MARKET BASKET INSIGHTS

IBM Naan Mudhalvan Phase 4: Development Part 2

Introduction:

In this phase we transform the data suitable for applying association rules such as Apriori to identify the products that frequently co-occur in transactions.

Table Of Contents:

1. Formatting Transaction Data For Analysis

Splitting the 'Itemname' column in the transaction_data DataFrame into individual items, creating a new detailed DataFrame, then combining the original and new DataFrames, offering a clearer representation of item transactions for analysis.

Code:

```
# Split the 'Itemname' column into individual items
items_df = transaction_data['Itemname'].str.split(', ',
expand=True)
# Concatenate the original DataFrame with the new
items DataFrame
transaction_data = pd.concat([transaction_data,
items_df], axis=1)
# Drop the original 'Itemname' column
transaction_data = transaction_data.drop('Itemname',
axis=1)
# Display the resulting DataFrame
print(transaction_data.head())
```

Output:

```
                                0                              1  \
0  WHITE HANGING HEART T-LIGHT HOLDER            WHITE METAL LANTERN
1             HAND WARMER UNION JACK      HAND WARMER RED POLKA DOT
2       ASSORTED COLOUR BIRD ORNAMENT      POPPY'S PLAYHOUSE BEDROOM
3           JAM MAKING SET WITH JARS  RED COAT RACK PARIS FASHION
4            BATH BUILDING BLOCK WORD                           None

                                 2                                    3  \
0  CREAM CUPID HEARTS COAT HANGER  KNITTED UNION FLAG HOT WATER BOTTLE
1                            None                                 None
2       POPPY'S PLAYHOUSE KITCHEN    FELTCRAFT PRINCESS CHARLOTTE DOLL
3  YELLOW COAT RACK PARIS FASHION          BLUE COAT RACK PARIS FASHION
4                            None                                 None

                                4                                    5  \
0  RED WOOLLY HOTTIE WHITE HEART.         SET 7 BABUSHKA NESTING BOXES
1                            None                                 None
2          IVORY KNITTED MUG COSY  BOX OF 6 ASSORTED COLOUR TEASPOONS
3                            None                                 None
4                            None                                 None

                                     6                              7  \
0  GLASS STAR FROSTED T-LIGHT HOLDER                              None
1                                 None                              None
2        BOX OF VINTAGE JIGSAW BLOCKS  BOX OF VINTAGE ALPHABET BLOCKS
3                                 None                              None
4                                 None                              None

                          8                           9  ...   534   535   536  \
0                      None                        None  ...  None  None  None
1                      None                        None  ...  None  None  None
2  HOME BUILDING BLOCK WORD  LOVE BUILDING BLOCK WORD  ...  None  None  None
3                      None                        None  ...  None  None  None
4                      None                        None  ...  None  None  None

    537   538   539   540   541   542   543
0  None  None  None  None  None  None  None
1  None  None  None  None  None  None  None
2  None  None  None  None  None  None  None
3  None  None  None  None  None  None  None
4  None  None  None  None  None  None  None

[5 rows x 544 columns]
```

2. Data Encoding: Converting Items into Boolean Representation

This step converts items in the transaction_data DataFrame into binary (1 or 0) values using one-hot encoding. Each item becomes a column, indicating its presence (1) or absence (0) in a transaction. Then we save it in a separate CSV file.

Code:

```
# Convert items to boolean columns
df_encoded = pd.get_dummies(transaction_data,
prefix='', prefix_sep='').groupby(level=0, axis=1).max()


# Save the transaction data to a CSV file
df_encoded.to_csv('transaction_data_encoded.csv',
index=False)
```

3. Applying Apriori Algorithm for Association Rule Mining

We apply the Apriori algorithm to perform association rule mining on the encoded transaction data.
The min_support parameter is set to 0.009 to filter out infrequent itemsets. The resulting frequent itemsets are then used to generate association rules based on a minimum confidence threshold of 0.5.

Code:

```
# Load transaction data into a DataFrame
df_encoded =
pd.read_csv('transaction_data_encoded.csv')


from mlxtend.frequent_patterns import apriori,
association_rules


# Association Rule Mining
frequent_itemsets = apriori(df_encoded,
min_support=0.009, use_colnames=True)
rules = association_rules(frequent_itemsets,
metric="confidence", min_threshold=0.5)
selected_columns = ['antecedents', 'consequents',
'support', 'confidence']
print("Association Rules:")
print(rules[selected_columns])
```

