 | 1.248690 |
| 428 | (Nutmeg, Yogurt) | (Kidney Beans) | 0.192192 | 0.408408 | 0.101101 | 0.526042 | 1.288028 | 0.022608 | 1.248193 |
| 432 | (Yogurt, chocolate) | (Milk) | 0.198198 | 0.405405 | 0.104104 | 0.525253 | 1.295623 | 0.023753 | 1.252444 |
| 433 | (Yogurt, Milk) | (chocolate) | 0.190190 | 0.421421 | 0.104104 | 0.547368 | 1.298862 | 0.023954 | 1.278255 |
| 434 | (chocolate, Milk) | (Yogurt) | 0.211211 | 0.420420 | 0.104104 | 0.492891 | 1.172376 | 0.015307 | 1.142909 |

From the plot it seems like the butter can be used as cross-selling with other products, it also acts as something to be offered with antecedents that is low. Thus, the customers are more likely to buy them if the butter are offered with cheaper price if they buy the antecedents that sold less in a store

# CONCLUSION:

Market Basket Analysis using AI is a valuable tool for businesses to gain a deeper understanding of customer behaviour and optimize sales strategies. By leveraging powerful algorithms and techniques, businesses can uncover hidden patterns and associations in customer purchase data, leading to improved decision-making, enhanced customer experience, and increased sales.