PHASE-5

PROJECT DOCUMENTATION & SUBMISSION

Date	**01-11-2023**
Team ID	**689**
Project Name	**Market Basket Analysis for Fresh Product Location Improvement**

Project Title: Market Basket Analysis for Fresh Product Location Improvement

Problem Statement

Objective: To strategically place fresh products in a retail store by analyzing customer purchase data, optimizing layout, and enhancing sales and customer satisfaction.

Problem identified:

The problem you've identified for market basket analysis in the context of fresh product location improvement involves understanding customer purchase behavior with respect to fresh products in a retail setting.By applying market basket analysis and location optimization, you can enhance the shopping experience for customers and potentially increase sales of fresh products.

Introduction:

In the world of retail, optimizing the placement of fresh products within a store is crucial for increasing sales, enhancing customer satisfaction, and improving overall shopping experiences. Market Basket Analysis is a powerful data-driven approach that can provide valuable insights to achieve these goals.

Market Basket Analysis, often referred to as association rule mining, is a data mining technique used to uncover patterns and relationships between products that are frequently purchased together. It enables retailers to understand customer purchasing behavior, identify product associations, and make data-driven decisions to enhance product placement.

Fresh products, such as fruits, vegetables, dairy, and meat, hold a special place in the retail industry due to their perishable nature and high customer demand. The strategic placement of fresh products can significantly impact sales and customer satisfaction. Therefore, leveraging Market Basket Analysis for fresh product location improvement is essential.

DATA:

Data consists of customer purchase records, detailing items bought together, enabling analysis to optimize fresh product placement in a retail store for improved sales and customer experience.

Key Challenges:

1. Data Quality:

Ensuring accurate and complete transaction data is crucial for meaningful insights.

2. Seasonal Variations:

Fresh products often have seasonal demand, making year-round optimization challenging.

3. Inventory Constraints:

Limited shelf space may restrict optimal placement options.

4. Customer Preferences:

Individual shopping habits and preferences can vary widely, making it complex to generalize placement decisions.

5. Dynamic Shopping Patterns:

Customer behavior evolves over time, necessitating continuous analysis and adjustment of product placement.

LITERATURE SURVEY

1. “Market Basket Analysis for Fresh Product Location Improvement:A case study of E-commerce Business Warehouse",Naragain Phumchusri[2022]

Market Basket Analysis (MBA) uses the data mining technique as an analysis tool to understand association among many items. It is a useful tool for extracting information from large amount of data in many industrial areas, e.g., grocery, supermarkets, retailers, warehouse, mobile showroom, libraries, zoos, etc. The case-study company sells fresh products in E-commence business and currently has inefficient product location in warehouse, causing delays in picking process. Thus, the goal of this paper is to propose a market basket analysis method to gain insights from historical transactions, a set of recording data result in connections with sales-purchase activities, of the case-study company. Apiori algorithm is applied for association rules to analyze 2366 transactions data between July and December 2021. The results of data analysis are then utilized in rearranging products location in warehouse to reduce average picking distance per order. The results show that the average distance per order can be reduced by 54.4%.

2. “Market Basket Analysis of Basket Data with Demographics:A Case Study in E-Retailing”,Inanc Kabasakal [2021]

Businesses overcome with a high degree of competition that necessitates customer-focused strategies in most industries. In a digitalized business environment, the implementation of such strategies often requires the analysis

of customer data. Market basket analysis is a well-known method in marketing that examines basket data to discover useful information about customers' purchase intentions. The analysis has been a playground for data mining researchers that aim to overcome with its practical challenges. Our study extends the conventional basket analysis by incorporating demographic variables along with purchase transactions. With such modification, we provide an example for the extraction of segment-specific rules that relate product-level purchase decisions with gender, location, and age group. For this purpose, we present a case study on monthly basket data obtained from an e-retailer in Turkey. Our findings demonstrate association rules that might guide marketing practitioners who need to discover segment-specific purchase patterns to designate personalized promotions.

3. "Market Basket Analysis Insights to Support Category Management",Luis Aburto [2018]

Purpose of this paper aims to present an approach to detect interrelations among product categories, which are then used to produce a partition of a retailer's business into subsets of categories. The methodology also yields a segmentation of shopping trips based on the composition of each shopping basket. Design/methodology/approach This work uses scanner data to uncover product category interdependencies. As the number of possible relationships among them can be very large, the authors introduce an approach that generates an intuitive graphical representation of these interrelationships by using data analysis techniques available in standard statistical packages, such as multidimensional scaling and clustering. Findings The methodology was validated using data from a supermarket store. The analysis for that particular store revealed four groups of products categories that are often jointly purchased. The study of each of these groups allowed us to conceive the retail store under study as a small set of sub-businesses. These conclusions reinforce the strategic need for proactive coordination of marketing activities across interrelated product categories. Research limitations/implications The approach is sufficiently general to be applied beyond the supermarket industry. However, the empirical findings are specific to the store under analysis. In addition, the proposed methodology identifies cross-category interrelations, but not their

underlying sources (e.g. marketing or non-marketing interrelations). Practical implications The results suggest that retailers could potentially benefit if they transition from the traditional category management approach where retailers manage product categories in isolation into a customer management approach where retailers identify, acknowledge and leverage interrelations among product categories. Originality/value The authors present a fast and wide-range approach to study the shopping behavior of customers, detect cross-category interrelations and segment the retailer's business and customers based on information about their shopping baskets. Compared to existing approaches, its simplicity should facilitate its implementation by practitioners.

4. "Contextual Market Basket Analysis During Covid-19",Sani M Isa[2023]

One form of Data Mining application to analyze Market Basket Analysis. Market Basket Analysis helps identify buying patterns formed from concurrent transactions. One of the problems with Market Basket Analysis is that customer needs vary according to season and time of day, especially during this covid-19 season. For this purpose, by using the Artificial Neural Network (ANN) Approach that is connected to Market Basket Analysis, it can analyze and compare purchasing patterns and can identify rules that were formed before and after covid-19; several rule changes were found due to changes in people's behavior patterns.

5. "Market Basket Analysis using Apriori Algorithm",Dr.Yojna Arora [2022]

A Technique that check for dependency for one Data item to another is Association Rule which is an old Data mining approach. Which is used to identify the next product that might interest a customer. The Apriori Algorithm is applied in this for mining frequent products sets and relevant Association rule. With this

algorithm we can use this for up-sell and also in cross-sell to show the Association rule with the help of the algorithm. These methods are widely used in global companies, so for the good understanding the companies used the methods to remain up to date that what customers demands with which products. The results helps the big retailers to identify a trend for customers buying patterns, which is very helpful information for the retailers to plan their big business operations.

DESIGN THINKING

Design Thinking Approach

1. Empathize:

- Understand customer shopping behaviors, preferences, and pain points through surveys, interviews, and observations.

2. Define:

- Clearly define the problem and goals. For instance, improving sales and customer experience through better fresh product placement.

3. Ideate:

- Brainstorm ideas and potential solutions. Consider innovative ways to optimize fresh product placement based on customer insights.

4. Prototype:

- Create mock layouts or store designs based on the ideated solutions. Experiment with different placements virtually or physically.

5. Test:

- Implement the prototype in a controlled store environment or simulate changes in a virtual setting.

- Collect data on customer responses, sales, and satisfaction.

6. Iterate:

- Analyze the test results and gather feedback from customers and store staff.

- Refine the placement strategy based on the findings.

7. Implement:

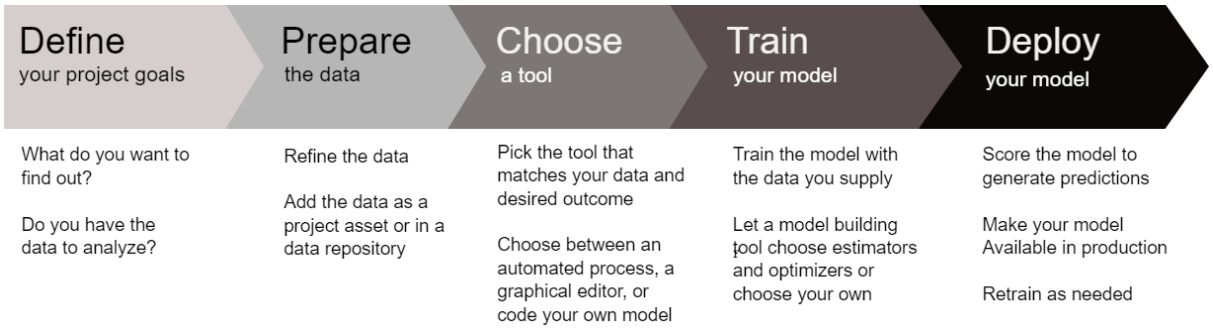
- Roll out the optimized fresh product placement strategy across the store or chain.

