Unveiling Consumer Insights: Market Basket Analysis in E-Commerce through Machine Learning Algorithms

Astha Awale
Faculty of Business
Humber College
Toronto, Canada
astha.awale@gmail.com

Meutia Putri
Faculty of Business
Humber College
Toronto, Canada
Mvtiaree@gmail.com

Anjali Patel
Faculty of Business
Humber College
Toronto, Canada
anjali.patel199899@gmail.com

Sairohit Chowdhary
Faculty of Business
Humber College
Toronto, Canada
sairohitchowdhary@gmail.com

Abstract

In the fiercely competitive landscape of e-commerce, where paramount importance is placed on customer satisfaction, this study addresses the challenges faced by a saturated online retail platform. Leveraging association rule mining, it meticulously deciphers intricate customer purchasing behaviors embedded within transactional data, revealing hitherto concealed patterns among frequently co-acquired products. The research aims to empower strategic cross-selling initiatives, optimize precise product placements, and deliver personalized recommendations. Notably, the study incorporates user-based collaborative filtering for market basket analysis and recommendation systems. However, due to the inherent limitations of collaborative filtering, the study advocates the adoption of the Apriori algorithm for association rule mining. The potency of the Apriori algorithm in identifying upselling and cross-selling prospects is emphasized, where the algorithm's ability to prioritize impactful suggestions enhances customer engagement, augments transactional value, and elevates the overall shopping experience. In adeptly addressing the multifaceted dynamics of the e-commerce realm, this research provides actionable insights for informed decision-making in a competitive marketplace.

.

I. Introduction

In the realm of e-commerce, where competition is fierce and customer satisfaction is paramount, the pursuit of effective marketing strategies and an enhanced shopping experience stand as a strategic imperative. This study focuses on addressing these challenges within the context of an online retail platform grappling with a saturated market. Leveraging the power of association rule mining, the company aims to decipher intricate customer purchasing behaviors hidden within its transactional data. The outcome of this pursuit holds the promise of uncovering hidden patterns and relationships among the products consumers frequently acquire together. This endeavor aims to provide the company with actionable insights, paving the way for astute cross-selling opportunities, precision product placements, and personalized recommendations. Ultimately, the ultimate goal is twofold: to bolster sales figures and to augment the overall level of customer satisfaction. In summary, this study strives to fuse the dynamism of data-driven analysis with the potential of association rule mining, all in service of rejuvenating marketing strategies and refining customer experiences in the intricate landscape of e-commerce.

II. Literature Review

To thrive in this dynamic environment, retail enterprises must harness the power of data-driven insights to enhance their understanding of customer behaviors and preferences. This literature review delves into the application of association rule mining as a pivotal tool for extracting meaningful patterns from transactional data, ultimately facilitating the optimization of marketing strategies and the elevation of customer experiences.

Ayse Nur Sagin and Berk Ayvaz conducted a market basket analysis on yearly data from a "Do it yourself" (DIY) retailer to determine associated product pairs. They analyzed the effects of various promotional activities on these complementary product pairs and made recommendations based on the results. Additionally, they addressed the problem of major purchasing patterns not being extractable in enterprises with chain stores due to the assumption that products are always on store shelves at all times. They added warehouse and time information to the rules developed in similar studies and aimed to use the association rules in the development of marketing strategy, product supply, inventory, and distribution strategies for the entire store chain. [1]

Another study by Dr. Onur Dogan titled “A Recommendation System in E-Commerce with Profit-Support Fuzzy Association Rule Mining” gives information on the rise of e-commerce and has emphasized the importance of understanding complex transactional data. To provide effective product recommendations, a novel approach named Profit-supported Association Rule Mining with Fuzzy Theory (P-FARM) is introduced. By considering both profitability and frequency, P-FARM enhances e-commerce platforms' decision-making potential, offering valuable insights into customer preferences and maximizing business profits. [2]

III. Source Of The Data and Description Of The Dataset

In this study, our objective is to amass a dataset's suitability for conducting Market Basket analysis using Association Rules. Market Basket analysis is a well-established method in the retail and e-commerce sectors, aimed at identifying correlations between frequently co-purchased products. This approach reveals valuable insights like product affinities and customer purchasing patterns, which can be leveraged for improved cross-selling strategies and targeted marketing efforts.

The dataset that is sourced from Kaggle is comprehensive, encompassing transactional data, product particulars, and customer demographics. Specifically designed for Market Basket Analysis, this dataset contains pivotal attributes such as BillNo, Item name, Quantity, Date, Price, CustomerID, and Country. These attributes collectively empower us to uncover significant patterns in customer buying behavior and product relationships. Our exploration aims to gain insights into customer preferences, optimize product placement, and enhance the overall shopping journey. The dataset comprises 7 attributes and 522,065 rows.

1. BillNo: A unique 6-digit identifier assigned to each transaction, aiding in efficient tracking and reference.

2. Itemname: Represents the product involved in each transaction, serving as nominal variable denoting categories or names.

3. Quantity: Indicates the number of products sold in a transaction, represented as a numeric variable.

4. Date: Contains transaction generation timestamps, enabling time-related trend analysis.

5. Price: Reflects product prices, essential for calculating transaction totals and understanding product value.

6. CustomerID: A 5-digit identifier unique to each customer, categorizing customers without implying numeric relationships.

7. Country: Contains customer residence countries, treated as nominal data, signifying different categories (countries) devoid of inherent numerical meaning.

IV. Data Preprocessing

Data preprocessing is a crucial phase within the data analysis and machine learning process. When raw data is collected from diverse sources, it often contains mistakes, contradictions, and gaps, posing a challenge for obtaining valuable insights or constructing dependable models. The procedures of data cleaning and data preparation are undertaken to convert the raw data into a structured, accurate, and consistent format suitable for analysis.

Data cleaning refers to the procedure of detecting and rectifying errors, incongruities, and inaccuracies present in the dataset. It encompasses tasks like managing missing values, eliminating duplicates, rectifying data input errors, and managing outliers.

