Market basket Insights Phase4

Formatting the transaction data in a suitable format for analysis

***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())

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]

***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)

***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=0.007, use\_colnames=True)

rules = association\_rules(frequent\_itemsets, metric="confidence", min\_threshold=0.5)

***Display information of the rules***

print("Association Rules:")

print(rules.head())

Association Rules:

antecedents consequents \

0 (CHOCOLATE BOX RIBBONS) (6 RIBBONS RUSTIC CHARM)

1 (60 CAKE CASES DOLLY GIRL DESIGN) (PACK OF 72 RETROSPOT CAKE CASES)

2 (60 TEATIME FAIRY CAKE CASES) (PACK OF 72 RETROSPOT CAKE CASES)

3 (ALARM CLOCK BAKELIKE CHOCOLATE) (ALARM CLOCK BAKELIKE GREEN)

4 (ALARM CLOCK BAKELIKE CHOCOLATE) (ALARM CLOCK BAKELIKE PINK)

antecedent support consequent support support confidence lift \

0 0.012368 0.039193 0.007036 0.568889 14.515044

1 0.018525 0.054529 0.010059 0.543027 9.958409

2 0.034631 0.054529 0.017315 0.500000 9.169355

3 0.017150 0.042931 0.011379 0.663462 15.454151

4 0.017150 0.032652 0.009125 0.532051 16.294742

leverage conviction zhangs\_metric

0 0.006551 2.228676 0.942766

1 0.009049 2.068984 0.916561

2 0.015427 1.890941 0.922902

3 0.010642 2.843862 0.951613

4 0.008565 2.067210 0.955009

## Visualizing Market Basket Analysis Results

import matplotlib.pyplot as plt

import seaborn as sns

***Plot scatterplot for Support vs. Confidence***

plt.figure(figsize=(12, 8))

sns.scatterplot(x="support", y="confidence", size="lift", data=rules, hue="lift", palette="viridis", sizes=(20, 200))

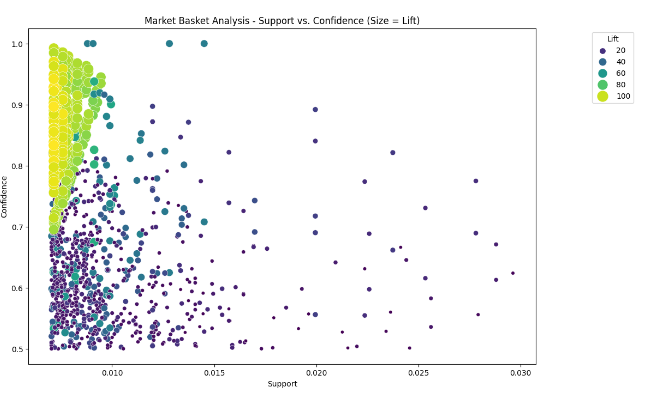
plt.title('Market Basket Analysis - Support vs. Confidence (Size = Lift)')

plt.xlabel('Support')

plt.ylabel('Confidence')

plt.legend(title='Lift', loc='upper right', bbox\_to\_anchor=(1.2, 1))

plt.show()



## Interactive Market Basket Analysis Visualization

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***

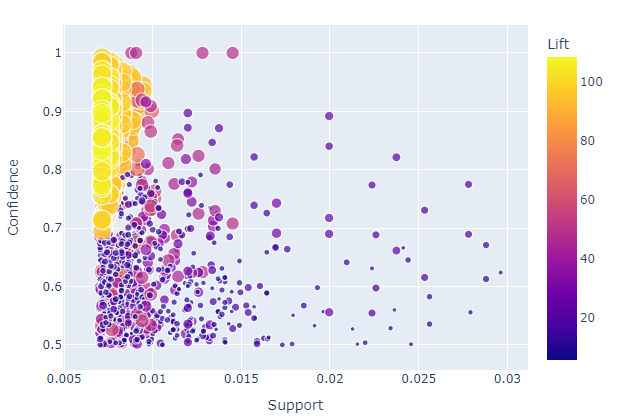
fig = px.scatter(rules, x="support", y="confidence", size="lift",

color="lift", hover\_name="consequents",

title='Market Basket Analysis - Support vs. Confidence',

labels={'support': 'Support', 'confidence': 'Confidence'})

***Customize the layout***



## Interactive Network Visualization for Association Rules

import networkx as nx

import matplotlib.pyplot as plt

import plotly.graph\_objects as go

***Create a directed graph***

G = nx.DiGraph()

***Add nodes and edges from association rules***

for idx, row **in** rules.iterrows():

G.add\_node(tuple(row['antecedents']), color='skyblue')

G.add\_node(tuple(row['consequents']), color='orange')

G.add\_edge(tuple(row['antecedents']), tuple(row['consequents']), weight=row['support'])

***Set node positions using a spring layout***

pos = nx.spring\_layout(G)

***Create an interactive plot using plotly***

edge\_x = []

edge\_y = []

for edge **in** G.edges(data=True):

x0, y0 = pos[edge[0]]

x1, y1 = pos[edge[1]]

edge\_x.append(x0)

edge\_x.append(x1)

edge\_x.append(None)

edge\_y.append(y0)

edge\_y.append(y1)

edge\_y.append(None)

edge\_trace = go.Scatter(

x=edge\_x, y=edge\_y,

line=dict(width=0.5, color='#888'),

hoverinfo='none',

mode='lines')

node\_x = []

node\_y = []

for node **in** G.nodes():

x, y = pos[node]

node\_x.append(x)

node\_y.append(y)

node\_trace = go.Scatter(

x=node\_x, y=node\_y,

mode='markers',

hoverinfo='text',

marker=dict(

showscale=True,

colorscale='YlGnBu',

size=10,

colorbar=dict(

thickness=15,

title='Node Connections',

xanchor='left',

titleside='right'

)

)

)

***Customize the layout***

layout = go.Layout(

showlegend=False,

hovermode='closest',

margin=dict(b=0, l=0, r=0, t=0),

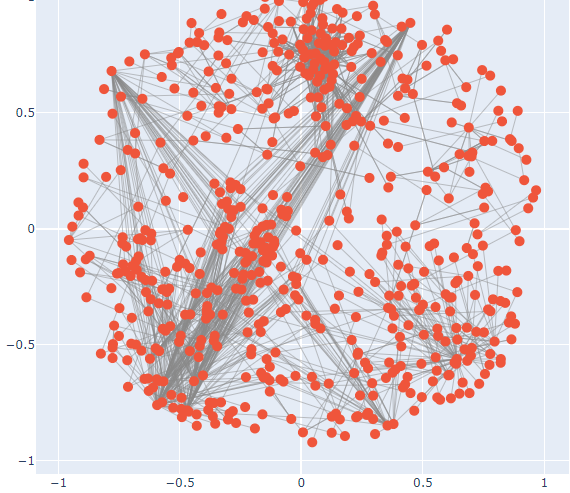
)

***Create the figure***

fig = go.Figure(data=[edge\_trace, node\_trace], layout=layout)

***Show the interactive graph***

fig.show()



## Interactive Sunburst Chart for Association Rules

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='YlGnBu')

***Customize the layout***

fig.update\_layout(

margin=dict(l=0, r=0, b=0, t=40),

)

***Show the interactive plot***

fig.show()

