BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH

BÁO CÁO MÔN
PHÂN TÍCH DỮ LIỆU

TÊN ĐỀ TÀI
XU HƯỚNG MUA SẮM CỦA KHÁCH HÀNG
(CUSTOMER SHOPPING TREND)

Giảng viên hướng dẫn: **Thầy Hồ Hướng Thiên**
Lớp: DH22IM01

Nhóm sinh viên thực hiện:
1. Võ Tấn Huy – 2254052031
2. Hồ Hoàn Hảo – 2254052025
3. Nguyễn Mai Hân – 2251010025

TP. HỒ CHÍ MINH, tháng 12 năm 2024

MỤC LỤC

MỞ ĐẦU

Trong bối cảnh cạnh tranh khốc liệt như hiện nay, việc nắm bắt và hiểu rõ xu hướng mua sắm của khách hàng là yếu tố quyết định sự thành bại của doanh nghiệp. Bằng cách phân tích sở thích, hành vi tiêu dùng và các yếu tố ảnh hưởng đến quyết định mua hàng, doanh nghiệp không chỉ đưa ra được những chiến lược kinh doanh hiệu quả mà còn xây dựng mối quan hệ bền vững với khách hàng.

Chính vì thế mục tiêu chính của đề tài này là phân tích xu hướng mua sắm của khách hàng. Từ đó, bản báo cáo này sẽ giúp chúng ta dự đoán được xu hướng mua sắm của khách hàng và dựa trên đó để đưa ra những chính sách giữ chân cũng như thu hút khách hàng. Như vậy, doanh nghiệp không chỉ duy trì sự cạnh tranh mà còn đáp ứng hiệu quả nhu cầu ngày càng cao của khách hàng, từ đó phát triển bền vững và tăng trưởng lợi nhuận.

Đề tài này sẽ sử dụng dữ liệu về những xu hướng mua sắm của khách với các dữ liệu như: tuổi, giới tính, mặt hàng đã mua, danh mục, số tiền mua, địa điểm, kích thước, màu sắc, mùa, xếp hạng đánh giá, trạng thái đăng ký, loại hình vận chuyển, chiết khấu được áp dụng, mã khuyến mại đã sử dụng, lần mua trước, phương thức thanh toán, tần suất mua hàng. Sau đó, tiến hành áp dụng các thuật toán và phân tích dữ liệu để phân tích xu hướng mua sắm của khách hàng.

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

1.1. Giới thiệu tổng quan

Phân tích xu hướng mua sắm của khách hàng là một trong những yếu tố quyết định doanh nghiệp thành công hay không. Bằng cách thu thập và phân tích dữ liệu, doanh nghiệp có thể xây dựng chân dung khách hàng chi tiết, nhờ vào đó đưa ra những quyết định kinh doanh sáng suốt. Việc hiểu rõ hành vi mua sắm của khách hàng giúp doanh nghiệp tạo ra những sản phẩm, dịch vụ phù hợp, nâng cao trải nghiệm mua sắm và tăng cường lòng trung thành của khách hàng.

1.2. Lý do chọn đề tài

Để có thể tồn tại trong thị trường kinh doanh đầy rẫy sự cạnh tranh thì việc hiểu rõ được hành vi và nhu cầu mua sắm của khách hàng là một trong những chìa khóa quan trọng. Hiểu rõ được những mong muốn của khách hàng giúp doanh nghiệp đưa ra được những chiến lược quan trọng để lôi kéo và giữ chân khách hàng. Từ đó duy trì mối quan hệ với khách hàng và đi kèm với sự phát triển kinh doanh của công ty.

1.3. Tóm tắt mô hình Dự đoán xu hướng mua sắm của khách hàng

1.3.1. Mục tiêu dự án

Mục tiêu chính của dự án là xây dựng một mô hình phân tích xu hướng mua sắm của khách hàng. Mô hình dự đoán này sẽ giúp chúng ta xác định rõ hơn về hành vi và nhu cầu của người tiêu dùng, từ đó tối ưu hóa chiến lược marketing, sản phẩm và dịch vụ. Không những thế đề tài này không chỉ hỗ trợ dự báo nhu cầu, cải thiện trải nghiệm khách hàng, mà còn giúp doanh nghiệp tối ưu hóa tồn kho, quản lý chiến dịch khuyến mãi và nâng cao hiệu quả kinh doanh.

