Report 3

**Subtask 1: Calculate RFM Metrics**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load your dataset

df = pd.read\_csv("ecommerce\_data\_final\_cleaned.csv")

# Rename columns for consistency

df = df.rename(columns={'value [USD]': 'total\_revenue', 'date': 'order\_date'})

# Convert date column to datetime

df['order\_date'] = pd.to\_datetime(df['order\_date'], errors='coerce')

# Drop rows with missing customer\_id or revenue or order\_date

df = df.dropna(subset=['customer\_id', 'order\_date', 'total\_revenue'])

# -------------------------------

# 1️⃣ Define a Reference Date

# -------------------------------

# Reference date = one day after the most recent purchase

ref\_date = df['order\_date'].max() + pd.Timedelta(days=1)

# -------------------------------

# 2️⃣ Group Data by Customer ID

# -------------------------------

rfm = df.groupby('customer\_id').agg({

'order\_date': lambda x: (ref\_date - x.max()).days, # Recency

'customer\_id': 'count', # Frequency

'total\_revenue': 'sum' # Monetary

}).rename(columns={

'order\_date': 'Recency',

'customer\_id': 'Frequency',

'total\_revenue': 'Monetary'

}).reset\_index()

# -------------------------------

# 3️⃣ Score Each RFM Metric

# -------------------------------

rfm['R\_Score'] = pd.qcut(rfm['Recency'], 5, labels=[5, 4, 3, 2, 1])

rfm['F\_Score'] = pd.qcut(rfm['Frequency'].rank(method='first'), 5, labels=[1, 2, 3, 4, 5])

rfm['M\_Score'] = pd.qcut(rfm['Monetary'], 5, labels=[1, 2, 3, 4, 5])

# Combine RFM Score

rfm['RFM\_Score'] = rfm['R\_Score'].astype(str) + rfm['F\_Score'].astype(str) + rfm['M\_Score'].astype(str)

# -------------------------------

# 4️⃣ Segment Customers

# -------------------------------

def rfm\_segment(row):

if row['RFM\_Score'] == '555':

return 'Champions'

elif row['R\_Score'] == '5':

return 'Loyal Customers'

elif row['R\_Score'] == '1':

return 'Lost Customers'

elif row['F\_Score'] == '5':

return 'Frequent Buyers'

elif row['M\_Score'] == '5':

return 'Big Spenders'

else:

return 'Others'

rfm['Segment'] = rfm.apply(rfm\_segment, axis=1)

# -------------------------------

# 5️⃣ Visualize Segments

# -------------------------------

plt.figure(figsize=(10, 6))

sns.countplot(data=rfm, x='Segment', order=rfm['Segment'].value\_counts().index, palette='Set2')

plt.title("🧠 RFM Customer Segments")

plt.ylabel("Number of Customers")

plt.xlabel("Segment")

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# -------------------------------

# 6️⃣ Summary of Key Insights

# -------------------------------

print("📊 RFM Segment Counts:\n")

print(rfm['Segment'].value\_counts())

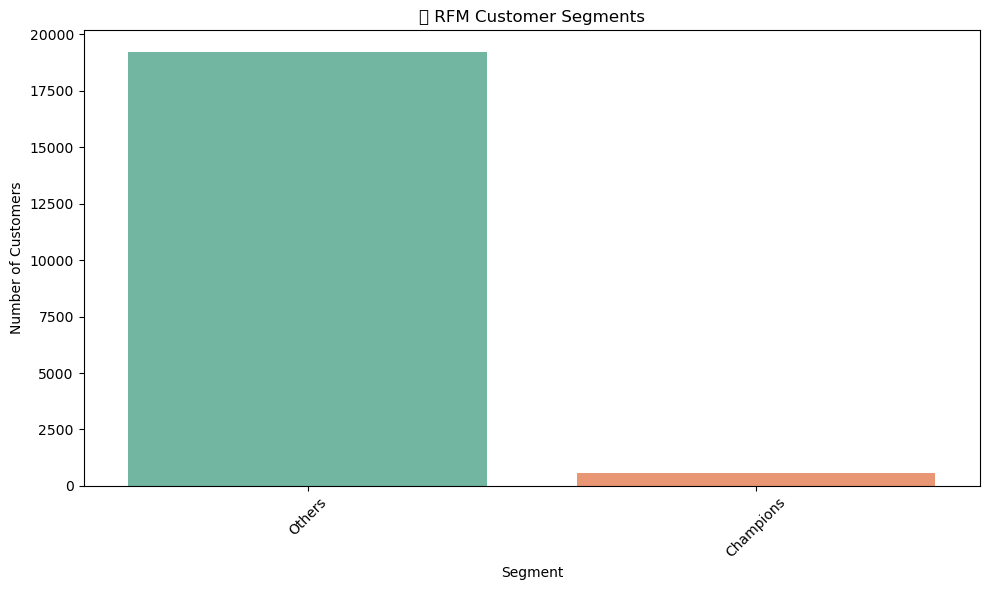
print("\n✅ Recommendations:")

