Project Title : # Market Basket Insights

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

Phase 4 : ## Development Part – 2

Continue building the market basket insights project by performing association analysis and generating insights .

Introduction :

Continuing the development of the Market Basket Insights project involves leveraging association analysis techniques to uncover valuable patterns and relationships within customer purchase data.

By analyzing the contents of market baskets, we can generate insights that will inform decision-making, improve marketing strategies, and enhance the overall shopping experience. In this phase, we will explore various association rules, like frequent itemsets and support-confidence measures, to identify which products tend to be purchased together and the strength of these associations.

The ultimate goal is to provide actionable insights that can drive product recommendations, optimize inventory management, and maximize cross-selling opportunities. This project promises to unlock a deeper understanding of customer behavior and empower businesses to make data-driven decisions for improved profitability and customer satisfaction.

Association Rules :

Association rules are generated using the Apriori algorithm, which is a popular algorithm for discovering interesting relationships or associations among items in a dataset. Association rule mining is commonly used in market basket analysis, where the goal is to find associations between items frequently purchased together.

The generated association rules provide insights into the relationships between different items or itemsets in the dataset. Each association rule consists of two parts: the antecedent (or left-hand side) and the consequent (or right-hand side). The antecedent represents the item(s) or itemset(s) that act as a condition or premise, while the consequent represents the item(s) or itemset(s) that are predicted or inferred from the antecedent.

The association rules are evaluated based on different metrics, such as support, confidence, lift, leverage, and conviction. These metrics provide measures of the interestingness or strength of the rules.

- Support measures the proportion of transactions in the dataset that contain both the antecedent and the consequent.
- Confidence measures the conditional probability of the consequent given the antecedent.
- Lift measures the ratio of observed support to expected support, indicating the strength of the association between the antecedent and the consequent.
- Leverage measures the difference between the observed support and the expected support, indicating the significance of the association.
- Conviction measures the ratio of the expected confidence to the observed confidence, indicating the degree of dependency between the antecedent and the consequent.

By examining the association rules, you can identify interesting relationships, co-occurrences, or patterns among items, which can be used for various purposes such as product recommendation, market segmentation, or inventory management.

To generate the association rules, we use the Apriori algorithm with a minimum support threshold of 0.05 (5%). This ensures that only itemsets with sufficient frequency in the dataset are considered.

Let's explore the generated association rules:

```
# Assign the original DataFrame to df2

df2 = df

# Filter rows based on item occurrences
item_counts = df2['Itemname'].value_counts(ascending=False)
filtered_items = item_counts.loc[item_counts > 1].reset_index()['index']
df2 = df2[df2['Itemname'].isin(filtered_items)]

# Filter rows based on bill number occurrences
bill_counts = df2['BillNo'].value_counts(ascending=False)
filtered_bills = bill_counts.loc[bill_counts > 1].reset_index()['index']
df2 = df2[df2['BillNo'].isin(filtered_bills)]
```

Filtering is done based on item occurrences:

The frequency count of each unique item name in the 'Itemname' column is calculated and stored in item_counts.

filtered_items is created by filtering item_counts to retain only item names that occur more than once.

Rows in df2 are filtered to keep only those where the item name in the 'Itemname' column is present in the filtered_items list.

Filtering is done based on bill number occurrences:

The frequency count of each unique bill number in the 'BillNo' column is calculated and stored in bill_counts.

filtered_bills is created by filtering bill_counts to retain only bill numbers that occur more than once.

Rows in df2 are filtered to keep only those where the bill number in the 'BillNo' column is present in the filtered_bills list.

After executing the code, the filtered DataFrame df2 will contain only the rows where both the item name and bill number occur more than once in the original df.

```
# Create a pivot table using the filtered DataFrame
pivot_table = pd.pivot_table(df2[['BillNo','Itemname']], index='BillNo', columns='Itemname', aggfunc=lambda x: True, fill_value=False)
```

The code creates a pivot table that represents the occurrence of items in bills. The pivot table provides a binary representation where each cell indicates whether a specific item appears in a particular bill. Here's how it works:

The original DataFrame df2 contains information about bills and corresponding item names.