8. Evaluate:

- Continuously monitor sales, customer satisfaction, and other relevant metrics.

- Make further adjustments as needed to maintain and enhance the solution.

TECHNOLOGY ARCHITECTURE



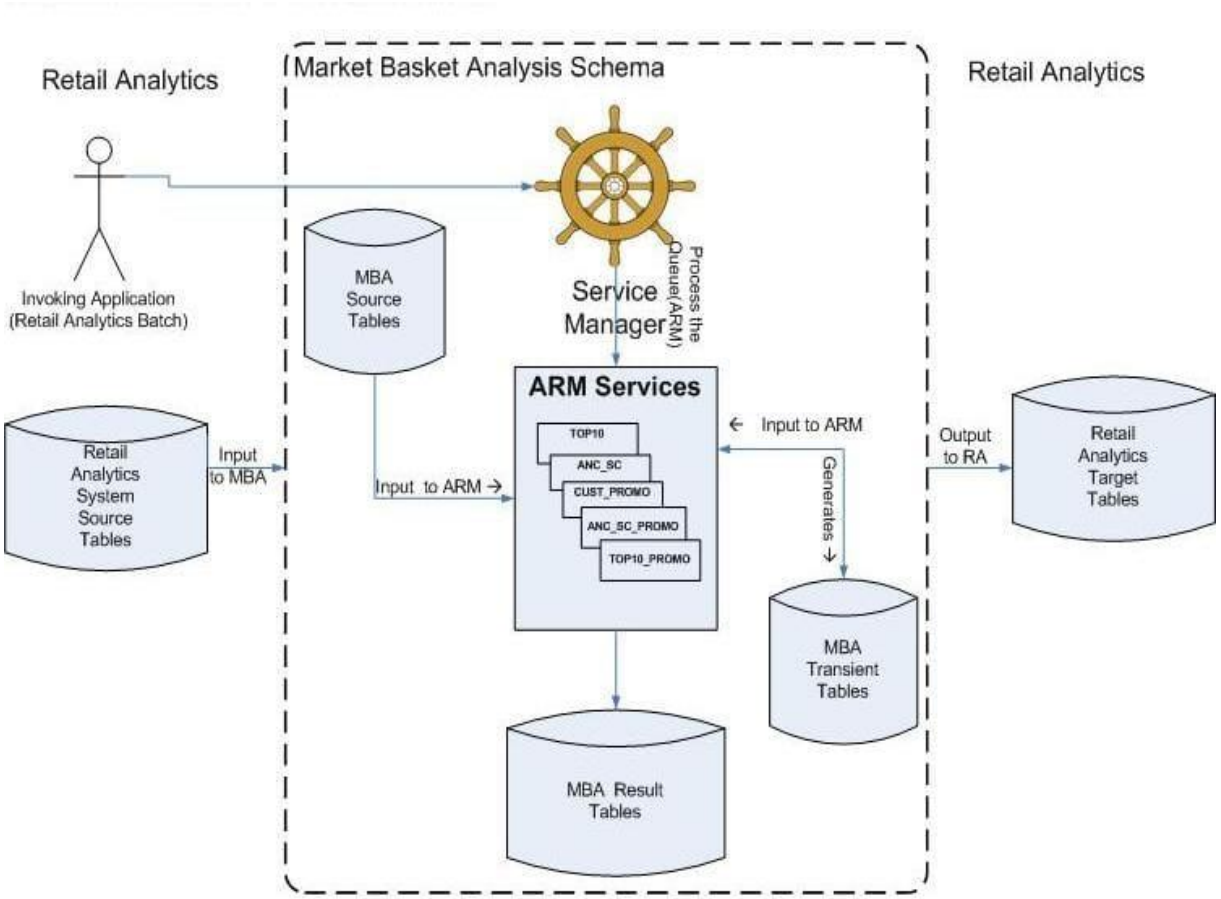


figure -1

Technology Architecture for Market Basket Insights :

The technology architecture for Market Basket Analysis for fresh product location improvement involves various components and tools to collect, process, and analyze data. This technology architecture provides the foundation for implementing Market Basket Analysis for fresh product location improvement, helping retailers make data-driven decisions to enhance their store layout and increase customer satisfaction.

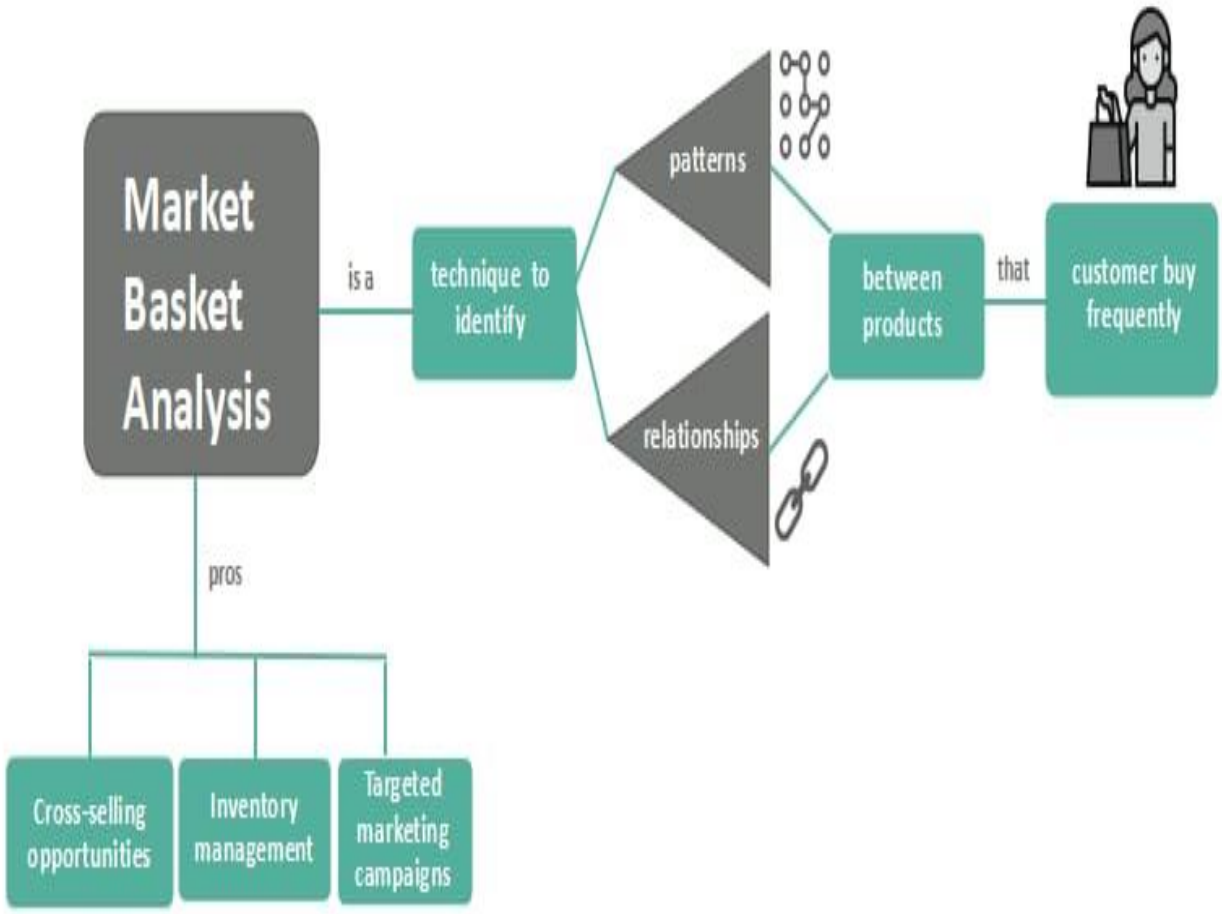


figure -2

1. Data Collection and Storage:

Data Collection:

a.Point of Sale (POS) Data:

Gather transaction data from your retail store, including what items were purchased together in each transaction. This data should include details like product IDs, transaction timestamps, and customer information.

b. Product Information:

Collect data about your fresh products, such as their categories, descriptions, and attributes (e.g., organic, brand, weight).

c. Store Layout:

Gather information about the physical layout of your store, including the location of fresh product aisles, shelves, and any other relevant details.

Data Storage:

a. Database:

Store the cleaned and analyzed data in a relational database (e.g., MySQL, PostgreSQL) for efficient querying and retrieval.

b. Data Warehouse:

Consider using a data warehouse solution (e.g., Amazon Redshift, Google BigQuery) for large-scale data storage and analysis.