1.3.2. Phạm vi dự án

Dự án sử dụng dữ liệu về những xu hướng mua sắm của khách với các dữ liệu như: tuổi, giới tính, mặt hàng đã mua, danh mục, số tiền mua, địa điểm, kích thước, màu sắc, mùa, xếp hạng đánh giá, trạng thái đăng ký, loại hình vận chuyển, chiết khấu được áp dụng, mã khuyến mại đã sử dụng, lần mua trước, phương thức thanh toán, tần suất mua hàng. Và sẽ tiến hành áp dụng các thuật toán và phân tích dữ liệu để phân tích xu hướng mua sắm của khách hàng và đánh giá hiệu suất của mô hình.

1.3.3. Cấu trúc xây dựng mô hình

Mô hình dự đoán trên được xây dựng dựa trên các phần bao gồm Tìm hiểu vấn đề, Xác định mục tiêu, Thu thập dữ liệu, Phân tích dữ liệu, Tiền xử lý dữ liệu, Xây dựng mô hình, Đánh giá hiệu suất, Kết luận và Áp dụng mô hình vào hệ thống.

CHƯƠNG 2: KIẾN TRÚC VÀ GIẢI PHÁP

2.1. Kiến trúc mô hình

Sử dụng mô hình học máy để dự đoán xu hướng mua sắm tiếp theo của người tiêu dùng, với kiến trúc mô hình được xây dựng bài bản, bao gồm các giai đoạn từ thu thập dữ liệu, tiền xử lý, trực quan hóa đến huấn luyện và tối ưu hóa mô hình.

2.2. Giải pháp

2.2.1. Thu thập dữ liệu

Trước khi bắt đầu xây dựng mô hình phân tích xu hướng mua sắm của khách hàng, bước đầu tiên và không thể thiếu là thu thập dữ liệu liên quan đến hành vi mua sắm và đặc điểm của khách hàng. Việc này đảm bảo rằng chúng ta có nguồn dữ liệu đầy đủ và chính xác để phát triển mô hình phân tích hiệu quả, từ đó hiểu rõ hơn về các xu hướng và động lực thúc đẩy quyết định mua hàng. Tuy nhiên, do dữ liệu thực tế về khách hàng trong lĩnh vực bán lẻ thường khó tiếp cận vì các lý do bảo mật, nhóm chúng em đã sử dụng một bộ dữ liệu mẫu từ nền tảng Kaggle để thực hiện nghiên cứu và phân tích.

Dataset: 

2.2.2. Phân tích và đánh giá dữ liệu

Tiến hành đánh giá sơ lược qua dữ liệu bằng Jupyter Notebook

Cài đặt các thư viện hỗ trợ cho việc tải dữ liệu lên và phân tích dữ liệu.

```
import pandas as pd
import numpy as np
```

Kết nối với Jupyter Notebook để tải dữ liệu lên sau đó đọc dữ liệu và xem sơ lược dữ liệu