By using the pd.pivot_table() function, we reshape the DataFrame to create a pivot table.

The pivot table has 'BillNo' as the index and 'Itemname' as the columns, grouping the data based on these two columns.

The goal is to determine whether a specific item appears in a particular bill.

Each cell in the pivot table is filled with either True or False:

If an item appears in a bill, the corresponding cell is marked as True.

If an item does not appear in a bill, the corresponding cell is marked as False.

This binary representation of item occurrence in bills allows us to easily analyze and identify patterns or associations between different items and bills.

The resulting pivot table provides a concise summary of the occurrence of items in bills, which can be used for various purposes such as market basket analysis, recommendation systems, or identifying frequent itemsets and association rules.

```
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

# Generate frequent itemsets with minimum support of 0.1 (10%)
```

```
frequent_itemsets = apriori(pivot_table, min_support=0.01,use_colnames=True)

# Generate association rules
rules = association_rules(frequent_itemsets, "confidence", min_threshold = 0.5)

# Print frequent itemsets
print("Frequent Itemsets:")
print(frequent_itemsets)

# Print association rules
print("\nAssociation Rules:")
rules
```

```
Frequent Itemsets:
       support                                           itemsets
0     0.017370                        (10 COLOUR SPACEBOY PEN)
1     0.013751                (12 MESSAGE CARDS WITH ENVELOPES)
2     0.019653                  (12 PENCIL SMALL TUBE WOODLAND)
3     0.019820            (12 PENCILS SMALL TUBE RED RETROSPOT)
4     0.019597                     (12 PENCILS SMALL TUBE SKULL)
...        ...                                                ...
2467  0.010355  (LUNCH BAG RED RETROSPOT, LUNCH BAG SUKI DESIG...
2468  0.010188  (LUNCH BAG RED RETROSPOT, LUNCH BAG SUKI DESIG...
2469  0.010300  (LUNCH BAG RED RETROSPOT, LUNCH BAG SPACEBOY D...
2470  0.010467  (LUNCH BAG RED RETROSPOT, LUNCH BAG PINK POLKA...
2471  0.011302  (CHARLOTTE BAG PINK POLKADOT, STRAWBERRY CHARL...

[2472 rows x 2 columns]
```

Association Rules:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(60 CAKE CASES DOLLY GIRL DESIGN)	(PACK OF 72 RETROSPOT CAKE CASES)	0.023160	0.071206	0.013028	0.562500	7.899629	0.011378	2.122958	0.894120
1	(60 TEATIME FAIRY	(PACK OF 72 RETROSPOT	0.044427	0.071206	0.024218	0.545113	7.655446	0.021054	2.041812	0.909794

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
	CAKE CASES)	CAKE CASES)								
2	(ALARM CLOCK BAKELIKE CHOCOLATE)	(ALARM CLOCK BAKELIKE GREEN)	0.023216	0.053558	0.015254	0.657074	12.268575	0.014011	2.759906	0.940321
3	(ALARM CLOCK BAKELIKE CHOCOLATE)	(ALARM CLOCK BAKELIKE PINK)	0.023216	0.042256	0.011691	0.503597	11.917802	0.010710	1.929369	0.937865
4	(ALARM CLOCK BAKELIKE CHOCOLATE)	(ALARM CLOCK BAKELIKE RED)	0.023216	0.057121	0.015811	0.681055	11.923112	0.014485	2.956246	0.937903
...	...	...	...	...	...	...	...	...	...	...
1392	(CHARLOTTE BAG SUKI DESIGN, STRAWBERRY CHARLOT...	(CHARLOTTE BAG PINK POLKADOT, WOODLAND CHARLOT...	0.018483	0.021824	0.011302	0.611446	28.017319	0.010898	2.517477	0.982467