Visualization and Reporting:

Create visualizations and reports to present the results of your market basket analysis. This could include insights on which fresh products are frequently purchased together and recommendations for improving their location within the store.

2. Data Preprocessing:

Data Cleaning:

Remove duplicates: Eliminate duplicate transactions or items within transactions.

Handle missing values: Address any missing data points, which can include transaction IDs, product IDs, or other relevant attributes.

Filter out irrelevant data: Exclude transactions or products that are not applicable to your analysis.

Transaction Identification:

Group items by transaction: Organize the data into groups of items purchased together in a single transaction. This is essential for association rule mining.

Create a transaction ID: Assign a unique identifier to each transaction for reference.

Data Transformation:

One-Hot Encoding: Transform categorical variables like product categories or brands into binary (0/1) values to make them suitable for analysis.

Quantify numerical data**:** Ensure that any numerical data is in a consistent format and scale for accurate analysis.

3. Model Development and Training:

Data Preparation:

Ensure that your dataset is preprocessed as discussed earlier, with transaction data organized in the right format.

Algorithm Selection:

Choose a suitable algorithm for market basket analysis. Popular choices include Apriori, FP-growth, or Eclat for frequent itemset mining.

Setting Parameters:

Configure parameters for the chosen algorithm, such as minimum support and minimum confidence levels. These values impact the rules generated during the analysis.

4. Model Deployment:

Plan the Deployment:

Define a clear plan for implementing the changes in fresh product location based on the insights from your market basket analysis.

Communication and Training:

Communicate the new strategies and location changes to store staff. Ensure they understand the goals and the reasons behind the changes.Train the staff on the new product placement guidelines.

Implementation:

Physically rearrange the store layout to reflect the recommended changes. For example, if the analysis suggests that bread and butter are frequently purchased together, place them closer on the shelves.Update signage and labels to reflect the new product locations**.**

5. Security and Access Control:

Data Encryption:

Implement encryption for data both in transit and at rest. Use secure communication protocols (e.g., HTTPS) and encryption mechanisms to protect the data stored in databases.

Access Control:

Restrict access to the market basket analysis data to authorized personnel only. Implement role-based access control (RBAC) to ensure that individuals have appropriate access privileges based on their roles and responsibilities.

Authentication:

Use strong authentication methods to verify the identity of users who access the data. Multi-factor authentication (MFA) can provide an additional layer of security.

6. Real-time Prediction:

Data Processing:

Continuously process and update your data to ensure it reflects current customer behaviors.

Real-Time Predictions:

Use machine learning models or algorithms to make real-time predictions about which fresh products should be placed together in the store to maximize sales and customer satisfaction. These predictions can be based on factors like historical purchase patterns, seasonality, and even weather conditions.

Store Layout Optimization:

Based on the predictions, adjust the placement of fresh products within the store to encourage cross-selling. For example, if tomatoes and lettuce are frequently bought together, place them in close proximity to each other**.**

7. Monitoring and Maintenance:

Technology and Tools:

Stay updated on the latest technologies and tools in data analytics and retail management to enhance your market basket analysis efforts.

Team Collaboration:

Foster collaboration between your data analysts, marketing, and store management teams to share insights and work together to optimize fresh product placement continually.

Long-Term Planning:

Develop a long-term strategy for fresh product location improvement to ensure that it aligns with your store's evolving goals and the changing needs of your customer base.

8. Scalability and Redundancy:

Data Scalability:

As your business grows, the volume of data you collect will increase. Ensure that your data infrastructure can handle this growth by using scalable databases and data storage solutions.

Computational Resources:

Implement a system that can scale with increased computational demands. This may involve using cloud computing services, which can easily scale up or down based on your needs.

Data Redundancy:

Maintain redundant copies of your critical data to ensure data availability in case of hardware failures. This may involve using RAID configurations or distributed databases.

Load Balancing:

Implement load balancing for your data processing and analysis tasks. This helps distribute the workload evenly and ensures that a single point of failure doesn't disrupt your operations**.**

9. User Interface (Optional):

User Customization:

Allow users to customize the dashboard and reports to focus on the specific aspects of market basket analysis that are most relevant to their needs.

Search and Filter Functionality:

Implement search and filter options for users to quickly find specific products, time periods, or store locations for analysis.

Historical Data Access:

Provide access to historical data and insights to help users track changes and improvements in fresh product placement over time.

Real-Time Updates:

Ensure that the UI can display real-time or near-real-time data updates to keep users informed of the latest market basket analysis results.

10. Data Feedback Loop (Optional):

Feedback Collection:

Gather feedback from store managers, employees, and customers. Their insights can help you understand the practical implications of the changes and whether they align with customer preferences.

Data Integration:

Integrate the feedback data with your market basket analysis dataset. This feedback can include comments on the impact of product placement changes, customer complaints, or positive feedback.

Continuous Analysis:

Regularly analyze the integrated data, looking for correlations between the implemented changes and customer responses.

Adjustments and Iteration:

Based on the analysis, make adjustments to your fresh product placement strategies as needed. This might involve fine-tuning recommendations or reverting changes that didn't yield positive results.

MODULES DESCRIPTION

1. Data Collection and Storage Module:

- Objective: This module focuses on gathering and storing datasets for training the market Basket insights model.

- Key Tasks:

- Identify and access relevant data sources (e.g., Kaggle).
- Extract datasets containing labeled insights and benign examples.
- Store datasets in on-premises database.

2. Data Preprocessing Module:

- Objective: Prepare the dataset for model training by cleaning, transforming, and engineering features.

- Key Tasks:

- Data cleaning to handle missing values and remove duplicates.

- Feature engineering to create relevant features.

- Data transformation, including encoding categorical features and scaling numerical features.

3. Model Development and Training Module:

- Objective: To develop and train machine learning models for basket insights detection.

- Key Tasks:

- Utilize IBM AI feature for model selection.

- Perform hyperparameter tuning to optimize model performance.

- Evaluate model performance using metrics like accuracy, precision, recall, and F1-score.

4. Model Deployment Module:

- Objective: Deploy the selected Jupyter notebook as a web service with an API endpoint.

- Key Tasks:

- Create a deployment space within Jupyter notebook.

- Deploy the model to the cloud, making it accessible via the API.

- Configure the API endpoint for real-time predictions.

5. Security and Access Control Module:

- **Objective:** Ensure secure access to the deployed model.

- **Key Tasks:**

  - Implement API key-based authentication to secure API access.

  - Configure permissions and access policies for controlling user access.

6. Real-time Prediction Module:

- **Objective:** Enable users and applications to make real-time predictions using the deployed model.

- **Key Tasks:**

  - Develop a user-friendly interface or integrate the API with other systems.

  - Enable users to send input data for prediction.

ALGORITHM AND TECHNOLOGY USED

1. Data Collection and Preprocessing:

- Technology: Python (for data handling)

- Description: Collect the dataset from Kaggle, which includes features related to the content. Preprocess the data by handling missing values, encoding categorical features, scaling numerical features, and any other necessary data transformations.

2. Model Development using AutoAI:

- Technology: Python

- Algorithm: Apriori Algorithm

- Description: Leverage Jupyter notebook feature to automate the model development process. AutoAI performs the following tasks:

  - Feature engineering

  - Model selection

  - Hyperparameter tuning

  - Cross-validation

3. Model Evaluation and Selection:

- Technology: Python

- Algorithm: Apriori Algorithm

- Description: It generates multiple candidate models using basket insights. Evaluate these models using performance metrics like accuracy, precision, recall, and F1-score to select the best-performing model.

4. Model Deployment:

- Technology: Python

- Algorithm: None (deployment is not algorithm-based)

- Description: Create a deployment space in python, then deploy the selected model as a web service with an API endpoint for real-time predictions. This process involves packaging the model and its dependencies for deployment.

5. Security and Access Control:

- Technology: Python

- Description: Implement security measures, including API key-based authentication, to secure access to the deployed model. Configure permissions and access policies to control who can use the API.