print("• 🎯 Focus campaigns on 'Champions' and 'Big Spenders'.")

print("• 🚀 Reward 'Frequent Buyers' to build loyalty.")

print("• 🧲 Try to re-engage 'Lost Customers' with offers.")

print("• 💡 Use personalized marketing for each segment to improve retention.")



📊 RFM Segment Counts:

Segment

Others 19223

Champions 551

Name: count, dtype: int64

✅ Recommendations:

• 🎯 Focus campaigns on 'Champions' and 'Big Spenders'.

• 🚀 Reward 'Frequent Buyers' to build loyalty.

• 🧲 Try to re-engage 'Lost Customers' with offers.

• 💡 Use personalized marketing for each segment to improve retention.

**Subtask 2: Segment Customers Based on RFM Scores**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load your dataset

df = pd.read\_csv("ecommerce\_data\_final\_cleaned.csv")

# --------------------------------------------

# Step 1: Format Date Column

# --------------------------------------------

# Replace with your actual date column

if 'order\_date' in df.columns:

df['order\_date'] = pd.to\_datetime(df['order\_date'], errors='coerce')

elif 'date' in df.columns:

df['order\_date'] = pd.to\_datetime(df['date'], errors='coerce')

else:

raise Exception("❌ No valid date column found!")

df = df[df['order\_date'].notnull()]

# --------------------------------------------

# Step 2: Prepare RFM Input

# --------------------------------------------

# Rename columns for consistency

df.rename(columns={

'customer\_id': 'CustomerID',

'value [USD]': 'TotalRevenue'

}, inplace=True)

# Remove rows with missing values

df = df.dropna(subset=['CustomerID', 'order\_date', 'TotalRevenue'])

# Reference date (latest date in dataset)

ref\_date = df['order\_date'].max()

# Group data by CustomerID

rfm = df.groupby('CustomerID').agg({

'order\_date': lambda x: (ref\_date - x.max()).days, # Recency

'CustomerID': 'count', # Frequency

'TotalRevenue': 'sum' # Monetary

}).rename(columns={

'order\_date': 'Recency',

'CustomerID': 'Frequency',

'TotalRevenue': 'Monetary'

}).reset\_index()

# --------------------------------------------

# Step 3: Create RFM Score (1–5)

# --------------------------------------------

rfm['R\_Score'] = pd.qcut(rfm['Recency'], 5, labels=[5, 4, 3, 2, 1])

rfm['F\_Score'] = pd.qcut(rfm['Frequency'].rank(method='first'), 5, labels=[1, 2, 3, 4, 5])

rfm['M\_Score'] = pd.qcut(rfm['Monetary'], 5, labels=[1, 2, 3, 4, 5])