Output:

```
                                   0                              1  \
0  WHITE HANGING HEART T-LIGHT HOLDER            WHITE METAL LANTERN
1            HAND WARMER UNION JACK         HAND WARMER RED POLKA DOT
2       ASSORTED COLOUR BIRD ORNAMENT       POPPY'S PLAYHOUSE BEDROOM
3          JAM MAKING SET WITH JARS   RED COAT RACK PARIS FASHION
4           BATH BUILDING BLOCK WORD                            None

                             2                                   3   \
0  CREAM CUPID HEARTS COAT HANGER  KNITTED UNION FLAG HOT WATER BOTTLE
1                            None                                 None
2       POPPY'S PLAYHOUSE KITCHEN     FELTCRAFT PRINCESS CHARLOTTE DOLL
3  YELLOW COAT RACK PARIS FASHION          BLUE COAT RACK PARIS FASHION
4                            None                                 None

                             4                                  5   \
0  RED WOOLLY HOTTIE WHITE HEART.         SET 7 BABUSHKA NESTING BOXES
1                            None                                 None
2          IVORY KNITTED MUG COSY  BOX OF 6 ASSORTED COLOUR TEASPOONS
3                            None                                 None
4                            None                                 None

                                   6                               7   \
0  GLASS STAR FROSTED T-LIGHT HOLDER                               None
1                               None                               None
2       BOX OF VINTAGE JIGSAW BLOCKS  BOX OF VINTAGE ALPHABET BLOCKS
3                               None                               None
4                               None                               None

                          8                          9  ...   534   535   536  \
0                      None                       None  ...  None  None  None
1                      None                       None  ...  None  None  None
2  HOME BUILDING BLOCK WORD  LOVE BUILDING BLOCK WORD  ...  None  None  None
3                      None                       None  ...  None  None  None
4                      None                       None  ...  None  None  None

    537   538   539   540   541   542   543
0  None  None  None  None  None  None  None
1  None  None  None  None  None  None  None
2  None  None  None  None  None  None  None
3  None  None  None  None  None  None  None
4  None  None  None  None  None  None  None

[5 rows x 544 columns]
```

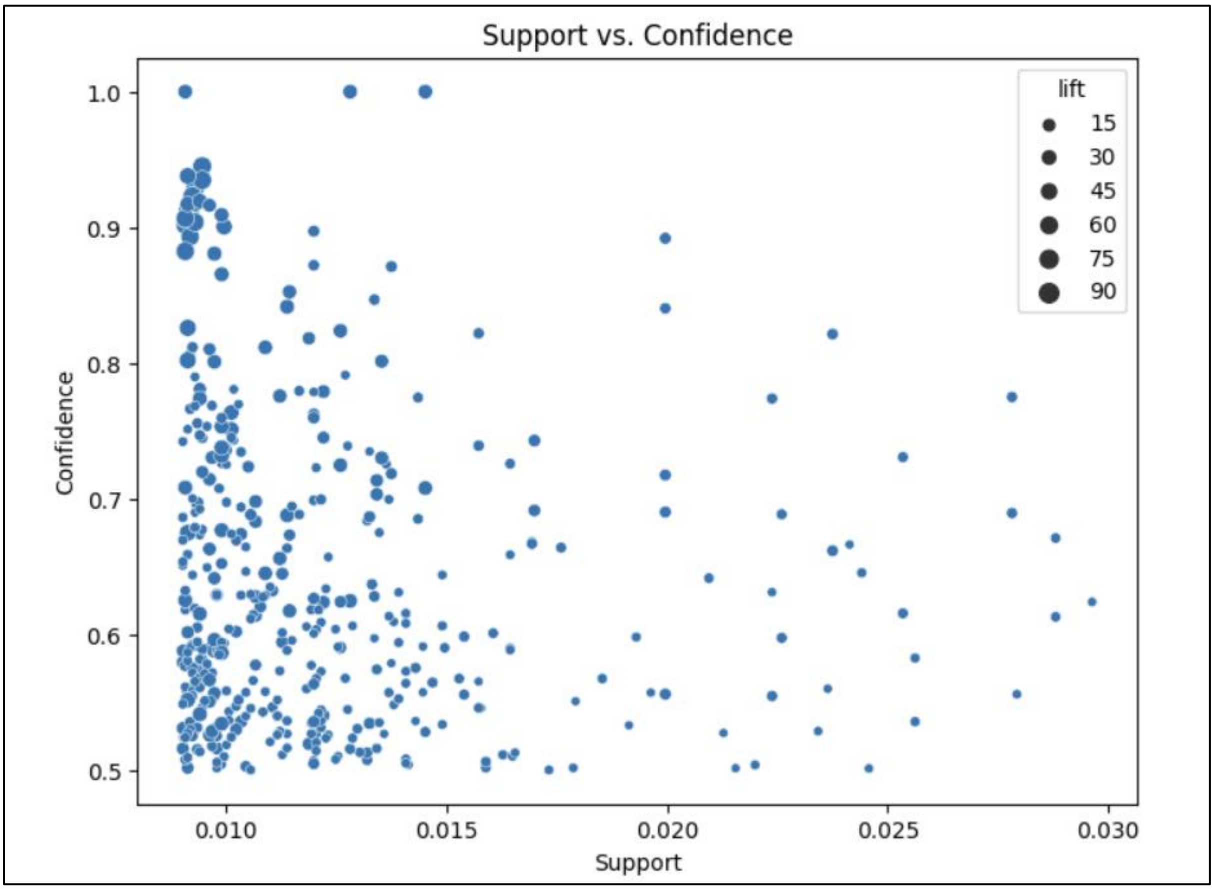
4. Generating Insights By Visualization Tools