6. Real-time Prediction:

- Technology:Python

- Description: Develop an interface or integrate the API with other systems or applications that require real-time market Basket insights. Users or applications can send data to the API for prediction.

```
import pandas as pd
import numpy as np

#for viz
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

#to avoid warning
import warnings
warnings.filterwarnings('ignore')

#to display all feature if the number increase
pd.set_option('display.max_columns', None)

data=pd.read_excel('/content/Assignment-1_Data.xlsx')
```

```
data.tail()
```

	BillNo	Itemname	Quantity	Date	Price	CustomerID	Country
522059	581587	PACK OF 20 SPACEBOY NAPKINS	12	2011-12-09 12:50:00	0.85	12680.0	France
522060	581587	CHILDREN'S APRON DOLLY GIRL	6	2011-12-09 12:50:00	2.10	12680.0	France
522061	581587	CHILDRENS CUTLERY DOLLY GIRL	4	2011-12-09 12:50:00	4.15	12680.0	France
522062	581587	CHILDRENS CUTLERY CIRCUS PARADE	4	2011-12-09 12:50:00	4.15	12680.0	France
522063	581587	BAKING SET 9 PIECE RETROSPOT	3	2011-12-09 12:50:00	4.95	12680.0	France

```
data.head()
```

	BillNo	Itemname	Quantity	Date	Price	CustomerID	Country
0	536365	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 522064 entries, 0 to 522063
Data columns (total 7 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   BillNo      522064 non-null  object
 1   Itemname    520609 non-null  object
 2   Quantity    522064 non-null  int64
 3   Date        522064 non-null  datetime64[ns]
 4   Price       522064 non-null  float64
 5   CustomerID  388023 non-null  float64
 6   Country     522064 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(1), object(3)
memory usage: 27.9+ MB
```

```
data.shape
```

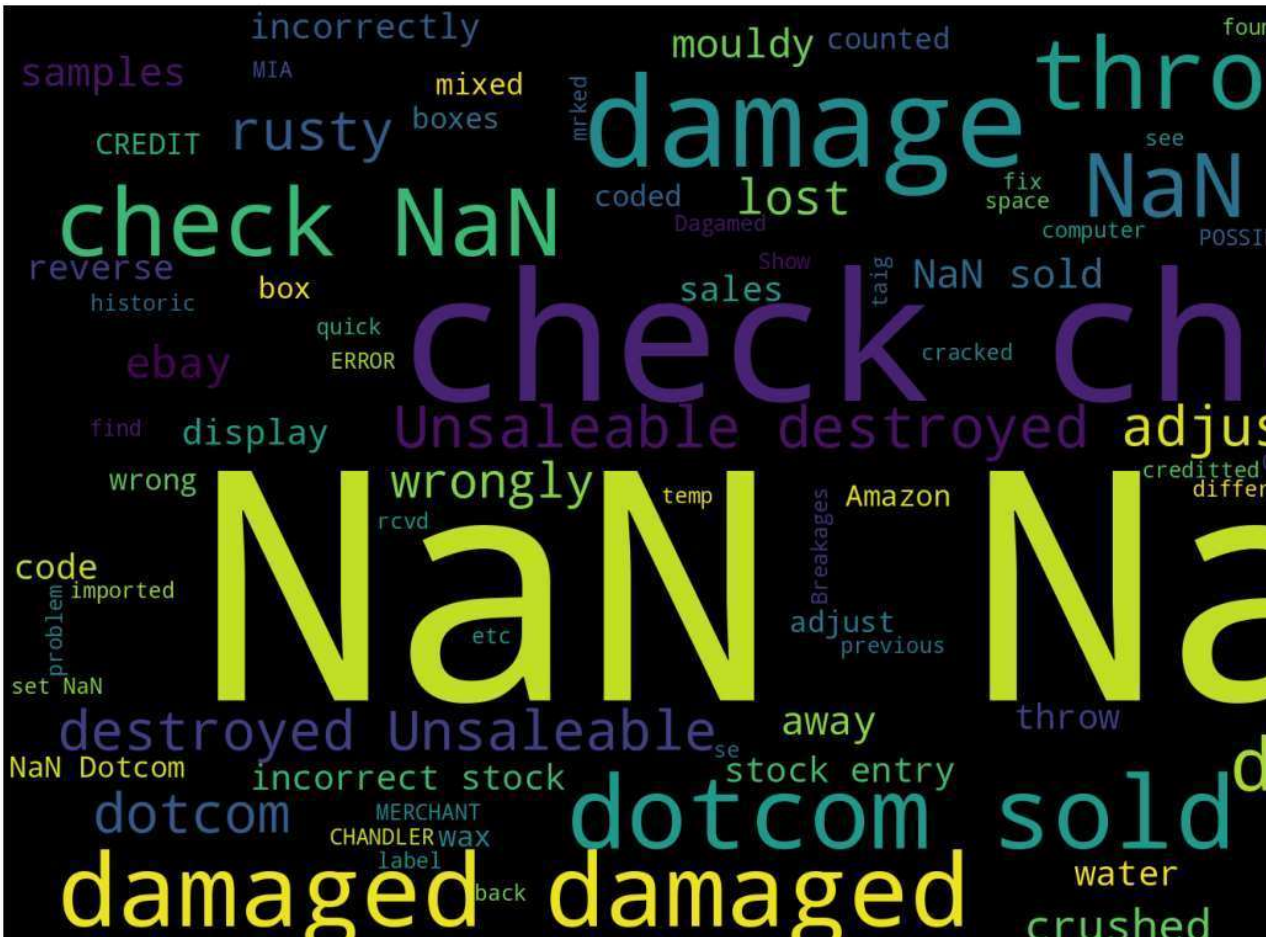
```
(522064, 7)
```

```
data.describe()
```

	Quantity	Price	CustomerID
count	522064.000000	522064.000000	388023.000000
mean	10.090435	3.826801	15316.931710
std	161.110525	41.900599	1721.846964
min	-9600.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13950.000000

```
from wordcloud import WordCloud, STOPWORDS
stopwords = STOPWORDS
worldcloud= WordCloud(background_color='Black',stopwords=stopwords, height=1000, width =2000)

temp=data[data['Quantity']<0]
body =temp['Itemname'].to_string(index=False)
### Generate word cloud
worldcloud.generate(body)
## Visualize
plt.figure(figsize=(22,10))
plt.imshow(worldcloud)
plt.axis("off")
```

