```
# -*- coding: utf-8 -*-
"""chips_analysis.ipynb

Automatically generated by Colab.

Original file is located at
    https://colab.research.google.com/drive/1-y2Jh-0zg9aLib-
Qcx9Pu4CyWC8yVkgg
"""

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re

# Step 1: Load the data
# Ensure QVI_transaction_data.xlsx and QVI_purchase_behaviour.csv are in
the same folder
transactions = pd.read_excel('QVI_transaction_data.xlsx')
customers = pd.read_csv('QVI_purchase_behaviour.csv')

# Step 2: Explore transaction data
print("First 5 rows of Transaction Data:")
print(transactions.head())
print("\nTransaction Data Info:")
print(transactions.info())
print("\nTransaction Data Summary Statistics:")
print(transactions.describe())

# Step 3: Clean transaction data
# Check for missing values
print("\nMissing Values in Transactions:")
print(transactions.isnull().sum())

# Convert DATE from Excel serial format to datetime (e.g., 43390 -> 2018-
07-01)
transactions['DATE'] = pd.to_datetime(transactions['DATE'], unit='D',
origin='1899-12-30')
print("\nData type of DATE column after conversion:")
print(transactions['DATE'].dtype)

# Filter for chip products using positive keywords
chip_keywords = ['Chip', 'Chips', 'Crisps', 'Crinkle', 'Tortilla',
'Twisties', 'Cheezels', 'Corn']
transactions['IS_CHIP'] =
transactions['PROD_NAME'].str.contains('|'.join(chip_keywords),
case=False, na=False)
transactions_cleaned = transactions[transactions['IS_CHIP']].copy()

# Handle outliers: Remove transactions with PROD_QTY > 5 (e.g., bulk
purchases)
print("\nPROD_QTY Distribution Before Cleaning:")
print(transactions_cleaned['PROD_QTY'].describe())
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
sns.boxplot(x=transactions_cleaned['PROD_QTY'])
plt.title('Box Plot of PROD_QTY')
plt.subplot(1, 2, 2)
sns.histplot(transactions_cleaned['PROD_QTY'], bins=20, kde=True)
plt.title('Histogram of PROD_QTY')
plt.tight_layout()
plt.savefig('prod_qty_distribution.png')
plt.close()

transactions_cleaned =
transactions_cleaned[transactions_cleaned['PROD_QTY'] <= 5].copy()
print("\nShape after removing outliers:", transactions_cleaned.shape)

# Step 4: Derive features
# Extract pack size (e.g., '175g' -> 175)
def extract_pack_size(name):
    match = re.search(r'(\d+)(g|G)', name)
    return int(match.group(1)) if match else None

transactions_cleaned['PACK_SIZE'] =
transactions_cleaned['PROD_NAME'].apply(extract_pack_size)
print("\nMissing PACK_SIZE values:",
transactions_cleaned['PACK_SIZE'].isnull().sum())
# Drop rows with missing PACK_SIZE
transactions_cleaned = transactions_cleaned.dropna(subset=['PACK_SIZE'])

# Extract and standardize brand
transactions_cleaned['BRAND'] =
transactions_cleaned['PROD_NAME'].str.split().str[0]
transactions_cleaned['BRAND'] = transactions_cleaned['BRAND'].replace({
    'Dorito': 'Doritos', 'Smith': 'Smiths', 'Infzns': 'Infuzions', 'WW':
'Woolworths',
    'RRD': 'Red Rock Deli', 'GrnWves': 'Grain Waves', 'Snbts':
'Sunbites', 'Natural': 'Natural Chip Co'
})
print("\nUnique Brands:", transactions_cleaned['BRAND'].unique())

# Step 5: Explore customer data
print("\nFirst 5 rows of Customer Data:")
print(customers.head())
print("\nCustomer Data Info:")
print(customers.info())
print("\nMissing Values in Customers:")
print(customers.isnull().sum())

# Step 6: Merge datasets
merged_data = pd.merge(transactions_cleaned, customers,
on='LYLTY_CARD_NBR', how='left')
print("\nMissing Values in Merged Data:")
print(merged_data.isnull().sum())
merged_data = merged_data.dropna(subset=['LIFESTAGE',
'PREMIUM_CUSTOMER'])

# Step 7: Calculate metrics
total_sales_by_segment = merged_data.groupby(['LIFESTAGE',
'PREMIUM_CUSTOMER'])['TOT_SALES'].sum().reset_index()
purchase_frequency_by_segment = merged_data.groupby(['LIFESTAGE',
'PREMIUM_CUSTOMER'])['TXN_ID'].count().reset_index()
purchase_frequency_by_segment.rename(columns={'TXN_ID':
'TRANSACTION_COUNT'}, inplace=True)
```

```
merged_data['AVG_SPEND_PER_UNIT'] = merged_data['TOT_SALES'] /
merged_data['PROD_QTY']
average_spend_by_segment = merged_data.groupby(['LIFESTAGE',
'PREMIUM_CUSTOMER'])['AVG_SPEND_PER_UNIT'].mean().reset_index()
brand_sales =
merged_data.groupby('BRAND')['TOT_SALES'].sum().reset_index()

# Step 8: Visualize insights
# Plot 1: Total sales by customer segment
plt.figure(figsize=(14, 6))
sns.barplot(data=total_sales_by_segment, x='LIFESTAGE', y='TOT_SALES',
hue='PREMIUM_CUSTOMER')
plt.title('Total Chip Sales by Lifestage and Premium Customer Segment')
plt.xlabel('Lifestage')
plt.ylabel('Total Sales ($)')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Customer Type')
plt.tight_layout()
plt.show()

# Plot 2: Pack size distribution
plt.figure(figsize=(10, 6))
sns.histplot(data=merged_data, x='PACK_SIZE', bins=20, edgecolor='black')
plt.title('Distribution of Chip Pack Sizes')
plt.xlabel('Pack Size (grams)')
plt.ylabel('Number of Transactions')
plt.tight_layout()
plt.show()

# Plot 3: Top 10 brands by sales
plt.figure(figsize=(10, 6))
sns.barplot(data=brand_sales.sort_values('TOT_SALES',
ascending=False).head(10), x='TOT_SALES', y='BRAND')
plt.title('Top 10 Chip Brands by Total Sales')
plt.xlabel('Total Sales ($)')
plt.ylabel('Brand')
plt.tight_layout()
plt.show()

# Step 9: Save outputs
merged_data.to_csv('cleaned_merged_data.csv', index=False)
total_sales_by_segment.to_csv('total_sales_by_segment.csv', index=False)
purchase_frequency_by_segment.to_csv('purchase_frequency_by_segment.csv',
index=False)
average_spend_by_segment.to_csv('average_spend_by_segment.csv',
index=False)
brand_sales.to_csv('brand_sales.csv', index=False)

print("Analysis complete! CSVs and PNGs saved.")
print("To submit to Quantium, run this in Jupyter Notebook and export as
PDF (File > Download as > PDF via LaTeX).")
print("Ensure pandoc and MiKTeX are installed for PDF export.")
```