##F3 RFM ANALYSIS

import pandas as pd

import plotly.express as px

import plotly.io as pio

import plotly.graph\_objects as go

pio.templates.default = "plotly\_white"

data = pd.read\_csv("C:/Users/Shabnam Kabir/Downloads/rfm\_data.csv")

print(data.head())

from datetime import datetime

import pandas as pd

# Convert 'PurchaseDate' to datetime

data['PurchaseDate'] = pd.to\_datetime(data['PurchaseDate'])

# Calculate Recency

data['Recency'] = (datetime.now() - data['PurchaseDate']).dt.days

# # Calculate Frequency

frequency\_data = data.groupby('CustomerID')['OrderID'].count().reset\_index()

frequency\_data.rename(columns={'OrderID': 'Frequency'}, inplace=True)

data = data.merge(frequency\_data, on='CustomerID', how='left')

# # Calculate Monetary Value

monetary\_data = data.groupby('CustomerID')['TransactionAmount'].sum().reset\_index()

monetary\_data.rename(columns={'TransactionAmount': 'MonetaryValue'}, inplace=True)

data = data.merge(monetary\_data, on='CustomerID', how='left')

print(data.head())

# Define scoring criteria for each RFM value

recency\_scores = [5, 4, 3, 2, 1]  # Higher score for lower recency (more recent)

frequency\_scores = [1, 2, 3, 4, 5]  # Higher score for higher frequency

monetary\_scores = [1, 2, 3, 4, 5]  # Higher score for higher monetary value

# Calculate RFM scores

data['RecencyScore'] = pd.cut(data['Recency'], bins=5, labels=recency\_scores)

data['FrequencyScore'] = pd.cut(data['Frequency'], bins=5, labels=frequency\_scores)

data['MonetaryScore'] = pd.cut(data['MonetaryValue'], bins=5, labels=monetary\_scores)

# Convert RFM scores to numeric type

data['RecencyScore'] = data['RecencyScore'].astype(int)

data['FrequencyScore'] = data['FrequencyScore'].astype(int)

data['MonetaryScore'] = data['MonetaryScore'].astype(int)

# Calculate RFM score by combining the individual scores

data['RFM\_Score'] = data['RecencyScore'] + data['FrequencyScore'] + data['MonetaryScore']

# Create RFM segments based on the RFM score

segment\_labels = ['Low-Value', 'Mid-Value', 'High-Value']

data['Value Segment'] = pd.qcut(data['RFM\_Score'], q=3, labels=segment\_labels)

print(data.head())

# RFM Segment Distribution

segment\_counts = data['Value Segment'].value\_counts().reset\_index()

segment\_counts.columns = ['Value Segment', 'Count']

pastel\_colors = px.colors.qualitative.Pastel

# Create the bar chart

fig\_segment\_dist = px.bar(segment\_counts, x='Value Segment', y='Count',

                          color='Value Segment', color\_discrete\_sequence=pastel\_colors,

                          title='RFM Value Segment Distribution')