# --------------------------------------------

# Step 4: Segment Customers Based on RFM Score

# --------------------------------------------

def assign\_segment(row):

r = int(row['R\_Score'])

f = int(row['F\_Score'])

m = int(row['M\_Score'])

if r == 5 and f == 5 and m == 5:

return 'High-Value Customers'

elif r >= 4 and f >= 4 and m >= 4:

return 'Loyal Customers'

elif r <= 2 and f >= 3 and m >= 3:

return 'At-Risk Customers'

elif r == 5 and f <= 2 and m <= 2:

return 'New Customers'

elif r == 1 and f == 1 and m == 1:

return 'Lost Customers'

elif r >= 4 and f >= 3 and m >= 3:

return 'Potential Loyalists'

else:

return 'Others'

rfm['Customer\_Segment'] = rfm.apply(assign\_segment, axis=1)

# --------------------------------------------

# Step 5: Visualize Customer Segments

# --------------------------------------------

plt.figure(figsize=(10, 6))

sns.countplot(data=rfm, x='Customer\_Segment', order=rfm['Customer\_Segment'].value\_counts().index, palette="Set3")

plt.title("📊 Customer Segments Based on RFM Scores")

plt.ylabel("Number of Customers")

plt.xlabel("Customer Segment")

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# --------------------------------------------

# Step 6: Insights & Marketing Strategy

# --------------------------------------------

print("📋 Customer Segment Counts:\n")

print(rfm['Customer\_Segment'].value\_counts())

print("\n🎯 Marketing Recommendations:")

print("• High-Value Customers → Reward with loyalty perks.")

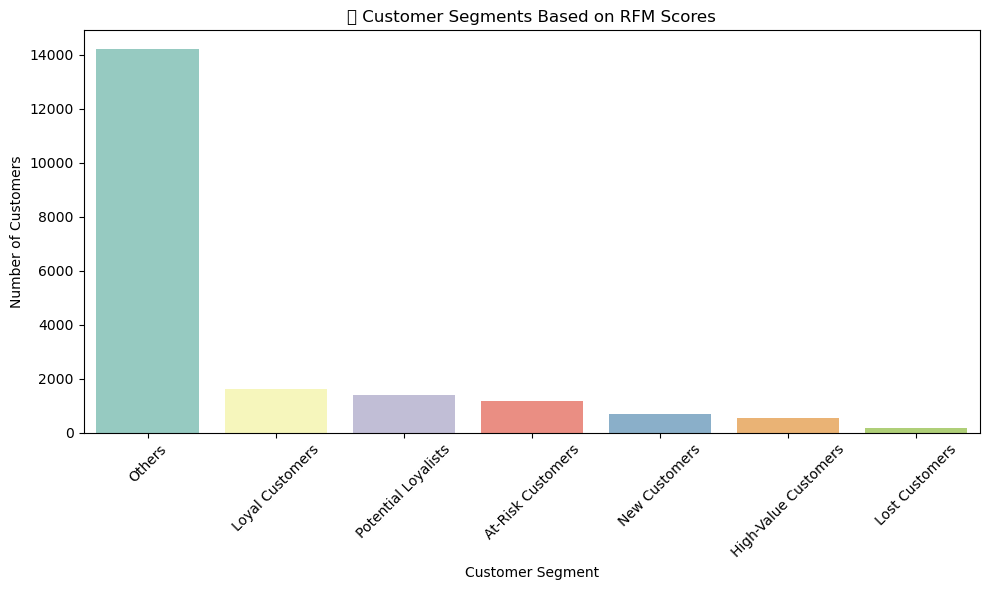
print("• Loyal Customers → Send exclusive product previews.")

print("• At-Risk Customers → Use reminder and discount emails.")

print("• New Customers → Nurture with onboarding offers.")

print("• Lost Customers → Win-back with aggressive offers.")

print("• Potential Loyalists → Encourage with reward points.")



📋 Customer Segment Counts:

Customer\_Segment

Others 14193

Loyal Customers 1625

Potential Loyalists 1385

At-Risk Customers 1165

New Customers 696

High-Value Customers 551

Lost Customers 159

Name: count, dtype: int64

🎯 Marketing Recommendations:

• High-Value Customers → Reward with loyalty perks.

• Loyal Customers → Send exclusive product previews.

• At-Risk Customers → Use reminder and discount emails.

• New Customers → Nurture with onboarding offers.