In contrast, data preparation concentrates on reshaping the cleaned data into a format that is appropriate for analysis or model development. This phase involves feature engineering, where new attributes may be generated, and existing attributes may be modified to extract more insightful information. Additionally, data normalization or scaling could be implemented to ensure that various attributes share a similar scale. The act of data preparation establishes the groundwork for precise and efficient data analysis, visualization, and model training.

1. Missing Values

By employing the data.isna().sum() function in Python, we can identify the presence of missing values in each column, as depicted in figure 1.1 below. The function allows us to observe the count of missing values in each column, enabling a comprehensive assessment of data quality.

Check for Missing Values

```
# Calculate the number of missing values for each column in the dataset
data.isna().sum()
```

```
BillNo             0
Itemname        1455
Quantity           0
Date               0
Price              0
CustomerID    134041
Country            0
dtype: int64
```

Fig 1.1

```
# Calculate the sum of 'Price' for rows where 'Itemname' is missing
data[data['Itemname'].isna()] ['Price'].sum()
```

```
0.0
```

To explore the data containing missing values in the 'Itemname' column, we can filter the dataset to show exclusively those rows. This subset of the data will offer valuable information about the
entries where the item names are absent.

```
# Filter the DataFrame to display rows where 'Itemname' is missing
data[data['Itemname'].isna()]
```

	BillNo	Itemname	Quantity	Date	Price	CustomerID	Country
613	536414	NaN	56	2010-12-01 11:52:00	0.0	NaN	United Kingdom
1937	536545	NaN	1	2010-12-01 14:32:00	0.0	NaN	United Kingdom
1938	536546	NaN	1	2010-12-01 14:33:00	0.0	NaN	United Kingdom
1939	536547	NaN	1	2010-12-01 14:33:00	0.0	NaN	United Kingdom
1940	536549	NaN	1	2010-12-01 14:34:00	0.0	NaN	United Kingdom
...	...	...	...	...	...	...	...
515623	581199	NaN	-2	2011-12-07 18:26:00	0.0	NaN	United Kingdom
515627	581203	NaN	15	2011-12-07 18:31:00	0.0	NaN	United Kingdom
515633	581209	NaN	6	2011-12-07 18:35:00	0.0	NaN	United Kingdom
517266	581234	NaN	27	2011-12-08 10:33:00	0.0	NaN	United Kingdom
518820	581408	NaN	20	2011-12-08 14:06:00	0.0	NaN	United Kingdom

1455 rows × 7 columns

Fig 1.2

After analyzing the data containing missing values in the 'Itemname' column as shown in Figure 1.2, it becomes apparent that these omissions do not offer any valuable insights. Since the item names are not present for these entries, it indicates that these instances may not significantly impact our analysis. Hence, we can treat these missing values as irrelevant and proceed with our analysis without including them.

Shifting the focus to the occurrence of missing values in the 'CustomerID' column, attention is directed towards investigating these omissions and discerning any potential problems or data quality concerns linked to them. Analyzing the repercussions of the absent 'CustomerID' values allows for an assessment of the dataset's completeness and reliability, facilitating informed decisions regarding the management or imputation of these data gaps. A more comprehensive exploration of this aspect is imperative to attain a thorough comprehension of the issues surrounding the absence of 'CustomerID' values.

```
# Select a random sample of 30 rows where 'CustomerID' is missing
data[data['CustomerID'].isna()].sample(30)
```

	BillNo	Itemname	Quantity	Date	Price	CustomerID	Country
240386	558904	CARD SUKI BIRTHDAY	3	2011-07-04 16:18:00	0.42	NaN	United Kingdom
242313	559052	4 IVORY DINNER CANDLES SILVER FLOCK	1	2011-07-05 16:53:00	3.29	NaN	United Kingdom
519719	581439	3 HEARTS HANGING DECORATION RUSTIC	2	2011-12-08 16:30:00	5.79	NaN	United Kingdom
101755	545216	BROCANTE COAT RACK	1	2011-02-28 16:50:00	19.96	NaN	United Kingdom
181590	553013	CAKE STAND VICTORIAN FILIGREE MED	1	2011-05-12 18:19:00	5.79	NaN	United Kingdom
412770	573553	COFFEE MUG CAT + BIRD DESIGN	1	2011-10-31 13:48:00	4.96	NaN	United Kingdom
52229	540848	6 RIBBONS RUSTIC CHARM	1	2011-01-12 09:26:00	3.36	NaN	United Kingdom
414577	573585	LAUNDRY 15C METAL SIGN	3	2011-10-31 14:41:00	2.46	NaN	United Kingdom
35339	539451	WOODEN FRAME ANTIQUE WHITE	1	2010-12-17 16:59:00	7.62	NaN	United Kingdom
359860	569372	STRAWBERRY CERAMIC TRINKET BOX	2	2011-10-03 16:04:00	2.46	NaN	United Kingdom
102765	545317	PINK FAIRY CAKE CHILDRENS APRON	1	2011-03-01 14:14:00	4.96	NaN	United Kingdom
65313	541827	PACK OF 12 PINK PAISLEY TISSUES	2	2011-01-21 17:05:00	0.83	NaN	United Kingdom
327131	566603	WOOD BLACK BOARD ANT WHITE FINISH	2	2011-09-13 16:12:00	16.63	NaN	United Kingdom

Fig 1.3

Upon meticulous examination of a subset of rows featuring absent 'CustomerID' values, as depicted in figure 1.3, there emerges no discernible pattern or explicit rationale accounting for their nonexistence. This observation indicates that the absence of 'CustomerID' entries does not stem from inadvertent oversights or systematic data anomalies. Rather, it is plausible that these omissions of values are intrinsic to the dataset, bereft of any prominent or underlying causation.

In spite of the significant prevalence of missing 'CustomerID' values, constituting a quarter of the entire dataset or 134,041 instances, the decision has been made against their exclusion. This choice is motivated by the recognition that such exclusion would entail a notable forfeiture of vital information.

2. Duplicate Values

In Python, the removal of duplicate values holds significance in maintaining data integrity and preventing potentially deceptive outcomes in analysis. Through the removal of duplicates, we preserve solely distinct occurrences, thereby augmenting the precision and dependability of our data intended for subsequent processing and analysis.