# Update the layout

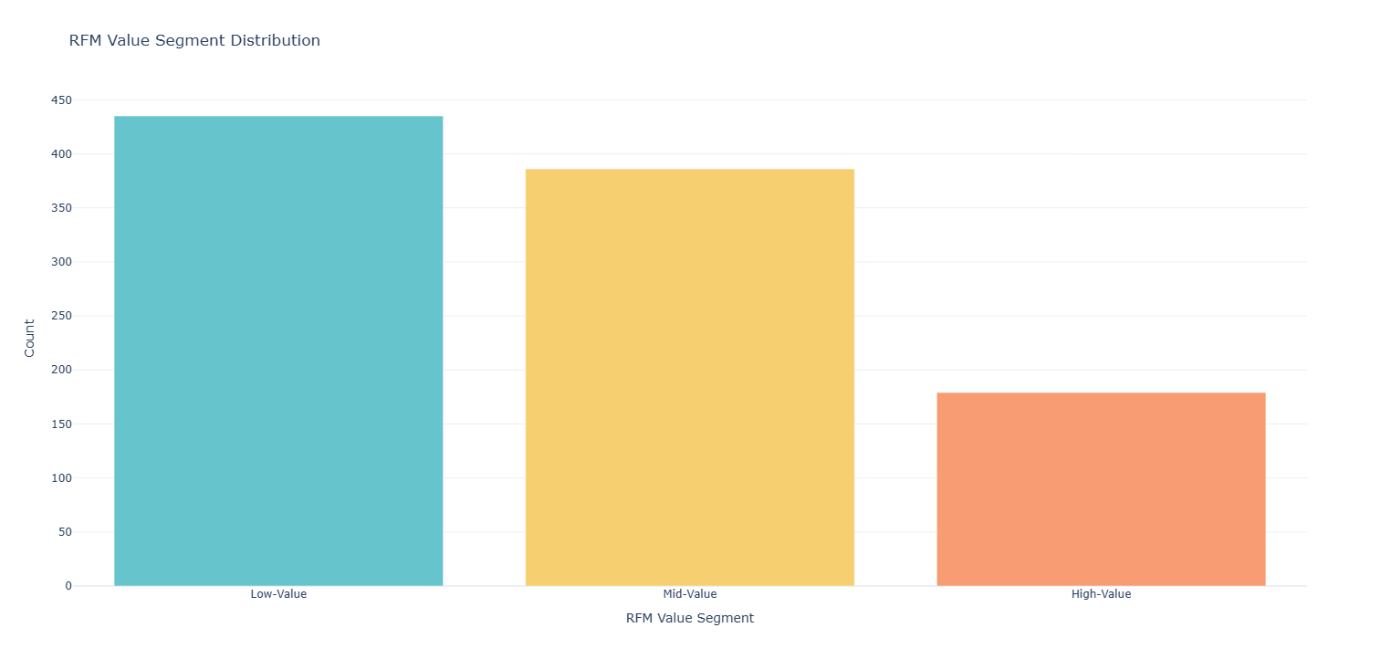
fig\_segment\_dist.update\_layout(xaxis\_title='RFM Value Segment',

                              yaxis\_title='Count',

                              showlegend=False)

# Show the figure

fig\_segment\_dist.show()



#RFM Customer Segments

data['RFM Customer Segments'] = ''

# Assign RFM segments based on the RFM score

data.loc[data['RFM\_Score'] >= 9, 'RFM Customer Segments'] = 'Champions'

data.loc[(data['RFM\_Score'] >= 6) & (data['RFM\_Score'] < 9), 'RFM Customer Segments'] = 'Potential Loyalists'

data.loc[(data['RFM\_Score'] >= 5) & (data['RFM\_Score'] < 6), 'RFM Customer Segments'] = 'At Risk Customers'

data.loc[(data['RFM\_Score'] >= 4) & (data['RFM\_Score'] < 5), 'RFM Customer Segments'] = "Can't Lose"

data.loc[(data['RFM\_Score'] >= 3) & (data['RFM\_Score'] < 4), 'RFM Customer Segments'] = "Lost"

# Print the updated data with RFM segments

print(data[['CustomerID', 'RFM Customer Segments']])

##RFM analysis

segment\_product\_counts = data.groupby(['Value Segment', 'RFM Customer Segments']).size().reset\_index(name='Count')

segment\_product\_counts = segment\_product\_counts.sort\_values('Count', ascending=False)