• Lost Customers → Win-back with aggressive offers.

• Potential Loyalists → Encourage with reward points.

**Subtask 3: Visualize Customer Segments**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Step 1: Load and clean data

df = pd.read\_csv("ecommerce\_data\_final\_cleaned.csv")

# Convert date column to datetime

df['date'] = pd.to\_datetime(df['date'], format='%d/%m/%Y')

# Drop rows with missing customer IDs or values

df.dropna(subset=['customer\_id', 'value [USD]'], inplace=True)

# Step 2: Calculate RFM Metrics

snapshot\_date = df['date'].max() + pd.Timedelta(days=1)

rfm = df.groupby('customer\_id').agg({

'date': lambda x: (snapshot\_date - x.max()).days,

'customer\_id': 'count',

'value [USD]': 'sum'

})

rfm.columns = ['Recency', 'Frequency', 'Monetary']

# Step 3: Create RFM scores (1–5 scale)

rfm['Recency\_Score'] = pd.qcut(rfm['Recency'], 5, labels=[5,4,3,2,1]).astype(int)

rfm['Frequency\_Score'] = pd.qcut(rfm['Frequency'].rank(method='first'), 5, labels=[1,2,3,4,5]).astype(int)

rfm['Monetary\_Score'] = pd.qcut(rfm['Monetary'], 5, labels=[1,2,3,4,5]).astype(int)

rfm['RFM\_Score'] = rfm['Recency\_Score'].astype(str) + rfm['Frequency\_Score'].astype(str) + rfm['Monetary\_Score'].astype(str)

# Step 4: Segment customers based on RFM score

def assign\_segment(row):

r, f, m = row['Recency\_Score'], row['Frequency\_Score'], row['Monetary\_Score']

if r == 5 and f == 5 and m == 5:

return 'High-Value Customers'

elif r >= 4 and f >= 4:

return 'Loyal Customers'

elif r <= 2 and f >= 3 and m >= 3:

return 'At-Risk Customers'

elif r == 5 and f <= 2:

return 'New Customers'

elif r <= 2 and f <= 2 and m <= 2:

return 'Lost Customers'

elif r >= 4 and f >= 3:

return 'Potential Loyalists'

else:

return 'Others'

rfm['Customer\_Segment'] = rfm.apply(assign\_segment, axis=1)

# 🎯 Summary Counts

print("\n📋 Customer Segment Counts:\n")

print(rfm['Customer\_Segment'].value\_counts())

# ----------------------------------------

# 📊 Visualizations

# ----------------------------------------

# 1️⃣ Bar Chart - Segment Distribution

plt.figure(figsize=(10, 6))

segment\_counts = rfm['Customer\_Segment'].value\_counts()

sns.barplot(x=segment\_counts.index, y=segment\_counts.values, palette='viridis')

plt.title("Customer Segment Distribution")

plt.ylabel("Number of Customers")

plt.xlabel("Segment")

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# 2️⃣ Scatter Plot - Recency vs Frequency

plt.figure(figsize=(8, 6))

sns.scatterplot(data=rfm, x='Recency', y='Frequency', hue='Customer\_Segment', palette='tab10')

plt.title("Recency vs Frequency by Segment")

plt.tight\_layout()

plt.show()

# 3️⃣ Heatmap - RFM Score Density

heatmap\_data = rfm.groupby(['Recency\_Score', 'Frequency\_Score']).size().unstack()

plt.figure(figsize=(8, 6))

sns.heatmap(heatmap\_data, cmap="YlGnBu", annot=True, fmt="g")

plt.title("Heatmap of Recency vs Frequency Scores")

plt.tight\_layout()

plt.show()

# 4️⃣ Pie Chart - Segment Proportions

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

rfm['Customer\_Segment'].value\_counts().plot.pie(autopct='%1.1f%%', startangle=140, colors=sns.color\_palette("Set2"))

plt.title("Customer Segment Proportions")

plt.ylabel("")

plt.tight\_layout()

plt.show()

# ✅ Optional: Save the final RFM DataFrame

rfm.to\_csv("rfm\_segmented\_customers.csv")

# 🧠 Marketing Suggestions

print("\n🎯 Marketing Recommendations:")

print("• High-Value Customers → Reward with loyalty perks.")

print("• Loyal Customers → Send exclusive product previews.")

print("• At-Risk Customers → Use reminder and discount emails.")

print("• New Customers → Nurture with onboarding offers.")

print("• Lost Customers → Win-back with aggressive offers.")

print("• Potential Loyalists → Encourage with reward points.")