By employing the Python function data.duplicated().sum(), we have adeptly pinpointed a total of 5,286 instances characterized by duplicity within the dataset. To address this concern and uphold data precision, we can utilize the data.drop_duplicates() function, which provides a viable means to systematically eliminate these duplicated entries.

Duplicate Data

```
data.duplicated().sum()
```

```
5286
```

```
# Identify duplicate rows
duplicates = data.duplicated()
```

Fig 2.1

```
# Drop duplicate rows
data = data.drop_duplicates()
```

```
# Checking duplicate again
data.duplicated().sum()
```

```
0
```

Fig 2.2

3. Irrelevant and Incorrect Data

Upon conducting a more in-depth analysis, it became evident that a vast majority of transactions (approximately 93%) in the dataset originate from the UK. As a result, the 'Country' column may not add substantial diversity or variability to the analysis. Therefore, we can opt to eliminate the 'Country' column from the DataFrame df, signifying our intent to drop this attribute. This decision enables us to concentrate on other features that could potentially offer more valuable insights for our analysis.

Removing Incorrect & Irrelevant Data

```
# Print the number of unique countries in the 'Country' column
print("Number of unique countries:", data['Country'].nunique())

# Calculate and print the normalized value counts of the top 5 countries in the 'Country' column
print(data['Country'].value_counts(normalize=True)[:5])
```

```
Number of unique countries: 30
United Kingdom    0.933292
Germany           0.017517
France            0.016287
Spain             0.004813
Netherlands       0.004585
Name: Country, dtype: float64
```

Fig 3.1

```
# Delete the 'Country' column from the DataFrame
data.drop('Country', axis=1, inplace=True)
```

```
# Since BillNo datatype is object, it means that there are non digit values in it.
# Filter the DataFrame to display rows where 'BillNo' column contains non-digit values
data[data['BillNo'].str.isdigit() == False]
```

	BillNo	Itemname	Quantity	Date	Price	CustomerID
288772	A563185	Adjust bad debt	1	2011-08-12 14:50:00	11062.06	NaN
288773	A563186	Adjust bad debt	1	2011-08-12 14:51:00	-11062.06	NaN
288774	A563187	Adjust bad debt	1	2011-08-12 14:52:00	-11062.06	NaN

Fig 3.2

Furthermore, we have identified that the item name "Adjust bad debt" was mistakenly entered, as illustrated in Figures 3.1 & 3.2 As this entry does not offer any valuable information for our analysis, we can opt to remove the corresponding rows from the DataFrame. The provided code snippet filters the DataFrame df, retaining only the rows where the 'Itemname' column does not contain the value "Adjust bad debt." By performing this operation, we successfully eliminate the rows associated with the erroneous data entry, thereby ensuring the dataset is devoid of this irrelevant item name.

```
# Remove rows where the 'Itemname' column contains "Adjust bad debt"
data = data[data['Itemname'] != "Adjust bad debt"]
```

```
# Checking if all BillNo doesn't inculde letters
data['BillNo'].astype("int64")
```

```
0         536365
1         536365
2         536365
3         536365
4         536365
           ...
522059    581587
522060    581587
522061    581587
522062    581587
522063    581587
Name: BillNo, Length: 515320, dtype: int64
```

Fig 3.3

The existence of negative values within the dataset has captured our attention. In order to attain a thorough comprehension of this phenomenon, our attention is directed towards these specific occurrences. We aim to elucidate the fundamental causes behind them through this investigative process. This endeavor is anticipated to yield valuable perspectives into the characteristics of these negative values and their possible ramifications on our analysis, thereby exposing captivating narratives concealed within this facet of the data.

```
# Filter the DataFrame to display rows where 'Quantity' is less than 1
data[data['Quantity'] < 1]
```

	BillNo	Itemname	Quantity	Date	Price	CustomerID
7122	537032	?	-30	2010-12-03 16:50:00	0.0	NaN
12926	537425	check	-20	2010-12-06 15:35:00	0.0	NaN
12927	537426	check	-35	2010-12-06 15:36:00	0.0	NaN
12973	537432	damages	-43	2010-12-06 16:10:00	0.0	NaN
20844	538072	faulty	-13	2010-12-09 14:10:00	0.0	NaN
...	...	...	...	...	...	...
515634	581210	check	-26	2011-12-07 18:36:00	0.0	NaN
515636	581212	lost	-1050	2011-12-07 18:38:00	0.0	NaN
515637	581213	check	-30	2011-12-07 18:38:00	0.0	NaN
517209	581226	missing	-338	2011-12-08 09:56:00	0.0	NaN
519172	581422	smashed	-235	2011-12-08 15:24:00	0.0	NaN

473 rows × 6 columns

Fig 3.4

```
# Remove rows where 'Quantity' is less than 1
data = data[data['Quantity'] >= 1]
```

Fig 3.5

```
# Counting the number of rows where the price is zero
zero_price_count = len(data[data['Price'] == 0])
print("Number of rows where price is zero:", zero_price_count)

# Counting the number of rows where the price is negative
negative_price_count = len(data[data['Price'] < 0])
print("Number of rows where price is negative:", negative_price_count)
```

```
Number of rows where price is zero: 578
Number of rows where price is negative: 0
```

Fig 3.6

As evident from the visual representation in figure 3.4, instances of negative quantities could potentially be linked to system anomalies or irrelevant data for the scope of our analysis. Hence, a judicious course of action involves excluding these rows from the dataset. This step not only upholds the precision and dependability of our data but also eradicates any latent biases or misleading information stemming from negative quantities.

Shifting our focus to the 'Price' column, a comprehensive examination is conducted to unearth possible irregularities or anomalies. Through meticulous scrutiny of the data within this column, our objective is to identify any inconsistencies or outliers that might impinge upon the overall quality and integrity of the dataset. This meticulous scrutiny of the 'Price' column holds paramount importance in ensuring accurate and reliable pricing data for our study. A more detailed exploration of the 'Price' column is warranted to address any arising concerns.