```
(-0.5, 1999.5, 999.5, -0.5)
```

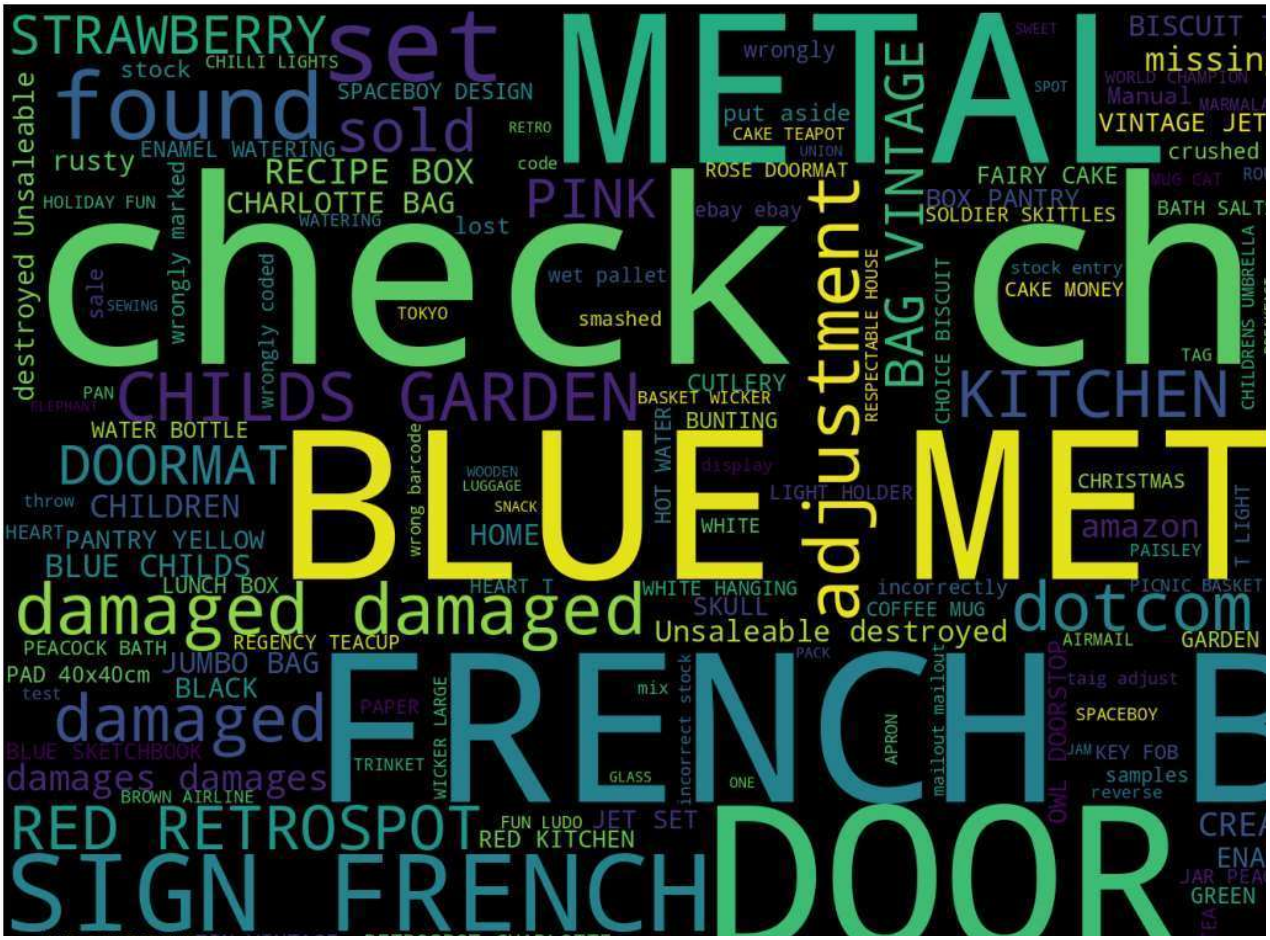



```
data[data['Quantity']<0]
```

	BillNo	Itemname	Quantity	Date	Price	CustomerID	Country
2359	536589	NaN	-10	2010-12-01 16:50:00	0.0	NaN	United Kingdom
4289	536764	NaN	-38	2010-12-02 14:42:00	0.0	NaN	United Kingdom
6998	536996	NaN	-20	2010-12-03 15:30:00	0.0	NaN	United Kingdom
6999	536997	NaN	-20	2010-12-03 15:30:00	0.0	NaN	United Kingdom
7000	536998	NaN	-6	2010-12-03 15:30:00	0.0	NaN	United Kingdom
...	...	...	...	...	...	...	...
515634	581210	check	-26	2011-12-07 18:36:00	0.0	NaN	United Kingdom

```
#next we can see for price small or equal to 0
temp=data[data['Price']<=0]
body =temp['Itemname'].dropna().to_string(index=False)
### Generate word cloud
worldcloud.generate(body)
## Visualize
plt.figure(figsize=(22,10))
plt.imshow(worldcloud)
plt.axis("off")
```

```
(-0.5, 1999.5, 999.5, -0.5)
```




```
# check for duplicate entries
data.duplicated().sum()
```

```
5286
```

```
# there are 5286 duplicates transcations are present in the dataset Lets remove them
data.drop_duplicates(inplace=True)
```

```
#Let remove the space in that word
data['Itemname'] = data['Itemname'].str.strip()

#Lets Check for null Values
data.isnull().sum()
```

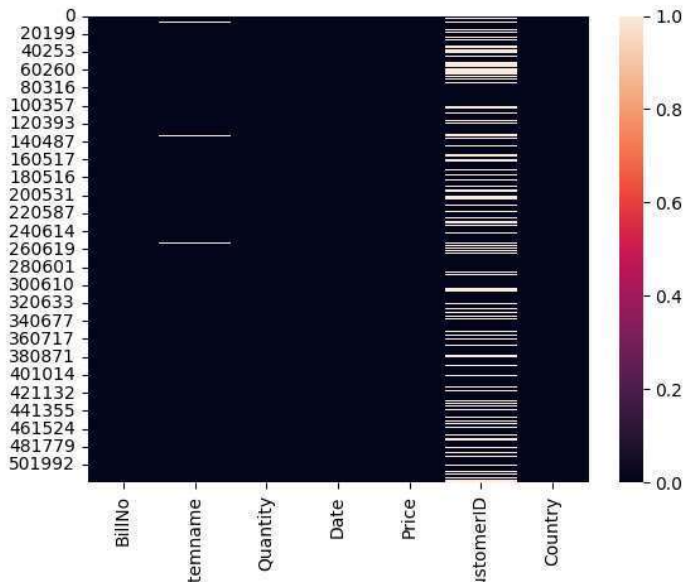
```
BillNo            0
Itemname       1455
Quantity          0
Date              0
Price             0
CustomerID    133967
Country           0
dtype: int64
```

```
data.isnull().mean()*100
```

```
BillNo         0.000000
Itemname       0.281552
Quantity       0.000000
Date           0.000000
Price          0.000000
CustomerID    25.923511
Country        0.000000
dtype: float64
```

```
sns.heatmap(data.isnull())
```

```
<Axes: >
```




```
data['Date']
```

```
0        2010-12-01 08:26:00
1        2010-12-01 08:26:00
2        2010-12-01 08:26:00
3        2010-12-01 08:26:00
4        2010-12-01 08:26:00
                ...
522059   2011-12-09 12:50:00
522060   2011-12-09 12:50:00
522061   2011-12-09 12:50:00
522062   2011-12-09 12:50:00
522063   2011-12-09 12:50:00
Name: Date, Length: 516778, dtype: datetime64[ns]
```

```
#we can spearate the Data and time to different columns

import datetime as datetime
from datetime import datetime

#datetime.strptime('2013-01-01 09:10:12', '%Y-%m-%d %H:%M:%S')
```

```
data['date'] = data['Date'].dt.date
data['hour'] = data['Date'].dt.hour

### Converting invoice date to data time
data['date']= pd.to_datetime(data['date'], infer_datetime_format= True)
data.drop('Date',inplace=True,axis=1)

data.head(3)
```

	BillNo	Itemname	Quantity	Price	CustomerID	Country	date	hour
0	536365	WHITE HANGING HEART T-LIGHT HOLDER	6	2.55	17850.0	United Kingdom	2010-12-01	8
1	536365	WHITE METAL LANTERN	6	3.39	17850.0	United Kingdom	2010-12-01	8
2	536365	CREAM CUPID HEARTS COAT HANGER	8	2.75	17850.0	United Kingdom	2010-12-01	8

```
data[data['Quantity']<=0]
```

	BillNo	Itemname	Quantity	Price	CustomerID	Country	date	hour
2359	536589	NaN	-10	0.0	NaN	United Kingdom	2010-12-01	16
4289	536764	NaN	-38	0.0	NaN	United Kingdom	2010-12-02	14
6998	536996	NaN	-20	0.0	NaN	United Kingdom	2010-12-03	15
6999	536997	NaN	-20	0.0	NaN	United Kingdom	2010-12-03	15
7000	536998	NaN	-6	0.0	NaN	United Kingdom	2010-12-03	15
...	...	...	...	...	...	...	...	...
515634	581210	check	-26	0.0	NaN	United Kingdom	2011-12-07	18
515636	581212	lost	-1050	0.0	NaN	United Kingdom	2011-12-07	18
515637	581213	check	-30	0.0	NaN	United Kingdom	2011-12-07	18
517209	581226	missing	-338	0.0	NaN	United Kingdom	2011-12-08	9
519172	581422	smashed	-235	0.0	NaN	United Kingdom	2011-12-08	15

```
data[data['Price']<=0]
```

	BillNo	Itemname	Quantity	Price	CustomerID	Country	date	hour
613	536414	NaN	56	0.0	NaN	United Kingdom	2010-12-01	11
1937	536545	NaN	1	0.0	NaN	United Kingdom	2010-12-01	14
1938	536546	NaN	1	0.0	NaN	United Kingdom	2010-12-01	14
1939	536547	NaN	1	0.0	NaN	United Kingdom	2010-12-01	14
1940	536549	NaN	1	0.0	NaN	United Kingdom	2010-12-01	14
...	...	...	...	...	...	...	...	...
517266	581234	NaN	27	0.0	NaN	United Kingdom	2011-12-08	10
518770	581406	POLYESTER FILLER PAD 45x45cm	240	0.0	NaN	United Kingdom	2011-12-08	13
518771	581406	POLYESTER FILLER PAD 40x40cm	300	0.0	NaN	United Kingdom	2011-12-08	13
518820	581408	NaN	20	0.0	NaN	United Kingdom	2011-12-08	14
519172	581422	smashed	-235	0.0	NaN	United Kingdom	2011-12-08	15

```
#remove the rows which has the buyed quality is small or equal to zero
data=data[data['Quantity']>0]

#remove the rows which price is small or equal to zero
data=data[data['Price']>0]
data.shape
```