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
1393	(CHARLOTTE BAG SUKI DESIGN, WOODLAND CHARLOTTE...	(CHARLOTTE BAG PINK POLKADOT, STRAWBERRY CHARL...	0.018595	0.020989	0.011302	0.607784	28.957623	0.010911	2.496105	0.983760
1394	(CHARLOTTE BAG PINK POLKADOT, STRAWBERRY CHARL...	(CHARLOTTE BAG SUKI DESIGN, WOODLAND CHARLOTTE...	0.020989	0.018595	0.011302	0.538462	28.957623	0.010911	2.126378	0.986165
1395	(CHARLOTTE BAG PINK POLKADOT, WOODLAND CHARLOT...	(CHARLOTTE BAG SUKI DESIGN, STRAWBERRY CHARLOT...	0.021824	0.018483	0.011302	0.517857	28.017319	0.010898	2.035738	0.985822
1396	(WOODLAND CHARLOTTE BAG, STRAWBERRY CHARLOTTE ...	(CHARLOTTE BAG PINK POLKADOT, CHARLOTTE BAG SU...	0.022492	0.018261	0.011302	0.502475	27.516648	0.010891	1.973247	0.985832

1397 rows × 10 columns

The code uses the apriori algorithm and association rule mining techniques to analyze the occurrence of items in bills. Here's the overall idea:

Frequent Itemsets Generation:

The apriori algorithm is applied to the pivot_table created earlier, which represents the occurrence of items in bills.

The algorithm identifies sets of items that frequently co-occur together in the bills.

The minimum support threshold of 0.01 (1%) is set, meaning that an itemset must occur in at least 1% of the bills to be considered frequent.

The resulting frequent itemsets represent combinations of items that are frequently observed together in bills.

Association Rules Generation:

Using the frequent itemsets, association rules are generated.

Association rules capture relationships and patterns between items based on their co-occurrence in bills.

The confidence metric is used to evaluate the strength of the rules. Confidence measures how often the consequent item(s) appear in bills when the antecedent item(s) are present.

A minimum confidence threshold of 0.5 (50%) is set, meaning that only rules with a confidence greater than or equal to 0.5 will be considered significant.

By applying these techniques to the pivot_table, the code enables the discovery of frequent itemsets and the extraction of meaningful association rules, helping to uncover hidden patterns and relationships in the data.

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

rules
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
17	(BEADED CRYSTAL HEART PINK ON STICK)	(DOTCOM POSTAGE)	0.011469	0.039305	0.011190	0.975728	24.824404	0.010740	39.580626	0.970851
614	(HERB MARKER CHIVES, HERB MARKER THYME)	(HERB MARKER PARSLEY)	0.010411	0.012916	0.010077	0.967914	74.938272	0.009942	30.764113	0.997036
607	(HERB MARKER CHIVES, HERB MARKER ROSEMARY)	(HERB MARKER PARSLEY)	0.010355	0.012916	0.010021	0.967742	74.924917	0.009887	30.599599	0.996977
619	(HERB MARKER CHIVES, HERB MARKER ROSEMARY)	(HERB MARKER THYME)	0.010355	0.012916	0.010021	0.967742	74.924917	0.009887	30.599599	0.996977
1217	(HERB MARKER BASIL, HERB MARKER ROSEMARY, HERB...	(HERB MARKER THYME)	0.010578	0.012916	0.010188	0.963158	74.570009	0.010052	26.792276	0.997137

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
...	...	...	...	...	...	...	...	...	...	...
25	(RED RETROS POT CUP)	(BLUE POLKA DOT CUP)	0.021378	0.018038	0.010689	0.500000	27.719136	0.010304	1.963924	0.984981
1159	(STRAWBERRY CHARLOTTE BAG, RED RETROS POT CHARL...	(CHARLOTTE BAG PINK POLKADOT, WOODLAND CHARLOT...	0.026834	0.021824	0.013417	0.500000	22.910714	0.012832	1.956352	0.982723
113	(HAND WARMER RED LOVE HEART)	(HAND WARMER SCOTTY DOG DESIGN)	0.021935	0.030286	0.010968	0.500000	16.509191	0.010303	1.939428	0.960496
147	(LOVE HOT WATER BOTTLE)	(HOT WATER BOTTLE KEEP CALM)	0.025832	0.042701	0.012916	0.500000	11.709257	0.011813	1.914597	0.938850
370	(CHARLOTTE BAG PINK POLKADOT,	(PACK OF 72 RETROS POT	0.021824	0.071206	0.010912	0.500000	7.021892	0.009358	1.857588	0.876722