The exploration of instances where products are offered at no cost, as illustrated in the subsequent figure 3.3, carries significant significance, as it has the potential to yield insights

into promotional endeavors, giveaways, or other distinctive facets within the dataset. By meticulously analyzing the data related to zero charges within the 'Price' column, we can glean deeper insights into these transactions and their potential impact on our analysis. Consequently, delving into the specifics of these transactions involving zero pricing is essential to uncover any noteworthy discoveries.

```
# Selecting a random sample of 20 rows where the price is zero
data[data['Price'] == 0].sample(20)
```

	BillNo	Itemname	Quantity	Date	Price	CustomerID
494772	579563	found	36	2011-11-30 11:41:00	0.0	NaN
186566	553521	DOORMAT WELCOME TO OUR HOME	1	2011-05-17 14:35:00	0.0	NaN
14061	537534	TV DINNER TRAY DOLLY GIRL	1	2010-12-07 11:48:00	0.0	NaN
461092	577144	found	96	2011-11-18 09:35:00	0.0	NaN
402034	572724	adjustment	4	2011-10-25 15:14:00	0.0	NaN
460917	577117	check	34	2011-11-17 18:04:00	0.0	NaN
408401	573293	adjustment	10	2011-10-28 15:14:00	0.0	NaN
100985	545160	SEASIDE FLYING DISC	1	2011-02-28 13:31:00	0.0	NaN
14081	537534	GLASS JAR KINGS CHOICE	1	2010-12-07 11:48:00	0.0	NaN
172113	552230	ASSTD MULTICOLOUR CIRCLES MUG	1	2011-05-06 15:43:00	0.0	NaN
233695	558340	CERAMIC STRAWBERRY MONEY BOX	2	2011-06-28 14:01:00	0.0	NaN
14050	537534	FRENCH BLUE METAL DOOR SIGN 1	3	2010-12-07 11:48:00	0.0	NaN
14055	537534	CHILDRENS GARDEN GLOVES BLUE	3	2010-12-07 11:48:00	0.0	NaN

Fig 3.7

After examining the sample of rows with a price of zero, we have discovered that these entries could potentially lead to misleading or inaccurate information for our analysis. Thus, it is advisable to eliminate these rows from the dataset to uphold the integrity and reliability of our analysis.

```
# Remove rows where the price is zero
data = data[data['Price'] != 0]
```

```
# Checking the number of rows where the price is zero
zero_price_count = len(data[data['Price'] == 0])
print("Number of rows where price is zero:", zero_price_count)
```

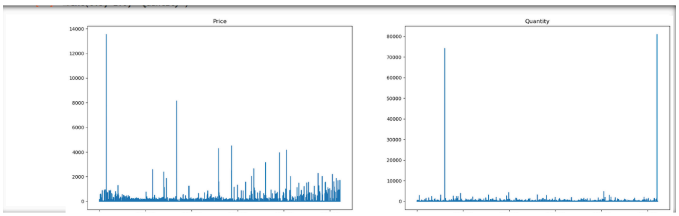
```
Number of rows where price is zero: 0
```

Fig 3.8

4. Outliers

```
In [28]: plt.figure(figsize=(22,7))
         plt.subplot(1,2,1)
         data.Price.plot()
         plt.title("Price")
         plt.subplot(1,2,2)
         data.Quantity.plot()
         plt.title("Quantity")
```

The easiest way to detect any outliers is to plot the graph, here in the above code we are plotting graphs for “price” and “quantity” to visually detect any outliers in the data.



Outliers are characterized as data points demonstrating exceptionally large (positive outliers) or unusually small (negative outliers) values in relation to the remainder of the dataset. These data instances are regarded as atypical and can be the result of diverse factors, including errors in measurement, inaccuracies in data entry, or occurrences of rare events.

As is visually apparent from the graphs, certain observations within the Price and Quantity variables manifest considerably elevated values that deviate significantly from the majority of the dataset. It is crucial to recognize that the presence of these items with high prices and quantities does not inherently label them as incorrect or unsuitable for inclusion in the analysis.

5. Data feature engineering:

```
# Calculate the total price by multiplying the quantity and price columns
data['Total_Price'] = data.Quantity * data.Price
data
```

This step adds a new column named “Total price” which is calculated through the multiplication of Quantity and price Column. This step is crucial as we can compare the sales more effectively and easily.

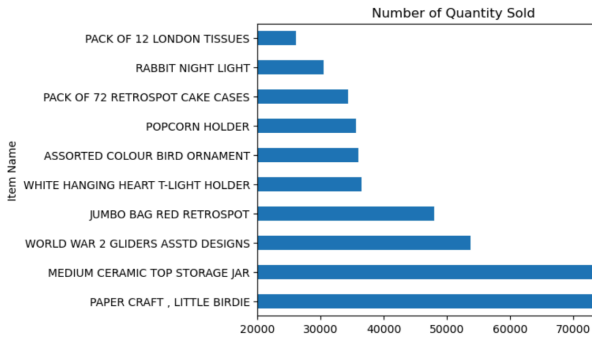
	BillNo	Itemname	Quantity	Date	Price	CustomerID	Total_Price
0	536365	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	15.30
1	536365	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	20.34
2	536365	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	22.00
3	536365	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	20.34
4	536365	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	20.34
...	...	...	...	...	...	...	...
522059	581587	PACK OF 20 SPACEBOY NAPKINS	12	2011-12-09 12:50:00	0.85	12680.0	10.20
522060	581587	CHILDREN'S APRON DOLLY GIRL	6	2011-12-09 12:50:00	2.10	12680.0	12.60
522061	581587	CHILDRENS CUTLERY DOLLY GIRL	4	2011-12-09 12:50:00	4.15	12680.0	16.60
522062	581587	CHILDRENS CUTLERY CIRCUS PARADE	4	2011-12-09 12:50:00	4.15	12680.0	16.60
522063	581587	BAKING SET 9 PIECE RETROSPOT	3	2011-12-09 12:50:00	4.95	12680.0	14.85