```
(514270, 8)
```

```
# Lets Check for Quantity
data.sort_values(by='Quantity',ascending=False)
```

	BillNo	Itemname	Quantity	Price	CustomerID	Country	date	hour
520583	581483	PAPER CRAFT , LITTLE BIRDIE	80995	2.08	16446.0	United Kingdom	2011-12-09	9
59999	541431	MEDIUM CERAMIC TOP STORAGE JAR	74215	1.04	12346.0	United Kingdom	2011-01-18	10
405138	573008	WORLD WAR 2 GLIDERS ASSTD DESIGNS	4800	0.21	12901.0	United Kingdom	2011-10-27	12
198929	554868	SMALL POPCORN HOLDER	4300	0.72	13135.0	United Kingdom	2011-05-27	10
94245	544612	EMPIRE DESIGN ROSETTE	3906	0.82	18087.0	United Kingdom	2011-02-22	10
...	...	...	...	...	...	...	...	...
382839	571243	LE GRAND TRAY CHIC SET	1	12.50	14595.0	United Kingdom	2011-10-14	15
382842	571243	REGENCY CAKESTAND 3 TIER	1	12.75	14595.0	United Kingdom	2011-10-14	15
160161	550835	DOORMAT ENGLISH ROSE	1	7.95	15034.0	United Kingdom	2011-04-21	10
160164	550836	CLASSIC METAL BIRDCAGE PLANT HOLDER	1	12.75	14759.0	United Kingdom	2011-04-21	10
356890	569208	GLASS APOTHECARY BOTTLE PERFUME	1	3.95	16693.0	United Kingdom	2011-10-02	11

514270 rows × 8 columns

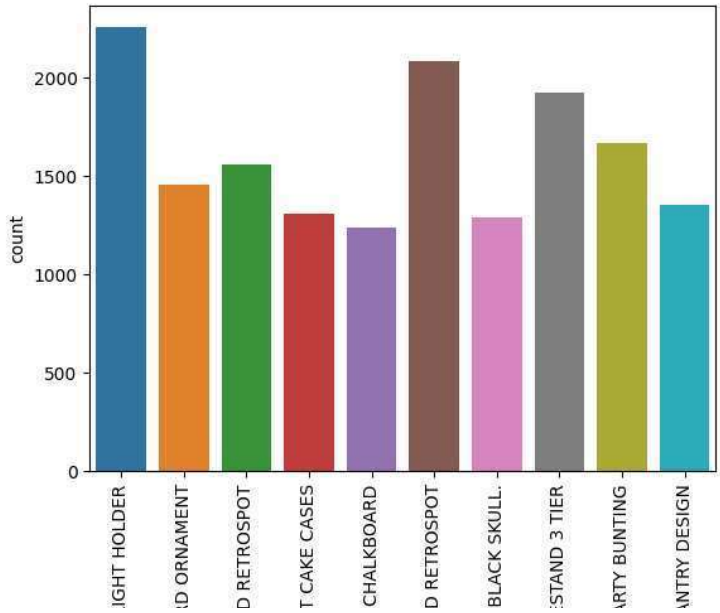
```
#Lets check For Price
data.sort_values(by='Price',ascending=False)
```

	BillNo	Itemname	Quantity	Price	CustomerID	Country	date	hour
14696	537632	AMAZON FEE	1	13541.330	NaN	United Kingdom	2010-12-07	15
288772	A563185	Adjust bad debt	1	11062.060	NaN	United Kingdom	2011-08-12	14
167329	551697	POSTAGE	1	8142.750	16029.0	United Kingdom	2011-05-03	13
286674	562955	DOTCOM POSTAGE	1	4505.170	NaN	United Kingdom	2011-08-11	10
258372	560373	Manual	1	4287.630	NaN	United Kingdom	2011-07-18	12
...	...	...	...	...	...	...	...	...
504147	580513	POPART WOODEN PENCILS ASST	100	0.040	14456.0	United Kingdom	2011-12-04	13
346188	568200	PADS TO MATCH ALL CUSHIONS	1	0.001	16198.0	United Kingdom	2011-09-25	14
347944	568375	Bank Charges	1	0.001	13405.0	United Kingdom	2011-09-26	17
268879	561226	PADS TO MATCH ALL CUSHIONS	1	0.001	15618.0	United Kingdom	2011-07-26	10
151547	550193	PADS TO MATCH ALL CUSHIONS	1	0.001	13952.0	United Kingdom	2011-04-15	9

514270 rows × 8 columns

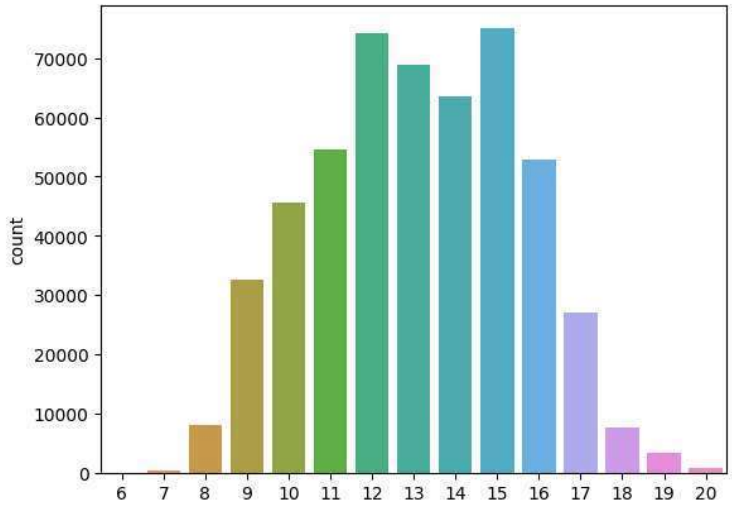
```
data=data[data['Price']<5000]
data=data[data['Quantity']<5000]

# Get the top 10 item names by count
top_10_items = data['Itemname'].value_counts().nlargest(10).index

# Filter the DataFrame to include only the top 10 item names
df_top_10 = data[data['Itemname'].isin(top_10_items)]

# Create a countplot for the top 10 item names
ax=sns.countplot(data=df_top_10, x='Itemname')
plt.xticks(rotation=90)
```

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'WHITE HANGING HEART T-LIGHT HOLDER'),
  Text(1, 0, 'ASSORTED COLOUR BIRD ORNAMENT'),
  Text(2, 0, 'LUNCH BAG RED RETROSPOT'),
  Text(3, 0, 'PACK OF 72 RETROSPOT CAKE CASES'),
  Text(4, 0, 'NATURAL SLATE HEART CHALKBOARD'),
  Text(5, 0, 'JUMBO BAG RED RETROSPOT'),
  Text(6, 0, 'LUNCH BAG  BLACK SKULL.'),
  Text(7, 0, 'REGENCY CAKESTAND 3 TIER'),
  Text(8, 0, 'PARTY BUNTING'),
  Text(9, 0, 'SET OF 3 CAKE TINS PANTRY DESIGN')])
```



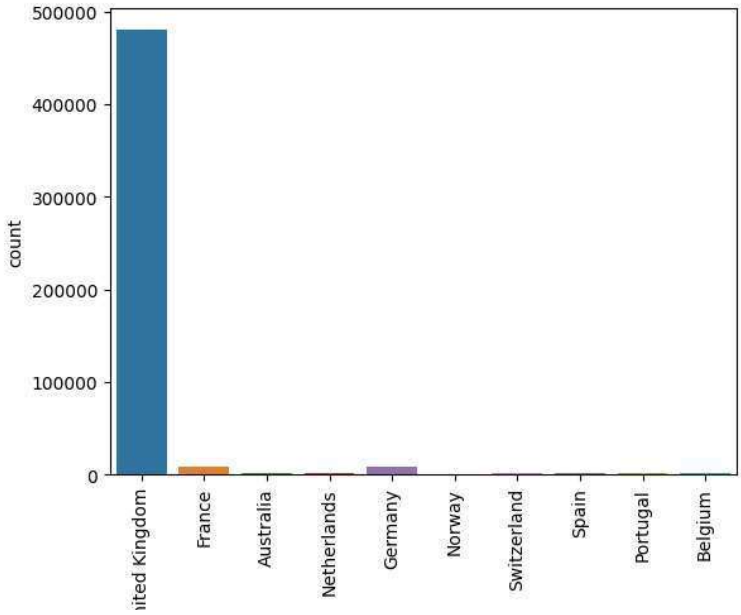

```
#lets do some viz

sns.countplot(data=data,x='hour')
```

```
<Axes: xlabel='hour', ylabel='count'>
```




```
# Get the top 5 country names by count
top_5_country = data['Country'].value_counts().nlargest(10).index

# Filter the DataFrame to include only the top 10 item names
df_top_5 = data[data['Country'].isin(top_5_country)]

# Create a countplot for the top 10 item names
ax=sns.countplot(data=df_top_5, x='Country')
plt.xticks(rotation=90)
```