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
	WOODLAND CHARLOT...	CAKE CASES)								

1397 rows × 10 columns

```
rules.sort_values(by='support', ascending=False)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
161	(JUMBO BAG PINK POLKA DOT)	(JUMBO BAG RED RETROSPOT)	0.067309	0.113963	0.045596	0.677419	5.944214	0.037926	2.746715	0.891795
105	(ROSES REGENCY TEACUP AND SAUCER)	(GREEN REGENCY TEACUP AND SAUCER)	0.056174	0.054170	0.040641	0.723489	13.355912	0.037598	3.420583	0.980188
104	(GREEN REGENCY TEACUP AND SAUCER)	(ROSES REGENCY TEACUP AND SAUCER)	0.054170	0.056174	0.040641	0.750257	13.355912	0.037598	3.779187	0.978111

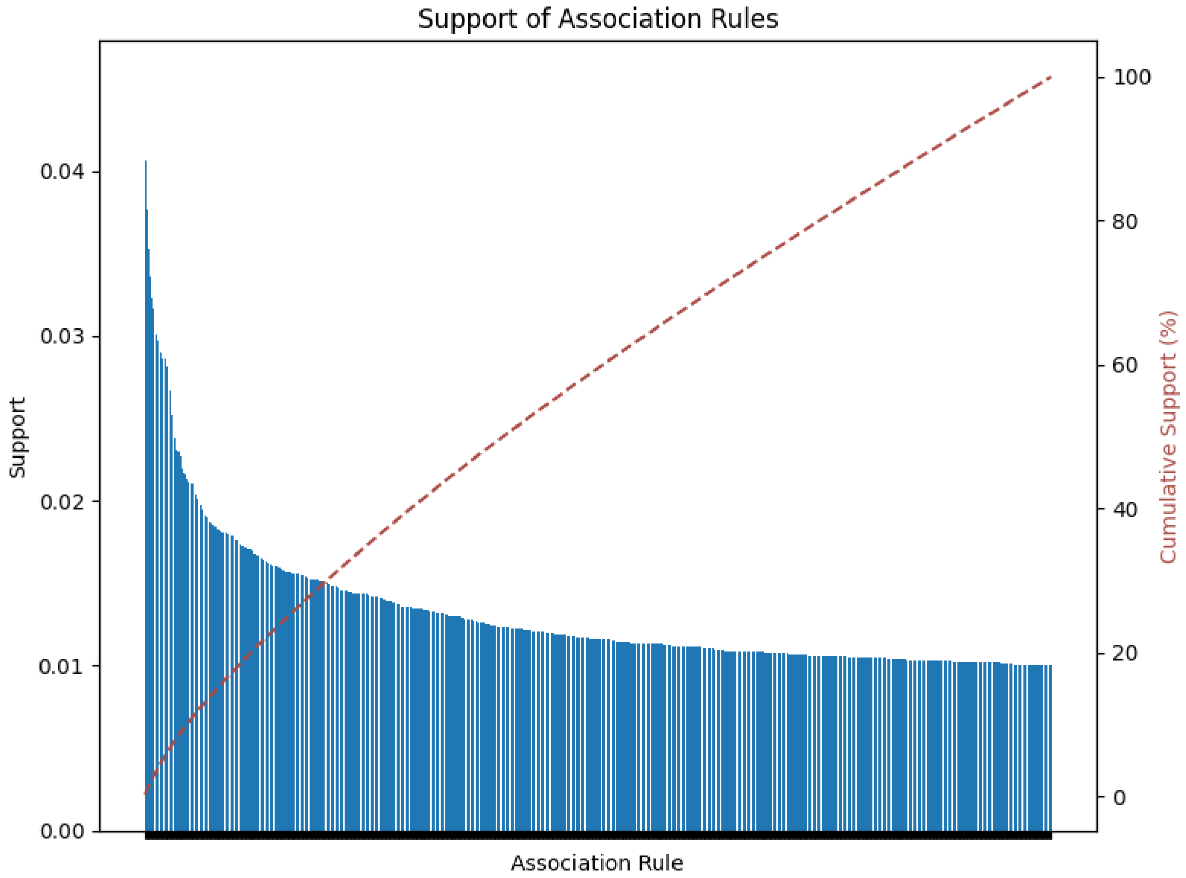
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
174	(JUMBO STORAGE BAG SUKI)	(JUMBO BAG RED RETROSPOT)	0.065583	0.113963	0.040140	0.612054	5.370650	0.032666	2.283921	0.870920
172	(JUMBO SHOPPER VINTAGE RED PAISLEY)	(JUMBO BAG RED RETROSPOT)	0.064859	0.113963	0.037635	0.580258	5.091639	0.030243	2.110907	0.859335