514847 rows × 7 columns

6. Exploratory Data Analysis:

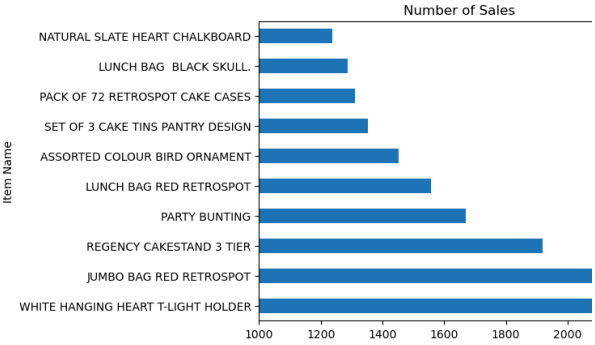
```
# Plotting the top 10 most sold products by quantity
data.groupby('Itemname')['Quantity'].sum().sort_values(ascending=False)[:10].plot(kind='barh', title='Number of Quantity Sold')
plt.ylabel('Item Name')
plt.xlim(20000, 82000)
plt.show()
```

To determine the distribution of the Number of Quantity sold and Item name with respect to each Exploratory data analysis in the form of a bar graph is done.




```
# Plotting the top 10 most sold products by transaction
d['Itemname'].value_counts(ascending=False)[:10].plot(kind='barh', title='Number of Sales
plt.ylabel('Item Name')
plt.xlim(1000, 2300)
plt.show()
```

Here, we have plotted a bar graph of ”sales” with respect to “Item name” to determine which product is most selling throughout the data.



Upon examining the plots, a compelling deduction becomes apparent, indicating that specific products exhibit a notably higher frequency of sales (count) compared to others, despite recording relatively lower quantities sold per individual transaction. This intriguing observation suggests the presence of items that are frequently purchased in considerable quantities during a single instance. These products might encompass those frequently acquired in bulk or those typically packaged and sold in larger amounts.

This insight underscores the critical significance of incorporating both the quantity sold and the sales count when engaging in an in-depth analysis of product popularity and demand. It implies that certain items might display heightened turnover rates due to frequent purchases, while others could showcase greater quantities per sale, leading to distinct sales trends and consumer behaviors. Gaining a comprehensive comprehension of these dynamics holds immense value in optimizing inventory management, formulating effective pricing strategies, and recognizing and catering to customer preferences.

Harnessing this knowledge empowers businesses to make well-informed decisions, ultimately maximizing profitability, elevating customer satisfaction, and cultivating enduring success within the ever-evolving market landscape.

7. Validation split:

As we will be employing unsupervised learning techniques, the need to partition the data into test and validation sets is not obligatory. The chosen algorithm for this study is the Apriori Algorithm, which is categorized under the domain of unsupervised learning.

V. Implementation

1. Apriori Algorithm for association rule generation

The Apriori algorithm is used to generate association rules, serving as a prominent method for unveiling captivating relationships or affiliations within datasets' item compositions. In particular, this algorithm finds extensive application within the domain of market basket analysis, aimed at discerning connections between items frequently purchased together. [3]

The result of association rules offers insights into the relationship among diverse items within the dataset. Each established association rule comprises a dual structure, encompassing the antecedent, positioned on the left-hand side, and the consequent, situated on the right-hand side. The antecedent encapsulates the item(s) or itemset(s) serving as conditions or foundational premises, whereas the consequent represents the item(s) or itemset(s) predicted or deduced from the antecedent. [4]

The minimum support threshold plays a pivotal role in the Apriori algorithm by determining what constitutes a "frequent" itemset. Selecting an excessively low-value results in a surplus of items being engaged in the rule-generation process. Conversely, opting for an exceedingly high-value results in fewer items being included, resulting in a significant loss of information. After the removal of duplicates, the dataset consists of 515,323 entries in total, and a minimum support threshold of 0.01 (1%) was chosen, which implies that an itemset must appear in at least 1% of the transactions or around 5,143 transactions to be considered significant or frequent. Setting a minimum support threshold serves as a filtering mechanism to focus the algorithm's attention on itemsets that exhibit a meaningful level of co-occurrence. In a large dataset with hundreds of thousands of entries, not all combinations of items are equally important or informative. A low support threshold, like 0.01, ensures that the algorithm captures itemsets that are somewhat prevalent but not excessively common. This threshold strikes a balance between finding relevant associations while avoiding overfitting the data with trivial itemsets that may not carry meaningful insights. [5]

The minimum confidence threshold comes into play during the generation of association rules. A confidence threshold of

0.5 (50%) indicates that an association rule will be considered significant only if the consequent item(s) appear with a minimum confidence of 50% when the antecedent item(s) are present in the transaction.

In a dataset with 515,323 entries, a minimum confidence threshold of 0.5 is chosen to ensure that the generated association rules are substantial and practically actionable. A confidence of 50% means that whenever the antecedent items are bought, there is at least a 50% likelihood that the consequent items will also be purchased. This threshold filters out weak or unreliable associations, allowing the algorithm to focus on rules that demonstrate a substantial correlation between items.

The choice of a minimum support threshold of 0.01 (1%) and a minimum confidence threshold of 0.5 in a dataset with 515,323 entries reflects a careful balance between capturing meaningful patterns in the data and avoiding spurious or trivial associations. These thresholds ensure that the Apriori algorithm identifies frequent itemsets with a reasonable occurrence rate and generates association rules with a significant level of confidence, contributing to more accurate and actionable insights.

2. Collaborative Filtering for Product Recommendation

Collaborative Filtering (CF) is a prominent paradigm within recommendation systems, driven by the fundamental principle that users who exhibit similar behaviors possess shared preferences. CF leverages collective intelligence to offer personalized recommendations, thereby enhancing user engagement and satisfaction in various domains, including e-commerce, online content, and social networks. Unlike content-based methods that rely on item attributes, collaborative filtering hinges on the premise that users who have displayed similar behaviors in the past are likely to manifest analogous preferences in the future. This intrinsic human tendency serves as the foundational basis for collaborative filtering's efficacy in providing tailored recommendations.

At its core, CF relies on the analysis of user-item interactions, commonly represented in a sparse matrix where rows denote users and columns represent items. The matrix encapsulates historical behaviors such as purchases, ratings, or clicks, serving as a canvas upon which user preferences are unveiled. By assessing the similarity between users based on their interaction profiles, CF unveils latent affinities, mirroring the phenomenon of social influence.