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size	Color	Season	Review Rating	Subscription Status	Shipping Type	Discount Applied	Promo Code Used	Previou Purchase
0	1	55.0	Male	Blouse	Clothing	53.0	Kentucky	L	Gray	Winter	3,1	Yes	Express	Yes	Yes	14
1	2	19.0	Male	Sweater	Clothing	64.0	Maine	L	Maroon	Winter	3,1	Yes	Express	Yes	Yes	2
2	3	NaN	Female	Jeans	Clothing	73.0	Massachusetts	S	Maroon	Spring	3,1	Yes	Free Shipping	Yes	Yes	23
3	4	21.0	Male	Sandals	Footwear	90.0	Rhode Island	M	Maroon	Spring	3,5	Yes	Next Day Air	Yes	Yes	49
4	5	110.0	Male	Blouse	Clothing	49.0	Oregon	M	Turquoise	Spring	2,7	Yes	Free Shipping	Yes	Yes	31
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
14095	13895	56.0	Female	Blouse	Clothing	59.0	South Carolina	M	Beige	Fall	2,7	Yes	Free Shipping	Yes	Yes	17
14096	13896	45.0	Female	Coat	Outerwear	48.0	West Virginia	S	Olive	Winter	3,6	No	Express	Yes	Yes	4
14097	13897	105.0	Female	Jewelry	Accessories	81.0	North Dakota	S	Gray	Summer	4,2	No	Next Day Air	No	No	49
14098	13898	104.0	Male	Jewelry	NaN	71.0	Alaska	S	Gray	Spring	2,8	No	NaN	Yes	Yes	48
14099	13899	61.0	Male	Sandals	Footwear	46.0	Ohio	L	Lavender	Fall	3,2	No	Next Day Air	No	No	19

14100 rows × 18 columns

Sơ lược về Dataset

Xem kích cỡ của dataset

```
dt.shape
```

```
(14100, 18)
```

Dữ liệu có 14100 dòng và 18 cột

Kiểm tra kiểu dữ liệu của các thuộc tính

```
dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14100 entries, 0 to 14099
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Customer ID             14100 non-null  int64
 1   Age                     13906 non-null  float64
 2   Gender                  14001 non-null  object
 3   Item Purchased          13926 non-null  object
 4   Category                13918 non-null  object
 5   Purchase Amount (USD)   13915 non-null  float64
 6   Location                13907 non-null  object
 7   Size                    13896 non-null  object
 8   Color                   13926 non-null  object
 9   Season                  13894 non-null  object
 10  Review Rating           13896 non-null  object
 11  Subscription Status     13893 non-null  object
 12  Shipping Type           13913 non-null  object
 13  Discount Applied        13944 non-null  object
 14  Promo Code Used         13879 non-null  object
 15  Previous Purchases      13914 non-null  float64
 16  Payment Method          13902 non-null  object
 17  Frequency of Purchases  13913 non-null  object
dtypes: float64(3), int64(1), object(14)
memory usage: 1.9+ MB
```

Thuộc tính dữ liệu

Bao gồm có: 3 thuộc tính kiểu dữ liệu float64, 1 thuộc tính kiểu dữ liệu int64, và 14 thuộc tính kiểu dữ liệu object

2.2.3. Tiền xử lý dữ liệu

- Kiểm tra số lượng dữ liệu bị trùng lặp và loại bỏ

```
dt.duplicated().sum()
```

```
117
```

```
dt.drop_duplicates(inplace=True)
dt
```

- Xóa thuộc tính không cần thiết

```
col = ['Customer ID', 'Discount Applied','Promo Code Used','Payment Method', 'Location', 'Size', 'Color',
       'Season', 'Review Rating', 'Subscription Status', 'Shipping Type']
dt= dt.drop(col, axis=1)
dt
```

	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Previous Purchases	Frequency of Purchases
0	55.0	Male	Blouse	Clothing	53.0	14.0	Fortnightly
1	19.0	Male	Sweater	Clothing	64.0	2.0	Fortnightly
2	NaN	Female	Jeans	Clothing	73.0	23.0	Weekly
3	21.0	Male	Sandals	Footwear	90.0	49.0	Weekly
4	110.0	Male	Blouse	Clothing	49.0	31.0	Annually
...	...	...	...	...	...	...	...
14095	56.0	Female	Blouse	Clothing	59.0	17.0	Fortnightly
14096	45.0	Female	Coat	Outerwear	48.0	4.0	Fortnightly
14097	105.0	Female	Jewelry	Accessories	81.0	49.0	Weekly
14098	104.0	Male	Jewelry	NaN	71.0	48.0	Every 3 Months
14099	61.0	Male	Sandals	Footwear	46.0	19.0	Every 3 Months

13983 rows × 7 columns

Sau khi thực hiện xóa dữ liệu bị trùng lặp và các thuộc tính không cần thiết, dữ liệu còn lại 13983 hàng và 7 cột.