fig\_treemap\_segment\_product = px.treemap(segment\_product\_counts,

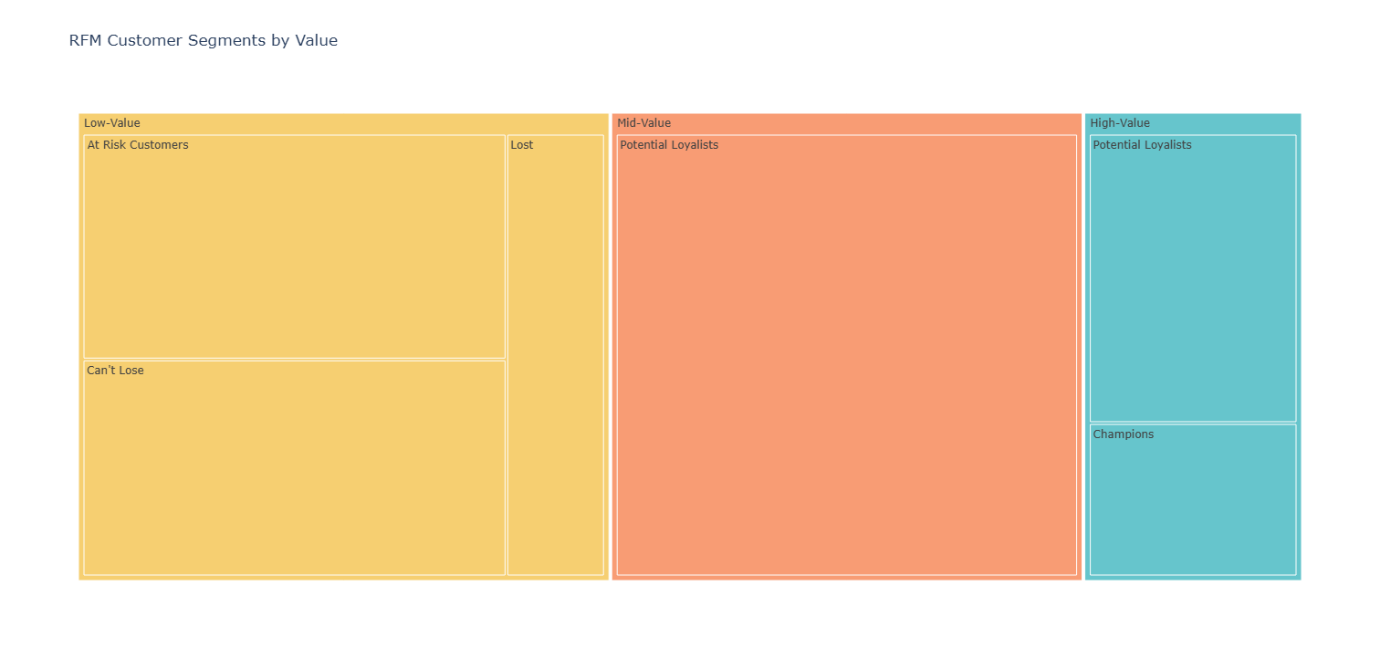
                                         path=['Value Segment', 'RFM Customer Segments'],

                                         values='Count',

                                         color='Value Segment', color\_discrete\_sequence=px.colors.qualitative.Pastel,

                                         title='RFM Customer Segments by Value')

fig\_treemap\_segment\_product.show()



#the distribution of RFM values within the Champions segment

champions\_segment = data[data['RFM Customer Segments'] == 'Champions']

fig = go.Figure()

fig.add\_trace(go.Box(y=champions\_segment['RecencyScore'], name='Recency'))

fig.add\_trace(go.Box(y=champions\_segment['FrequencyScore'], name='Frequency'))

fig.add\_trace(go.Box(y=champions\_segment['MonetaryScore'], name='Monetary'))

fig.update\_layout(title='Distribution of RFM Values within Champions Segment',

                  yaxis\_title='RFM Value',

                  showlegend=True)

fig.show()



#correlation of the recency, frequency, and monetary scores within the champions segment

correlation\_matrix = champions\_segment[['RecencyScore', 'FrequencyScore', 'MonetaryScore']].corr()