Cosine Similarity: One prevalent metric employed in CF is cosine similarity. This measure quantifies the angle between two user vectors in the interaction matrix, reflecting the degree of alignment in their preferences. A higher cosine similarity indicates greater congruence in user behaviors, thereby establishing a foundation for recommendation generation [6].

Neighborhood-Based Approach: CF often adopts a neighborhood-based approach, identifying a subset of users similar to a target user. This neighborhood comprises users whose interactions bear a resemblance to the target user's activities. Recommendations are then generated by aggregating items favored by the neighboring users. Items that have garnered attention and engagement from analogous users emerge as potential recommendations. The recommendations are rooted in the principle of social influence, where the actions of one user serve as a proxy for guiding another user's choices. Notably, this approach reflects the social dynamics where users are influenced by their peers. The resultant list of recommendations is often ranked according to the level of consensus among similar users, amplifying the accuracy and relevance of suggestions.[7]

User-Item Matrix Factorization: An advanced technique in CF involves matrix factorization, where the interaction matrix is decomposed into latent factors. By capturing latent dimensions underlying user preferences, this approach overcomes data sparsity and offers enhanced prediction accuracy [8].

CF is not without challenges. The "cold start" problem arises when new users or items lack sufficient interactions, impeding accurate recommendations. Additionally, the "filter bubble" effect, where users are exposed to limited content, may hinder serendipitous discovery. To address these, hybrid approaches integrating CF with content-based filtering have emerged [9].

VI. Model Evaluation

1. Evaluation for Apriori Algorithm

When analyzing association rules with the Apriori Algorithm, various evaluation metrics are employed to understand the significance and strength of the relationships between different items or itemsets. These metrics offer insights into the co-occurrence patterns and predictive power of these associations. Support, a fundamental metric, reveals the proportion of transactions where both the antecedent and consequent items are present. Confidence, on the other hand, quantifies the conditional probability of observing the consequent given the antecedent, providing an understanding of how reliably one item predicts another's occurrence. Lift, a pivotal measure, compares the actual co-occurrence to what would be expected if the items were independent, thus gauging the strength of association. Leverage computes the

difference between observed and expected co-occurrence, signaling the degree of association beyond randomness. Conviction offers insight into the dependency between antecedent and consequent, with higher values indicating stronger dependency. Lastly, Zhang's Metric combines support and confidence to comprehensively assess the rule's strength.[10] These metrics collectively aid in distinguishing meaningful associations from coincidental occurrences, facilitating informed decision-making in various domains.

```
In [94]: rules = rules.sort_values(['confidence', 'lift'], ascending =[False, False])
         rules
```

Out[94]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
17	(BEADED CRYSTAL HEART PINK ON STICK)	(DOTCOM POSTAGE)	0.011471	0.039314	0.011193	0.975728	24.818876	0.010742	39.580265	0.970845
613	(HERB MARKER CHIVES, HERB MARKER THYME)	(HERB MARKER PARSLEY)	0.010413	0.012919	0.010079	0.967914	74.921584	0.009945	30.764023	0.997035
607	(HERB MARKER ROSEMARY, HERB MARKER CHIVES)	(HERB MARKER PARSLEY)	0.010358	0.012919	0.010023	0.967742	74.908231	0.009890	30.599510	0.996977
619	(HERB MARKER ROSEMARY, HERB MARKER CHIVES)	(HERB MARKER THYME)	0.010358	0.012919	0.010023	0.967742	74.908231	0.009890	30.599510	0.996977
1218	(HERB MARKER ROSEMARY, HERB MARKER BASIL, HERB...	(HERB MARKER THYME)	0.010580	0.012919	0.010190	0.963158	74.553403	0.010054	26.792198	0.997137
...	...	...	...	...	...	...	...	...	...	...

These results are from the generated association rules based on the dataset. Each rule specifies antecedents (items that are "if" or "when" part) and consequents (items that are "then" part). The metrics like support, confidence, lift, and others provide insights into the strength and significance of these associations.

The provided code snippet arranges the generated association rules in a specific order based on two criteria: confidence and lift. The rules are sorted in descending order for both confidence and lift. This arrangement aims to prioritize rules with higher confidence values and stronger lift values, which indicates a substantial relationship between the antecedent and consequent items in the rule.

The association rule analysis has yielded an insightful result highlighting a strong and intriguing connection between the items involved. The rule demonstrates that when customers purchase the item "BEADED CRYSTAL HEART PINK ON STICK," they are remarkably likely to also acquire "DOTCOM POSTAGE," as indicated by the high confidence of 97.57%. This implies that almost all instances of the former item's purchase are accompanied by the latter. The lift value of 24.824 further underscores the strength of this relationship, showcasing that the occurrence of "DOTCOM POSTAGE" is significantly amplified when the antecedent item is present. The positive leverage value of 0.010740 confirms that the association is beyond random chance. Moreover, the conviction score of 39.580626 suggests a substantial degree of dependency between the two items, and Zhang's Metric of 0.970851 attests to the robustness of this association in terms of both support and confidence. In essence, these metrics collectively emphasize the substantial co-occurrence, predictability, and non-random nature of the relationship between "BEADED CRYSTAL HEART PINK ON STICK" and "DOTCOM POSTAGE," shedding light on a potentially strategic pairing for marketing and sales efforts.

Similarly, the subsequent rules also follow the same pattern, showing the confidence, lift, and other metrics for each association rule, giving insights into the relationships and patterns identified by the Apriori algorithm.