...	...	...	...	...	...	...	...	...	...	...
608	(HERB MARKER ROSEMARY, HERB MARKER PARSLEY)	(HERB MARKER CHIVES)	0.011691	0.011469	0.010021	0.857143	74.737864	0.009887	6.919719	0.998291
623	(HERB MARKER ROSEMARY)	(HERB MARKER CHIVES, HERB MARKER THYME)	0.013028	0.010411	0.010021	0.769231	73.887289	0.009886	4.288220	0.999487

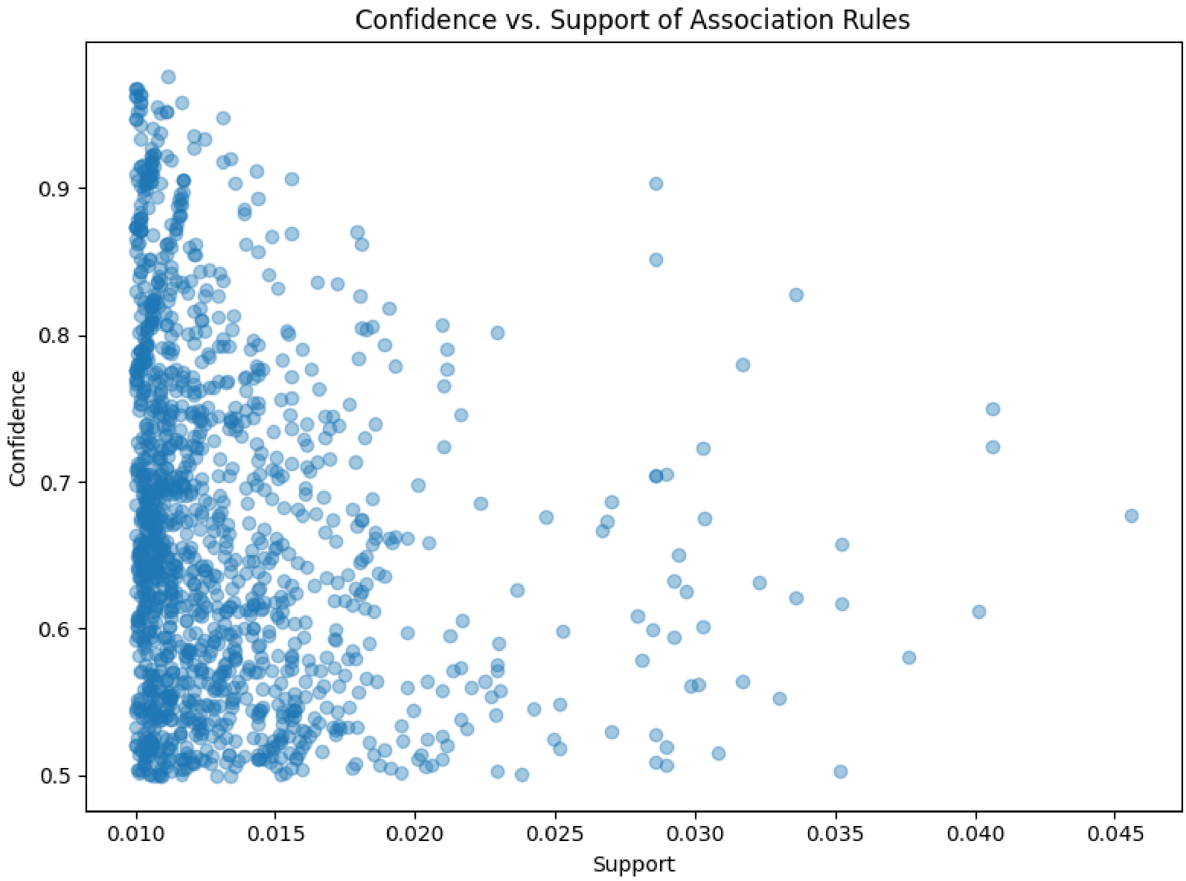
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
987	(LUNCH BAG PINK POLKA DOT, LUNCH BAG APPLE DESIGN)	(LUNCH BAG SPACEBOY DESIGN)	0.019263	0.063857	0.010021	0.520231	8.146812	0.008791	1.951238	0.894483
673	(LUNCH BAG RED RETROSPOT, JUMBO BAG BAROQUE B...	(JUMBO BAG RED RETROSPOT)	0.014364	0.113963	0.010021	0.697674	6.121948	0.008384	2.930738	0.848846
431	(LUNCH BOX I LOVE LONDON, DOLLY GIRL LUNCH BOX)	(SPACEBOY LUNCH BOX)	0.014141	0.049215	0.010021	0.708661	14.399295	0.009325	3.263505	0.943900

1397 rows × 10 columns