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'United Kingdom'),
  Text(1, 0, 'France'),
  Text(2, 0, 'Australia'),
  Text(3, 0, 'Netherlands'),
  Text(4, 0, 'Germany'),
  Text(5, 0, 'Norway'),
  Text(6, 0, 'Switzerland'),
  Text(7, 0, 'Spain'),
  Text(8, 0, 'Portugal'),
  Text(9, 0, 'Belgium')])
```




```
data.Country.value_counts().head(10)
```

```
United Kingdom    479903
Germany             9025
France              8392
Spain               2479
Netherlands         2359
Belgium             2031
Switzerland         1958
Portugal            1492
Australia           1181
Norway              1071
Name: Country, dtype: int64
```

```
#lets import the models
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

mybasket= (data[data['Country'] =="Germany"]
          .groupby(['BillNo', 'Itemname'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('BillNo'))

mybasket
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_c
  and should_run_async(code)
```

Itemname	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	12 PENCIL SMALL TUBE WOODLAND	12 PENCILS SMALL TUBE RED RETROSPOT	12 PENCILS SMALL TUBE SKULL	12 PENCILS TALL TUBE POSY	12 PENCILS TALL TUBE RED RETROSPOT	12 PENCILS TALL TUBE SKULLS	12 PENCILS TALL TUBE WOODLAND	12 PINK HEN+CHICKS IN BASKET	R S
BillNo													
536527	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
536840	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

```
a=data[data['Country']=='Germany']
a['BillNo'].nunique()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_
  and should_run_async(code)
457
```

```
def my_encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1

my_basket = mybasket.applymap(my_encode_units)
my_basket.drop('POSTAGE', inplace=True, axis=1) #Remove "postage" as an item
### Display sample of set
my_basket
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_c
  and should_run_async(code)
```

Itemname	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	12 PENCIL SMALL TUBE WOODLAND	12 PENCILS SMALL TUBE RED RETROSPOT	12 PENCILS SMALL TUBE SKULL	12 PENCILS TALL TUBE POSY	12 PENCILS TALL TUBE RED RETROSPOT	12 PENCILS TALL TUBE SKULLS	12 PENCILS TALL TUBE WOODLAND	12 PINK HEN+CHICKS IN BASKET	R S
BillNo													
536527	0	0	0	0	0	0	0	0	0	0	0	0	
536840	0	0	0	0	0	0	0	0	0	0	0	0	
536861	0	0	0	0	0	0	0	0	0	0	0	0	
536967	0	0	0	0	0	0	0	0	0	0	0	0	
536983	0	0	0	0	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
581266	0	0	0	0	0	0	0	0	0	0	0	0	
581494	0	0	0	0	0	0	0	0	0	0	0	0	
581570	0	0	0	0	0	0	0	0	0	0	0	0	
581574	0	0	0	0	0	0	0	0	0	0	0	0	
581578	0	0	0	0	0	0	0	0	0	0	0	0	

```
data[data['BillNo']==536527]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_c
  and should_run_async(code)
```

	BillNo	Itemname	Quantity	Price	CustomerID	Country	date	hour
1099	536527	SET OF 6 T-LIGHTS SANTA	6	2.95	12662.0	Germany	2010-12-01	13
1100	536527	ROTATING SILVER ANGELS T-LIGHT HLDR	6	2.55	12662.0	Germany	2010-12-01	13
1101	536527	MULTI COLOUR SILVER T-LIGHT HOLDER	12	0.85	12662.0	Germany	2010-12-01	13
1102	536527	5 HOOK HANGER MAGIC TOADSTOOL	12	1.65	12662.0	Germany	2010-12-01	13
1103	536527	3 HOOK HANGER MAGIC GARDEN	12	1.95	12662.0	Germany	2010-12-01	13
1104	536527	5 HOOK HANGER RED MAGIC TOADSTOOL	12	1.65	12662.0	Germany	2010-12-01	13
1105	536527	ASSORTED COLOUR LIZARD SUCTION HOOK	24	0.42	12662.0	Germany	2010-12-01	13
1106	536527	JUMBO BAG WOODLAND ANIMALS	10	1.95	12662.0	Germany	2010-12-01	13
1107	536527	JUMBO BAG OWLS	10	1.95	12662.0	Germany	2010-12-01	13
1108	536527	HOT WATER BOTTLE BABUSHKA	4	4.65	12662.0	Germany	2010-12-01	13
1109	536527	HOMEMADE JAM SCENTED CANDLES	12	1.45	12662.0	Germany	2010-12-01	13

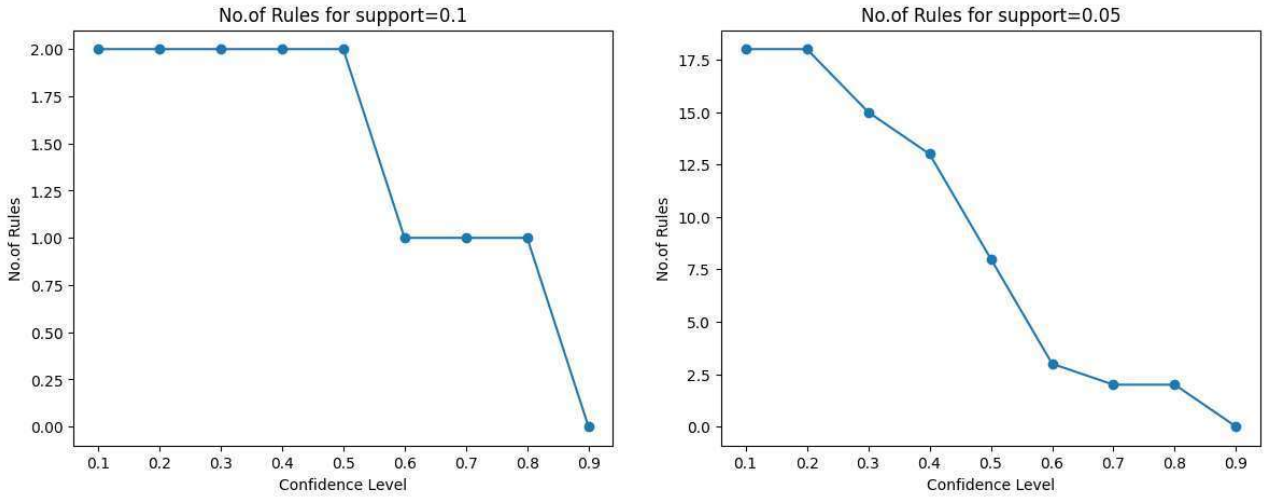
```
support=[0.1, 0.05, 0.01]
confidenceLevels=[0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1]

# Empty lists
rules_sup10=[0]*9
rules_sup5=[0]*9
rules_sup1=[0]*9


#rules for support 0.1
my_frequent_itemsets01 = apriori(my_basket, min_support=0.1, use_colnames=True)
for i in range(len(confidenceLevels)):
    rules_sup10[i]=len(association_rules(my_frequent_itemsets01, metric="confidence", min_thresh

#rules for support 0.05
my_frequent_itemsets005 = apriori(my_basket, min_support=0.05, use_colnames=True)
for i in range(len(confidenceLevels)):
    rules_sup5[i]=len(association_rules(my_frequent_itemsets005, metric="confidence", min_thresh

#rules for support 0.01
my_frequent_itemsets001 = apriori(my_basket, min_support=0.01, use_colnames=True)
for i in range(len(confidenceLevels)):
    rules_sup1[i]=len(association_rules(my_frequent_itemsets001, metric="confidence", min_thresh
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_
  and should_run_async(code)
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool typ
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool typ
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool typ
  warnings.warn(
```

```
# Plot the data
plt.figure(figsize=(22,5))
plt.subplot(1,3,1)
plt.plot(confidenceLevels,rules_sup10, marker='o', label='Support=0.1')
plt.title("No.of Rules for support=0.1")
plt.xlabel("Confidence Level")
plt.ylabel("No.of Rules")
plt.subplot(1,3,2)
plt.plot(confidenceLevels,rules_sup5, marker='o', label='Support=0.1')
plt.title("No.of Rules for support=0.05")
plt.xlabel("Confidence Level")
plt.ylabel("No.of Rules")
plt.subplot(1,3,3)
plt.plot(confidenceLevels,rules_sup1, marker='o', label='Support=0.1')
plt.title("No.of Rules for support=0.001")
```

```
plt.xlabel("Confidence Level")
plt.ylabel("No.of Rules")
plt.savefig("comparision")
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_c
  and should_run_async(code)
```



Let's analyze the results,

- Support level of 10% = We only identify a few rules with very low confidence levels. This means that there are no relatively frequent associations in our data set. We can't choose this value, the resulting rules are unrepresentative.
- Support level of 5% = We only identify a rule with a confidence of at least 50%. It seems that we have to look for support levels below 5% to obtain a greater number of rules with a reasonable confidence.
- Support level of 1% = We started to get dozens of rules