In order to extract more valuable insights, we refine our focus by narrowing down the set of generated association rules to specifically highlight those that signify cross-selling opportunities. This is achieved by filtering rules where both the antecedent and consequent consist of individual items. Subsequently, to prioritize the most robust and reliable recommendations, we sort these cross-selling rules based on high levels of confidence and support. The confidence metric reflects the likelihood of observing the consequent given the antecedent, while support quantifies the frequency of occurrence of both items together. By sorting in descending order, we emphasize rules with the strongest backing. Finally, from this refined and sorted collection, we pinpoint the top 5 recommendations that exhibit the greatest potential for effective cross-selling strategies. This approach ensures that the identified opportunities hold significant promise for enhancing sales and customer satisfaction through targeted cross-selling initiatives. The snippet of the code can be seen below.

```
# Filter association rules for cross-selling opportunities
cross_selling_rules = rules[(rules['antecedents'].apply(len) == 1) & (rules['consequents'].apply(len) == 1)]

# Sort rules based on confidence and support
cross_selling_rules = cross_selling_rules.sort_values(by=['confidence', 'support'], ascending=False)

# Select top cross-selling recommendations
top_cross_selling = cross_selling_rules.head(5)
```

```
# Display cross-selling recommendations
print("Cross-Selling Recommendations:")
for idx, row in top_cross_selling.iterrows():
    antecedent = list(row['antecedents'])[0]
    consequent = list(row['consequents'])[0]
    print(f"Customers who bought '{antecedent}' also bought '{consequent}'.")
```

```
Cross-Selling Recommendations:
Customers who bought 'BEADED CRYSTAL HEART PINK ON STICK' also bought 'DOTCOM POSTAGE'.
Customers who bought 'HERB MARKER THYME' also bought 'HERB MARKER ROSEMARY'.
Customers who bought 'HERB MARKER ROSEMARY' also bought 'HERB MARKER THYME'.
Customers who bought 'HERB MARKER CHIVES' also bought 'HERB MARKER PARSLEY'.
Customers who bought 'REGENCY TEA PLATE PINK' also bought 'REGENCY TEA PLATE GREEN'.
```

The outcomes derived from the Apriori algorithm present valuable insights for cross-selling recommendations within the retail domain. The discovered association rules offer actionable suggestions for enhancing customer purchasing experiences. For instance, the analysis reveals that customers who purchased the item 'BEADED CRYSTAL HEART PINK ON STICK' are highly inclined to also acquire 'DOTCOM POSTAGE.' This presents an opportunity to bundle these items together or suggest 'DOTCOM POSTAGE' during the purchase of the former, potentially increasing the likelihood of multiple purchases. Similarly, the observed pattern where customers buying 'HERB MARKER THYME' are likely to purchase 'HERB MARKER ROSEMARY,' and vice versa, underscores the feasibility of recommending complementary herb marker sets. Likewise, the correlation between 'HERB MARKER CHIVES' and 'HERB MARKER PARSLEY' indicates a potential pairing for customers seeking a variety of herb markers. Lastly, the connection between 'REGENCY TEA PLATE PINK' and 'REGENCY TEA PLATE GREEN' opens avenues for suggesting coordinated sets of tea plates,

catering to customers' preferences for color-coordinated selections. These cross-selling recommendations, informed by the association rules, can not only enhance customer satisfaction but also contribute to increased sales and improved market strategies.

To further enhance the depth of our insights, we narrow our attention to pinpoint potential opportunities for upselling. This is achieved through a meticulous filtering process where we isolate association rules involving a scenario in which a customer's purchase of a single item could be leveraged to recommend or facilitate the sale of multiple items. This strategic approach is geared towards encouraging customers to consider complementary or additional products, thereby increasing the overall transaction value.

Subsequent to this filtering, the rules are subjected to a sorting procedure based on two critical factors: confidence and support. Confidence signifies the likelihood of the suggested items being purchased alongside the initial item, while support quantifies the frequency of this co-purchase pattern across transactions. By organizing the rules in descending order of these metrics, we prioritize the most robust and reliable recommendations.

The culmination of this process involves selecting the top 5 recommendations that exhibit the greatest potential for successful upselling endeavors. These recommendations represent instances where the evidence of customer behavior strongly suggests the viability of offering additional products, resulting in higher transaction values and enhanced customer satisfaction. The snippet of the code can be seen below.

```
# Filter association rules for upselling opportunities
upselling_rules = rules[(rules['antecedents'].apply(len) == 1) & (rules['consequents'].apply(len) > 1)]

# Sort rules based on confidence and support
upselling_rules = upselling_rules.sort_values(by=['confidence', 'support'], ascending=False)

# Select top upselling recommendations
top_upselling = upselling_rules.head(5)
```

```
top_upselling = upselling_rules.sort_values(['confidence', 'support'],
                                            ascending=False).drop_duplicates('antecedents')[:5]
print("\nUpselling Recommendations:")
for idx, row in top_upselling.iterrows():
    antecedent = list(row['antecedents'])[0]
    consequents = list(row['consequents'])
    print(f"For customers who bought '{antecedent}', recommend the following upgrades: {', '.join(consequents)}.")
```

```
Upselling Recommendations:
For customers who bought 'HERB MARKER CHIVES', recommend the following upgrades: HERB MARKER THYME, HERB MARKER PARSL
EY.
For customers who bought 'HERB MARKER THYME', recommend the following upgrades: HERB MARKER ROSEMARY, HERB MARKER PAR
SLEY.
For customers who bought 'HERB MARKER PARSLEY', recommend the following upgrades: HERB MARKER ROSEMARY, HERB MARKER T
HYME.
For customers who bought 'HERB MARKER ROSEMARY', recommend the following upgrades: HERB MARKER THYME, HERB MARKER PAR
SLEY.
For customers who bought 'REGENCY TEA PLATE PINK', recommend the following upgrades: REGENCY TEA PLATE GREEN, REGENCY
TEA PLATE ROSES.
```

Based on the insights extracted from the Apriori algorithm, we have formulated a series of strategic upselling recommendations that cater to varying customer preferences. These recommendations are designed to encourage customers to consider enhancing their initial purchases by suggesting complementary or upgraded items that align with their preferences. Specifically, we have identified key product pairs for which customers have displayed a propensity to engage in cross-purchasing behavior. For instance, customers who have chosen 'HERB MARKER CHIVES' are encouraged to explore the upgrades of 'HERB MARKER THYME' and 'HERB MARKER PARSLEY.' Similarly, we propose complementary choices for customers who bought 'HERB MARKER THYME,' 'HERB MARKER PARSLEY,' 'HERB MARKER ROSEMARY,' and 'REGENCY TEA PLATE PINK.' By recommending items that seamlessly integrate with their initial selections, we aim to provide a tailored shopping experience that aligns with their preferences. Ultimately, these upselling recommendations not only contribute to a more personalized engagement but also have the potential to increase customer satisfaction and overall transaction value.