📋 Customer Segment Counts:

Customer\_Segment

Others 10975

Loyal Customers 3252

Potential Loyalists 1406

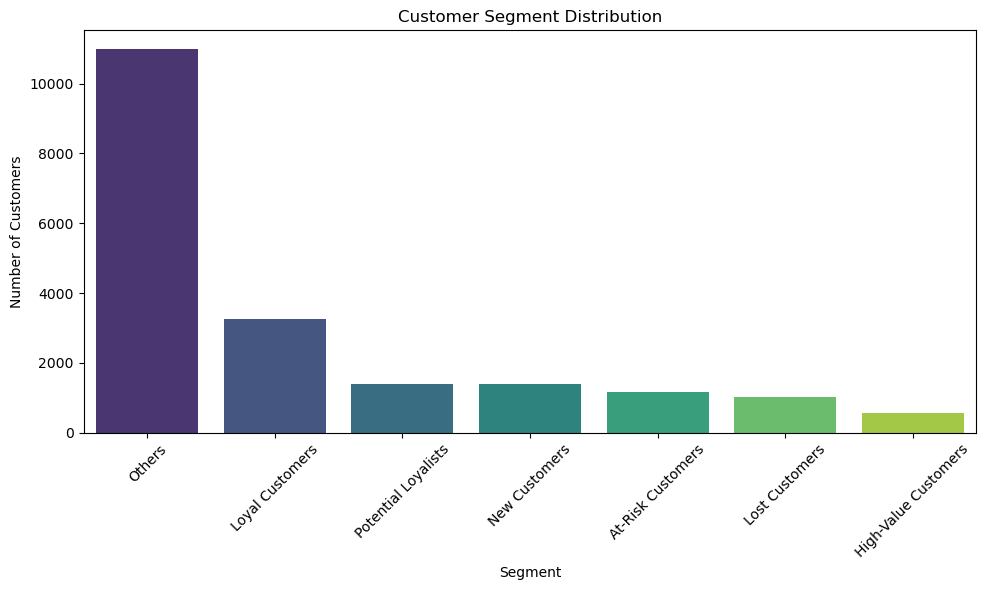
New Customers 1401

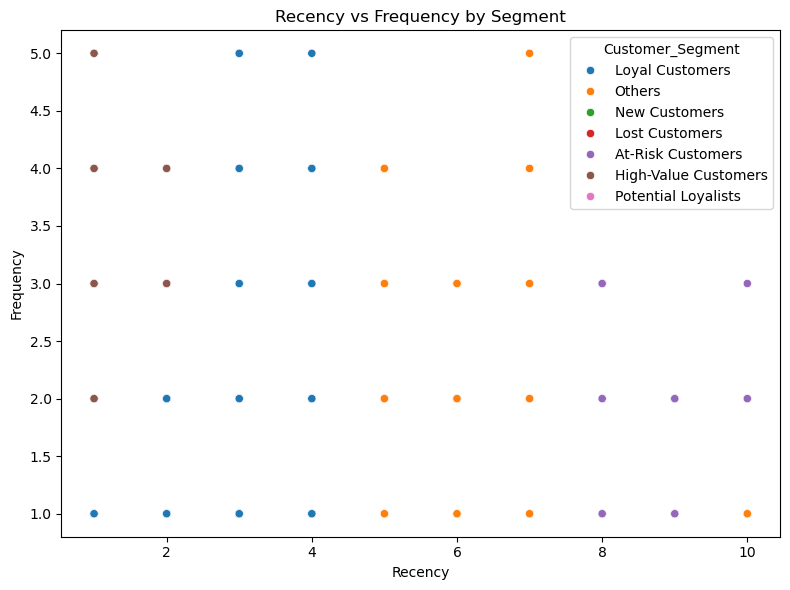
At-Risk Customers 1165

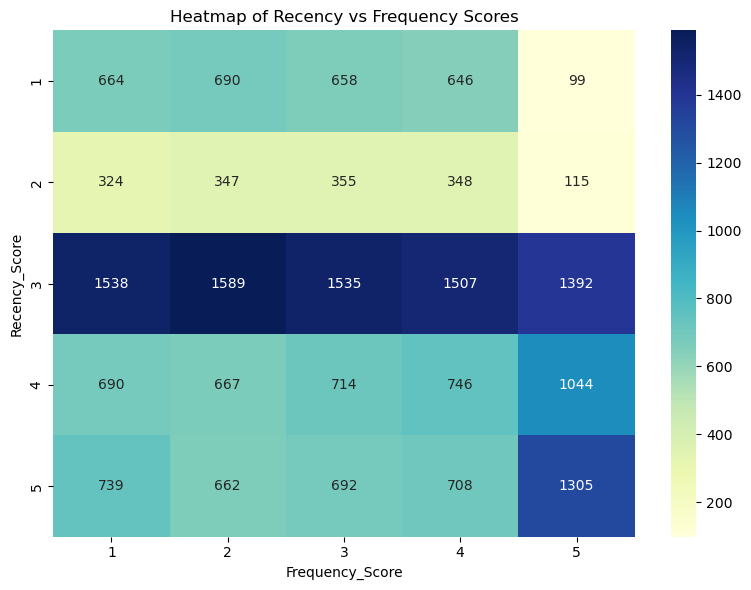
Lost Customers 1024

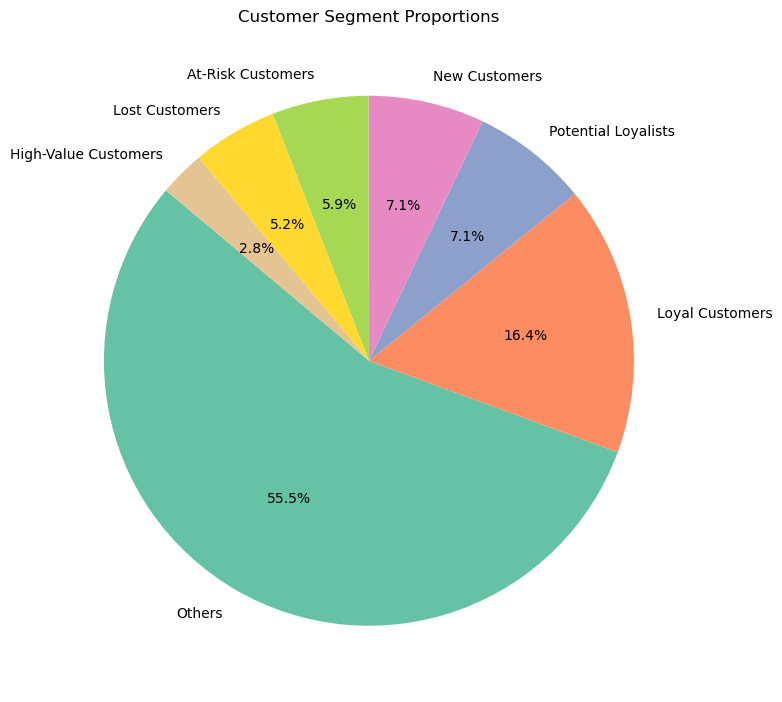
High-Value Customers 551

Name: count, dtype: int64









🎯 Marketing Recommendations:

• High-Value Customers → Reward with loyalty perks.

• Loyal Customers → Send exclusive product previews.

• At-Risk Customers → Use reminder and discount emails.

• New Customers → Nurture with onboarding offers.

• Lost Customers → Win-back with aggressive offers.

• Potential Loyalists → Encourage with reward points.