# Visualize the correlation matrix using a heatmap

fig\_heatmap = go.Figure(data=go.Heatmap(

                   z=correlation\_matrix.values,

                   x=correlation\_matrix.columns,

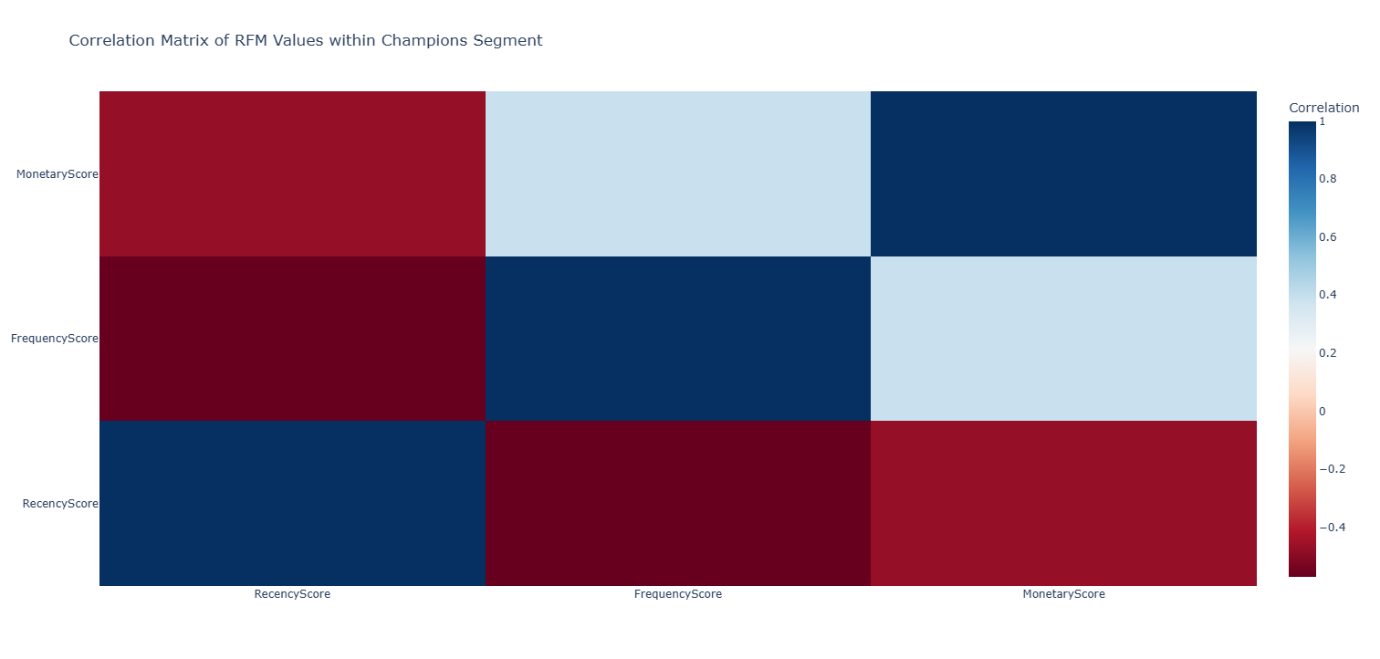
                   y=correlation\_matrix.columns,

                   colorscale='RdBu',

                   colorbar=dict(title='Correlation')))

fig\_heatmap.update\_layout(title='Correlation Matrix of RFM Values within Champions Segment')

fig\_heatmap.show()



##no of customers in all the segments:

import plotly.colors

pastel\_colors = plotly.colors.qualitative.Pastel

segment\_counts = data['RFM Customer Segments'].value\_counts()

# Create a bar chart to compare segment counts

fig = go.Figure(data=[go.Bar(x=segment\_counts.index, y=segment\_counts.values,

                            marker=dict(color=pastel\_colors))])

# Set the color of the Champions segment as a different color

champions\_color = 'rgb(158, 202, 225)'

fig.update\_traces(marker\_color=[champions\_color if segment == 'Champions' else pastel\_colors[i]

                                for i, segment in enumerate(segment\_counts.index)],

                  marker\_line\_color='rgb(8, 48, 107)',

                  marker\_line\_width=1.5, opacity=0.6)

# Update the layout

fig.update\_layout(title='Comparison of RFM Segments',

                  xaxis\_title='RFM Segments',

                  yaxis\_title='Number of Customers',

                  showlegend=False)

fig.show()