To summarize, the application of the Apriori algorithm has yielded substantial insights into both upselling and cross-selling opportunities within the dataset. By meticulously analyzing the association rules, we have identified valuable recommendations that empower effective strategies for encouraging customers to consider complementary or upgraded purchases. The algorithm's filtering mechanisms have allowed us to pinpoint precise scenarios where such recommendations hold significant potential. Through rigorous sorting based on confidence and support metrics, we have prioritized the most impactful suggestions, ensuring strategic decision-making. This comprehensive approach has enabled us to not only identify customers who could benefit from cross-selling but also propose tailored upgrades that align with their preferences. The algorithm's performance in uncovering these opportunities showcases its proficiency in capturing meaningful patterns from transaction data. Ultimately, the Apriori algorithm has provided a robust framework for enhancing customer engagement, driving transaction value, and refining the overall shopping experience.

2. Evaluation for Collaborative Filtering

```
# Make top 5 recommendations for a specific user
user_id = 17850.0
recommendations = recommend_items(user_id)
print("Recommended Items:")
print(recommendations.head(5))
```

```
Recommended Items:
Itemname
MEDIUM CERAMIC TOP STORAGE JAR    5
SMALL CERAMIC TOP STORAGE JAR     2
10 COLOUR SPACEBOY PEN            1
WHITE FRANGIPANI NECKLACE         1
REX CASH+CARRY JUMBO SHOPPER      1
dtype: int64
```

The generated output presents the results of a collaborative filtering approach.

Upon analyzing a specific user's historical transactions, the system identifies users who exhibit analogous purchasing tendencies. The algorithm quantifies the resemblance between users by computing the cosine similarity, a measure of the

cosine of the angle between their transaction vectors. This cosine similarity value serves as an indication of the degree of similarity between users, forming the basis for generating recommendations.

The recommendation system subsequently selects a subset of users with the highest cosine similarity to the target user. This selection aims to capture those individuals whose purchase history closely aligns with that of the user under consideration. The items purchased by these similar users are then aggregated, and the system recommends items that have been frequently acquired by this cohort of similar users.

In the provided output, the system has identified several items that were frequently purchased by users closely resembling the target user. Each recommended item is accompanied by a numerical score, indicating the frequency of its occurrence among the suggested group of similar users. The score signifies the system's confidence in the recommendation, with higher scores reflecting greater consensus among similar users regarding the appeal of the item.

For instance, the output indicates that the "MEDIUM CERAMIC TOP STORAGE JAR" is the most prevalent recommendation, appearing in the purchase histories of multiple users akin to the target user. Similarly, other items such as the "SMALL CERAMIC TOP STORAGE JAR," "10 COLOUR SPACEBOY PEN," "WHITE FRANGIPANI NECKLACE," and the "REX CASH+CARRY JUMBO SHOPPER" are also suggested based on their frequent acquisition by the identified group of similar users.

It is important to note that while the presented output provides valuable insights into potential recommendations, the collaborative filtering approach can be further refined and augmented with additional factors such as user preferences, ratings, and contextual information to enhance the accuracy and personalization of the recommendations. The application of collaborative filtering, as demonstrated by this output, serves as a foundational step toward building an effective and user-centric recommendation system.

VII. Conclusion

In conclusion, our investigation into enhancing cross-selling and upselling strategies within the context of an e-commerce platform underscores the remarkable effectiveness of the Apriori algorithm. As we navigated the intricate realm of customer purchasing behaviors and transactional patterns, the Apriori algorithm emerged as a potent tool for uncovering hidden associations and optimizing product recommendations. Through its adept mining of association rules, the algorithm not only provided actionable insights into frequently co-acquired products but also prioritized those insights based on their potential impact on customer engagement and transactional value.

While our study acknowledges the value of user-based collaborative filtering in market basket analysis and recommendation systems, the Apriori algorithm surpassed its limitations, offering a scalable and efficient solution tailored to the unique demands of the e-commerce landscape. Its ability to navigate vast datasets and efficiently generate high-confidence association rules positions it as a preferred approach for generating cross-selling and upselling opportunities.

The insights derived from this study hold practical implications for businesses seeking to optimize their product placement, drive customer engagement, and enhance overall profitability. By harnessing the power of the Apriori algorithm, e-commerce companies can confidently design and implement cross-selling and upselling strategies that align with customer preferences, thereby enriching the shopping experience and achieving their revenue objectives. As the e-commerce industry continues to evolve, the Apriori algorithm stands as a robust and dependable tool for uncovering the intricate web of consumer preferences and driving effective marketing strategies.

References

[1] Southeast Europe Journal of Soft Computing. Determination of Association Rules with Market Basket Analysis: Application in the Retail Sector. Ayse Nur Sagin and Berk Ayvaz
http://scjournal.ius.edu.ba/index.php/scjournal/article/view/149

[2] MDPI. A Recommendation System in E-Commerce with Profit-Support Fuzzy Association Rule Mining (P-FARM). Dr. Onur Dogan. https://www.mdpi.com/0718-1876/18/2/43

[3] Y. Lim, “Data Mining: Market Basket Analysis with Apriori algorithm,” Medium, https://towardsdatascience.com/data-mining-market-basket-analysis-with-apriori-algorithm-970ff256a92c

[4] B. Lutkevich, “What are association rules in Data Mining? definition from TechTarget,” Business Analytics, https://www.techtarget.com/searchbusinessanalytics/definition/association-rules-in-data-mining

[5] E. Hikmawati, N. U. Maulidevi, and K. Surendro, “Minimum threshold determination method based on dataset characteristics in Association Rule Mining,” *Journal of Big Data*, vol. 8, no. 1, 2021. doi:10.1186/s40537-021-00538-3

[6] Resnick, P., & Varian, H. R. (1997). Recommender systems. Communications of the ACM, 40(3).

[7] Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1).

[8] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8).

[9] Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734-749.

[10] B. Naibei, “Getting started with Apriori algorithm in Python,” Section, https://www.section.io/engineering-education/apriori-algorithm-in-python/ (accessed Aug. 9, 2023).