```
my_frequent_itemsets = apriori(my_basket, min_support=0.07, use_colnames=True)
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_
  and should_run_async(code)
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool typ
  warnings.warn(
```

```
myrules=association_rules(my_frequent_itemsets, metric="lift", min_threshold=1)
```

```
myrules
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_c
  and should_run_async(code)
```

	antecedents	consequents	antecedent support	consequent support	su
0	(ROUND SNACK BOXES SET OF4 WOODLAND)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.245077	0.137856	0.07
1	(PLASTERS IN TIN WOODLAND ANIMALS)	(ROUND SNACK BOXES SET OF4 WOODLAND)	0.137856	0.245077	0.07
2	(ROUND SNACK BOXES SET OF 4 FRUITS)	(ROUND SNACK BOXES SET OF4 WOODLAND)	0.157549	0.245077	0.1
3	(ROUND SNACK BOXES SET OF4 WOODLAND)	(ROUND SNACK BOXES SET OF 4 FRUITS)	0.245077	0.157549	0.1
4	(ROUND SNACK BOXES SET OF4 WOODLAND)	(SPACEBOY LUNCH BOX)	0.245077	0.102845	0.07
5	(SPACEBOY LUNCH BOX)	(ROUND SNACK BOXES SET OF4 WOODLAND)	0.102845	0.245077	0.07

```
mybasket= (data[data['Country'] =="France"]
          .groupby(['BillNo', 'Itemname'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('BillNo'))
```

```
mybasket
```

```
def my_encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1

my_basket = mybasket.applymap(my_encode_units)
my_basket.drop('POSTAGE', inplace=True, axis=1) #Remove "postage" as an item
### Display sample of set
my_basket
```

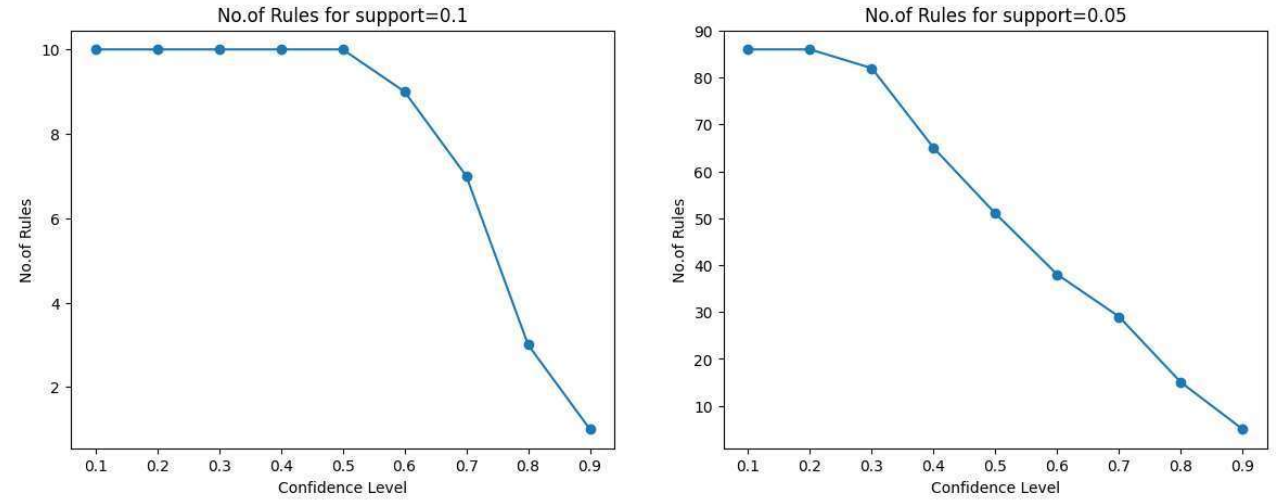
```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_c
  and should_run_async(code)
```

Itemname	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 EGG HOUSE PAINTED WOOD	12 MESSAGE CARDS WITH ENVELOPES	12 PENCIL SMALL TUBE WOODLAND	12 PENCILS SMALL TUBE RED RETROSPOT	12 PENCILS SMALL TUBE SKULL	12 PENCILS TALL TUBE POSY	12 PENCILS TALL TUBE RED RETROSPOT	12 PENCILS TALL TUBE WOODLAND	15CM CHRISTMAS GLASS BALL 20 LIGHTS	16 PIECE CUTLERY SET PANTRY DESIGN	DIS
BillNo													
536370	0	0	0	0	0	0	0	0	0	0	0	0	
536852	0	0	0	0	0	0	0	0	0	0	0	0	
536974	0	0	0	0	0	0	0	0	0	0	0	0	
537065	0	0	0	0	0	0	0	0	0	0	0	0	
537463	0	0	0	0	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
580986	0	0	0	0	0	0	0	0	0	0	0	0	
581001	0	0	0	0	0	0	0	0	0	0	0	0	
581171	0	0	0	0	0	0	0	0	0	0	0	0	
581279	0	0	0	0	0	0	0	0	0	0	0	0	
581587	0	0	0	0	0	0	0	0	0	0	0	0	

```
support=[0.1, 0.05, 0.01]
confidenceLevels=[0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1]

# Empty lists
rules_sup10=[0]*9
rules_sup5=[0]*9
rules_sup1=[0]*9

#rules for support 0.1
my_frequent_itemsets01 = apriori(my_basket, min_support=0.1, use_colnames=True)
for i in range(len(confidenceLevels)):
    rules_sup10[i]=len(association_rules(my_frequent_itemsets01, metric="confidence", min_thresh

#rules for support 0.05
my_frequent_itemsets005 = apriori(my_basket, min_support=0.05, use_colnames=True)
for i in range(len(confidenceLevels)):
    rules_sup5[i]=len(association_rules(my_frequent_itemsets005, metric="confidence", min_thresh

#rules for support 0.01
my_frequent_itemsets001 = apriori(my_basket, min_support=0.01, use_colnames=True)
for i in range(len(confidenceLevels)):
    rules_sup1[i]=len(association_rules(my_frequent_itemsets001, metric="confidence", min_thresh


# Plot the data
plt.figure(figsize=(22,5))
```

```
plt.subplot(1,3,1)
plt.plot(confidenceLevels,rules_sup10, marker='o', label='Support=0.1')
plt.title("No.of Rules for support=0.1")
plt.xlabel("Confidence Level")
plt.ylabel("No.of Rules")
plt.subplot(1,3,2)
plt.plot(confidenceLevels,rules_sup5, marker='o', label='Support=0.1')
plt.title("No.of Rules for support=0.05")
plt.xlabel("Confidence Level")
plt.ylabel("No.of Rules")
plt.subplot(1,3,3)
plt.plot(confidenceLevels,rules_sup1, marker='o', label='Support=0.1')
plt.title("No.of Rules for support=0.001")
plt.xlabel("Confidence Level")
plt.ylabel("No.of Rules")
plt.savefig("comparision")
#support =0.1 means if there are 100 trnascation altest 10 transcation will have that item
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_c
  and should_run_async(code)
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool type
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool type
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool type
  warnings.warn(
```




```
#lets chosse support=0.1
my_frequent_itemsets = apriori(my_basket, min_support=0.10, use_colnames=True)
myrules=association_rules(my_frequent_itemsets, metric="lift", min_threshold=1)
myrules
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_
  and should_run_async(code)
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool typ
  warnings.warn(
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(PLASTERS IN TIN WOODLAND ANIMALS)	(PLASTERS IN TIN CIRCUS PARADE)	0.170918	0.168367	0.102041	0.597015	3.545907	0.073264	2.063681	0.866000
1	(PLASTERS IN TIN CIRCUS PARADE)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.168367	0.170918	0.102041	0.606061	3.545907	0.073264	2.104592	0.863344
2	(PLASTERS IN TIN WOODLAND ANIMALS)	(PLASTERS IN TIN SPACEBOY)	0.170918	0.137755	0.104592	0.611940	4.442233	0.081047	2.221939	0.934634
3	(PLASTERS IN TIN SPACEBOY)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.137755	0.170918	0.104592	0.759259	4.442233	0.081047	3.443878	0.898687

Conclusion:

The conclusion of a market basket analysis for fresh product location improvement would depend on the specific findings and insights gained from the analysis .In summary, the conclusion of a market basket analysis for fresh product location improvement should provide actionable insights to optimize product placement, enhance the shopping experience, and increase sales of fresh items in a retailenvironment.