Scatter Plot

Code:

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='support', y='confidence', size='lift', data=rules)
plt.title('Support vs. Confidence')
plt.xlabel('Support')
plt.ylabel('Confidence')
plt.show()
```

Output:

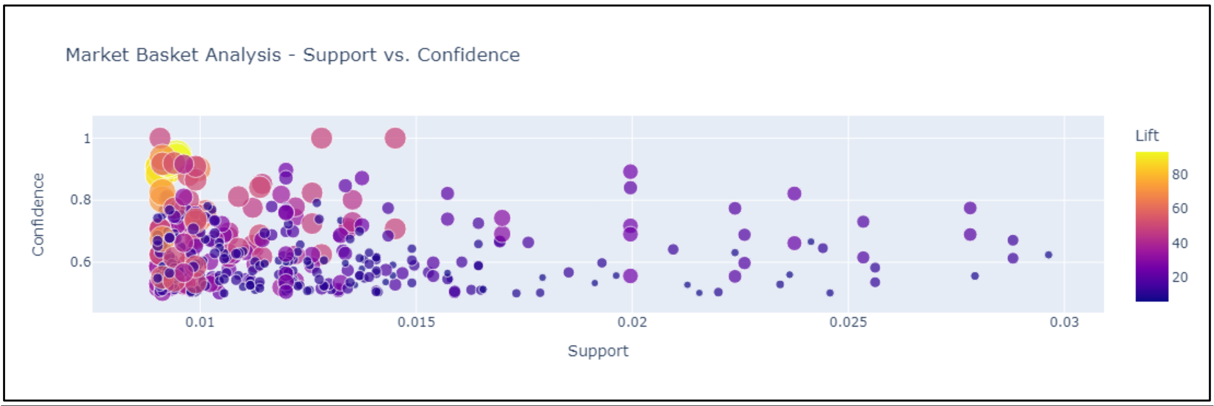


Code:

```
import plotly.express as px

# Convert frozensets to lists for serialization
rules['antecedents'] = rules['antecedents'].apply(list)
rules['consequents'] = rules['consequents'].apply(list)

# Create an interactive scatter plot using plotly express
```

```
fig = px.scatter(rules, x="support", y="confidence",
size="lift",color="lift", hover_name="consequents",
title='Market Basket Analysis - Support vs.
Confidence',labels={'support': 'Support', 'confidence':
'Confidence'})

# Customize the layout
fig.update_layout(
    xaxis_title='Support',
    yaxis_title='Confidence',
    coloraxis_colorbar_title='Lift',
    showlegend=True
)

# Show the interactive plot
fig.show()
```

Output:

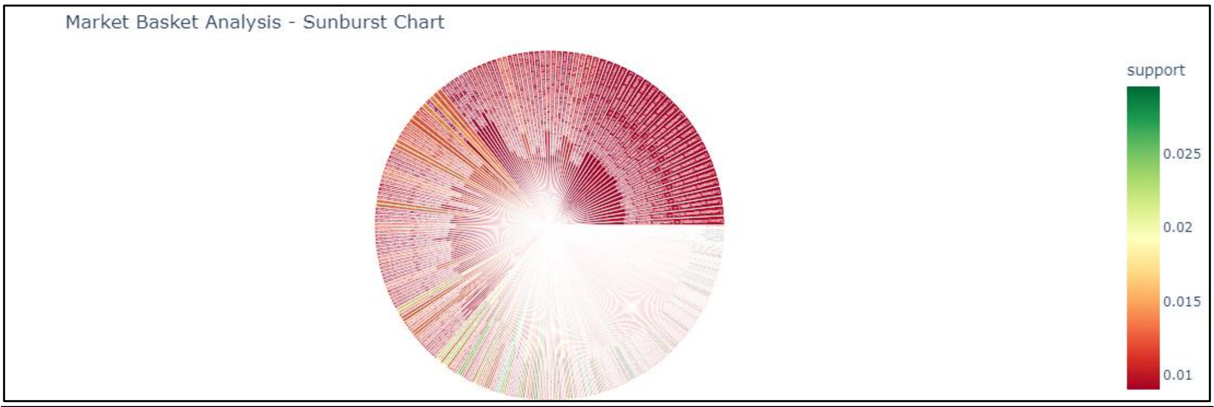


Code:

```
import plotly.express as px

# Combine antecedents and consequents into a single column for each rule
rules['rule'] = rules['antecedents'].astype(str) + ' -> ' + rules['consequents'].astype(str)

# Create a sunburst chart
fig = px.sunburst(rules, path=['rule'], values='lift',
title='Market Basket Analysis - Sunburst Chart', color='support', color_continuous_scale='rdylgn')

# Customize the layout
fig.update_layout(
    margin=dict(l=0, r=0, b=0, t=40),
```

```
)

# Show the interactive plot
fig.show()
```

Output:



Conclusion:

Thus we applied association rules on the given dataset and identified frequently occurring items in an itemsets and generated insights by using various visualization tools.