```
# Sort rules by support in descending order
sorted_rules = rules.sort_values(by='support', ascending=False)

# Calculate cumulative support
cumulative_support = np.cumsum(sorted_rules['support'] / np.sum(sorted_rules['support']) * 100
)

# Bar plot for Support
```

```
fig, ax1 = plt.subplots(figsize=(8, 6))
ax1.bar(range(len(sorted_rules)), sorted_rules['support'], align='center')
plt.xticks(range(len(sorted_rules)), ['' for _ in range(len(sorted_rules))])  # Remove x-axis labels
ax1.set_xlabel('Association Rule')
ax1.set_ylabel('Support')
ax1.set_title('Support of Association Rules')

# CDF plot for cumulative support
ax2 = ax1.twinx()
ax2.plot(range(len(sorted_rules)), cumulative_support, color='#AA4A44', linestyle='--')
ax2.set_ylabel('Cumulative Support (%)', c='#AA4A44')

plt.tight_layout()
plt.show()

# Scatter plot for Confidence vs. Support
plt.figure(figsize=(8, 6))
plt.scatter(rules['support'], rules['confidence'], alpha=0.4)
plt.xlabel('Support')
plt.ylabel('Confidence')
plt.title('Confidence vs. Support of Association Rules')
plt.tight_layout()
plt.show()
```





These two visualizations explore the association rules: a bar plot for the support of association rules and a scatter plot for the confidence vs. support of association rules.

The bar plot represents the support values of the association rules. Each bar corresponds to a rule, and its height represents the support value, indicating how frequently the rule occurs in the dataset. The y-axis represents the support, while the x-axis does not display any labels, focusing solely on the visualization of support values.

The cumulative distribution function (CDF) plot showcases the cumulative support of the association rules as a percentage. It helps understand the distribution of support values across the rules in a cumulative manner. The red dashed line in the CDF plot connects the cumulative support values for each rule, providing insights into the accumulation of support as the rules progress.

The scatter plot displays the relationship between confidence and support for the association rules. Each point represents a rule, with the x-axis representing the support and the y-axis representing the confidence. The plot shows how the confidence varies with different levels of support, helping identify any patterns or trends between these two metrics.

These visualizations offer valuable insights into the support, confidence, and their relationships within the association rules, aiding in the interpretation and analysis of the rules' strength and significance.

Cross-Selling and Upselling

```python
# 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)

# 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)

# 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}'.")

# Display upselling recommendations
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)}.")
```

Cross-Selling Recommendations :

```
Customers who bought 'BEADED CRYSTAL HEART PINK ON STICK' also bought 'DOTCOM PO
STAGE'.
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
'.

