Data_Segmentation_Visualization

September 21, 2025

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sqlalchemy import create_engine
from urllib.parse import quote_plus

# Set a clean style for the plots
sns.set_style("whitegrid")
plt.style.use("seaborn-v0_8-deep")

# ========================================
# === Database Connection and Data Loading
# ========================================

# Database credentials
user = "root"
password = "Root7878"
host = "localhost"
port = 3306
database = "DataWarehouse"

# Encode password safely (important if it has special chars like @ or $)
password = quote_plus(password)

# Create SQLAlchemy engine
try:
    engine = create_engine(f"mysql+pymysql://{user}:{password}@{host}:{port}/
 ↪{database}")

    # SQL query for product cost segmentation
    product_sql_query = """
    WITH product_segments AS (
        SELECT
            product_key,
            cost,
            CASE WHEN cost < 100 THEN 'Below 100'
                 WHEN cost BETWEEN 100 AND 500 THEN '100-500'
```

```
            WHEN cost BETWEEN 500 AND 1000 THEN '500-1000'
            ELSE 'Above 1000'
        END cost_range
    FROM dim_products
)
SELECT
    cost_range,
    COUNT(product_key) AS total_products
FROM product_segments
GROUP BY cost_range
ORDER BY total_products DESC;
"""
df_products = pd.read_sql(product_sql_query, engine)

# SQL query for customer segmentation
customer_sql_query = """
WITH customer_spending AS (
    SELECT
        c.customer_key,
        SUM(f.sales_amount) AS total_spending,
        TIMESTAMPDIFF(MONTH, MIN(order_date), MAX(order_date)) AS lifespan
    FROM fact_sales f
    LEFT JOIN dim_customers c
    ON f.customer_key = c.customer_key
    GROUP BY c.customer_key
)
SELECT
    customer_segment,
    COUNT(customer_key) AS total_customers
FROM (
    SELECT
        customer_key,
        CASE WHEN lifespan > 12 AND total_spending > 5000 THEN 'VIP'
            WHEN lifespan > 12 AND total_spending <= 5000 THEN 'Regular'
            ELSE 'New'
        END AS customer_segment
    FROM customer_spending) t
GROUP BY customer_segment
ORDER BY total_customers DESC;
"""
df_customers = pd.read_sql(customer_sql_query, engine)

print("Product Segmentation DataFrame Head:")
print(df_products.head())
print("-" * 50)
print("Customer Segmentation DataFrame Head:")
print(df_customers.head())
```

```
    print("-" * 50)

except Exception as e:
    print(f"Error connecting to the database or loading data: {e}")
    print("Please ensure your database credentials are correct and the database␣
 ↪is running.")
    df_products = pd.DataFrame()
    df_customers = pd.DataFrame()
```

```
Product Segmentation DataFrame Head:
   cost_range  total_products
0   Below 100             110
1     100-500             101
2    500-1000              45
3  Above 1000              39
--------------------------------------------------
Customer Segmentation DataFrame Head:
  customer_segment  total_customers
0              New            15112
1          Regular             1809
2              VIP             1563
--------------------------------------------------
```

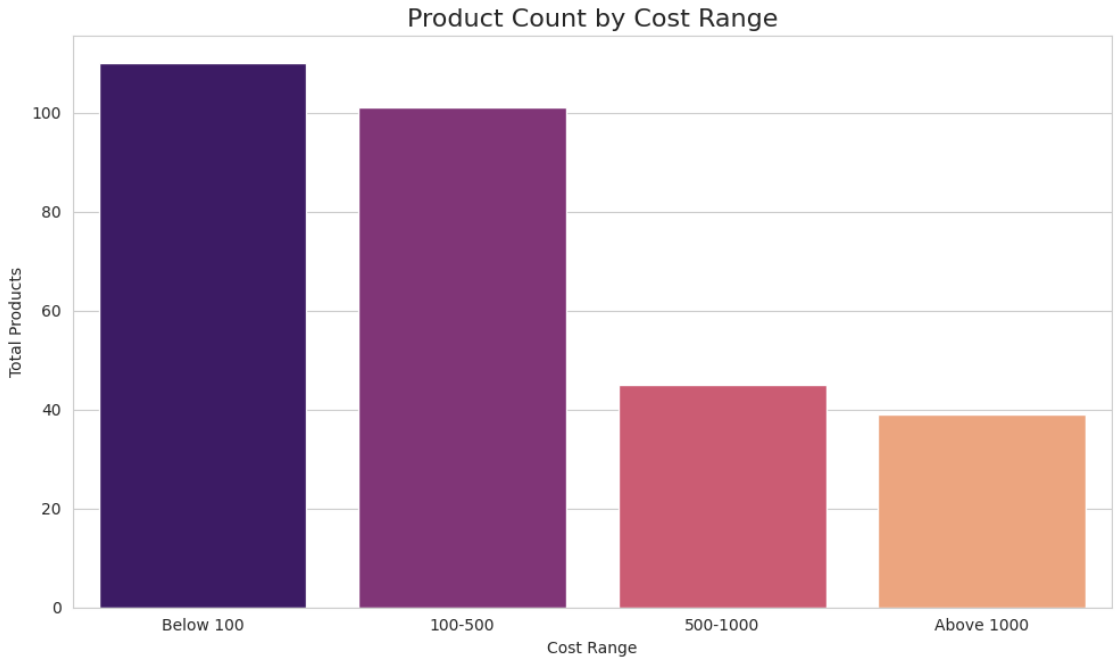
[ ]:
```
# ================================
# === Data Visualizations
# ================================
```

[2]:
```
if not df_products.empty and not df_customers.empty:

    # --- 1. Product Count by Cost Range ---
    plt.figure(figsize=(10, 6))
    sns.barplot(
        x='cost_range',
        y='total_products',
        data=df_products,
        palette='magma',
        hue='cost_range',
        legend=False
    )
    plt.title('Product Count by Cost Range', fontsize=16)
    plt.xlabel('Cost Range')
    plt.ylabel('Total Products')
    plt.tight_layout()
    plt.show()

else:
```

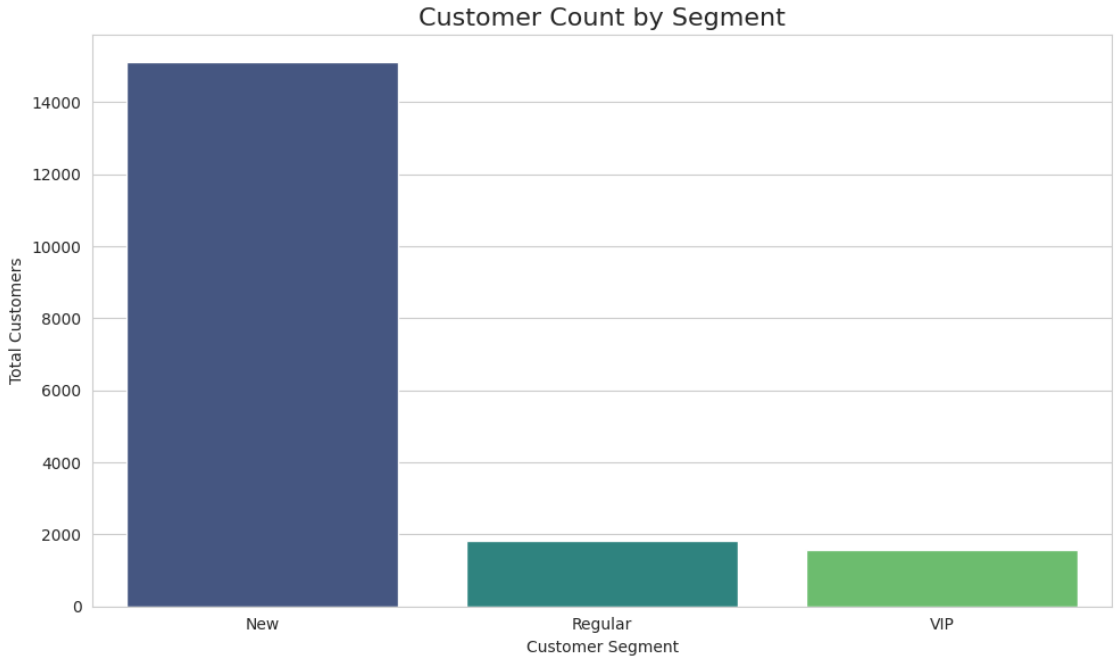
```
    print("One or both DataFrames are empty. No visualizations will be␣
 ↪generated.")
```



[3]:
```
if not df_products.empty and not df_customers.empty:

    # --- 2. Customer Count by Segment ---
    plt.figure(figsize=(10, 6))
    sns.barplot(
        x='customer_segment',
        y='total_customers',
        data=df_customers,
        palette='viridis',
        hue='customer_segment',
        legend=False
    )
    plt.title('Customer Count by Segment', fontsize=16)
    plt.xlabel('Customer Segment')
    plt.ylabel('Total Customers')
    plt.tight_layout()
    plt.show()

else:
    print("One or both DataFrames are empty. No visualizations will be␣
 ↪generated.")
```


Customer Count by Segment
Total Customers
14000
12000
10000
8000
6000
4000
2000
0
New
Regular
VIP
Customer Segment