Upselling Recommendations:
For customers who bought 'HERB MARKER CHIVES', recommend the following upgrades: HE
RB MARKER PARSLEY, HERB MARKER THYME.
For customers who bought 'HERB MARKER CHIVES', recommend the following upgrades: HE
RB MARKER PARSLEY, HERB MARKER MINT.
```

```
For customers who bought 'HERB MARKER CHIVES', recommend the following upgrades: HE
RB MARKER ROSEMARY, HERB MARKER PARSLEY.
For customers who bought 'HERB MARKER CHIVES', recommend the following upgrades: HE
RB MARKER ROSEMARY, HERB MARKER THYME.
For customers who bought 'HERB MARKER THYME', recommend the following upgrades: HE
RB MARKER ROSEMARY, HERB MARKER PARSLEY.
```

Upselling Recommendations

During the analysis of upselling opportunities, it was observed that multiple product recommendations were being made for the same item. To address this issue and provide more diverse recommendations, a modification was made to recommend only one product for each top item instead of recommending based on the top confidence values.

By implementing this change, we ensure that the upselling recommendations do not repeatedly suggest the same product to customers. This approach enhances the variety of product recommendations and increases the chances of cross-selling and upselling success.

The updated recommendation strategy focuses on identifying the top items and selecting a single recommended product for each of them. This adjustment aims to optimize the upselling strategy by suggesting different upgrades or add-on products to customers, resulting in a more compelling and varied range of recommendations.

```
top_upselling = upselling_rules.sort_values(['confidence', 'support'], ascending=False).drop_duplic
ates('antecedents')[:5]
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)}.")
```

```
For customers who bought 'HERB MARKER CHIVES', recommend the following upgrades: HE
RB MARKER PARSLEY, HERB MARKER THYME.
For customers who bought 'HERB MARKER THYME', recommend the following upgrades: HE
RB MARKER ROSEMARY, HERB MARKER PARSLEY.
For customers who bought 'HERB MARKER PARSLEY', recommend the following upgrades: H
ERB MARKER ROSEMARY, HERB MARKER THYME.
For customers who bought 'HERB MARKER ROSEMARY', recommend the following upgrades:
HERB MARKER PARSLEY, HERB MARKER THYME.
For customers who bought 'REGENCY TEA PLATE PINK', recommend the following upgrades:
REGENCY TEA PLATE GREEN, REGENCY TEA PLATE ROSES.
```

Conclusion

In this project, we explored the concept of association rules using the Apriori algorithm and the mlxtend library in Python. Association rules analysis provides valuable insights into the relationships and patterns within a dataset, enabling businesses to uncover hidden associations between items and make informed decisions for various applications.

We started by preparing the data and filtering out infrequent items and irrelevant transactions. Then, we generated frequent itemsets and association rules based on predefined thresholds for support and confidence. These rules allowed us to identify significant associations between items and quantify their strength.

The generated association rules provided actionable insights for different business scenarios. We explored cross-selling opportunities by identifying products frequently purchased together. By leveraging these associations, businesses can implement effective cross-selling strategies, offering relevant add-on products or upgrades to customers, thereby increasing revenue.

Additionally, we examined upselling recommendations, focusing on identifying suitable product upgrades or higher-priced alternatives for customers. By considering only one product recommendation for each top item, we ensured diverse and relevant suggestions, avoiding repetitive recommendations and enhancing the upselling strategy.

Furthermore, we discussed the importance of interpreting the support, confidence, lift, leverage, and conviction metrics associated with association rules. These metrics provide quantitative measures of the strength, significance, and impact of the associations, enabling businesses to prioritize and optimize their decision-making processes.

Overall, association rules analysis offers valuable insights and practical applications across various domains, such as marketing, product recommendations, cross-selling strategies, and process optimization. By understanding the associations between items, businesses can make data-driven decisions, improve customer satisfaction, enhance marketing campaigns, and drive business growth.

It is important to note that the analysis and insights provided in this project are specific to the dataset and parameters used. The results can be further refined and customized based on the specific requirements, domain knowledge, and business objectives.