- Kiểm tra xem có thuộc tính nào bị NaN hoặc Null dữ liệu

```
dt.isnull().sum()
```

```
Age                          192
Gender                        99
Item Purchased               173
Category                     179
Purchase Amount (USD)        184
Previous Purchases           186
Frequency of Purchases       185
dtype: int64
```

```
dt.isna().sum()
```

```
Age                          192
Gender                        99
Item Purchased               173
Category                     179
Purchase Amount (USD)        184
Previous Purchases           186
Frequency of Purchases       185
dtype: int64
```

- Xử lí dữ liệu NaN hoặc Null:

```
# Xử lý dữ liệu khuyết từng cột
# 1. Điền bằng trung bình hoặc trung vị cho dữ liệu số
dt['Age'] = dt['Age'].fillna(round(dt['Age'].median()))
dt['Purchase Amount (USD)'] = dt['Purchase Amount (USD)'].fillna(round(dt['Purchase Amount (USD)'].mean()))
dt['Previous Purchases'] = dt['Previous Purchases'].fillna(round(dt['Previous Purchases'].mean()))

# 2. Điền bằng giá trị thường xuyên nhất (mode) cho dữ liệu phân loại
categorical_cols = ['Gender', 'Category', 'Item Purchased']
for col in categorical_cols:
    if col in dt.columns:
        dt[col] = dt[col].fillna(dt[col].mode()[0])

# Ít thường xuyên cho Frequency of Purchases
# Lặp qua các cột
categorical_cols1 = ['Frequency of Purchases']
for col in categorical_cols1:
    if col in dt.columns:
        # Tính giá trị ít xuất hiện nhất trong mỗi cột
        value_counts = dt[col].value_counts()
        least_frequent_value = value_counts.idxmin()

        # Thay thế 'Unknown' bằng giá trị ít xuất hiện nhất
        dt[col] = dt[col].replace('Unknown', least_frequent_value)

        # Điền các giá trị thiếu (NaN) trong cột 'Frequency of Purchases' bằng giá trị ít xuất hiện nhất
        dt['Frequency of Purchases'] = dt['Frequency of Purchases'].fillna(least_frequent_value)

# Kiểm tra lại dữ liệu khuyết sau xử lý
print("Kiểm tra dữ liệu khuyết sau xử lý:")
print(dt.isnull().sum())
```

```
Kiểm tra dữ liệu khuyết sau xử lý:
Customer ID                 0
Age                         0
Gender                      0
Item Purchased              0
Category                    0
Purchase Amount (USD)       0
Previous Purchases          0
Frequency of Purchases      0
dtype: int64
```

- Kiểm tra dữ liệu vượt ngưỡng xác định

```python
# Lọc các giá trị lớn hơn 80 hoặc nhỏ hơn 16 trong cột 'Age'
age_over = dt[(dt['Age'] > 80) | (dt['Age'] < 16)]

# Kiểm tra và in ra các cột 'Customer ID' và 'Age'
if not age_over.empty:
    print("Cột 'Age' có giá trị vượt ngưỡng xác định.")
    print(age_over[['Age']])
else:
    print("Cột 'Age' không có giá trị vượt ngưỡng xác định")
```

```
Cột 'Age' có giá trị vượt ngưỡng xác định.
         Age
4      110.0
7      108.0
11     104.0
13     103.0
24     105.0
...      ...
14068  101.0
14072  105.0
14081  102.0
14097  105.0
14098  104.0

[2415 rows x 1 columns]
```

Có 2415 dòng có dữ liệu vượt ngưỡng

- Xử lí thay thế các giá trị vượt ngưỡng

```
# Tính giá trị trung bình của cột 'Age'
mean_age = round(dt['Age'].mean())

# Xác định điều kiện trước khi thay thế
condition = (dt['Age'] > 80) | (dt['Age'] < 16)

# Thay thế các giá trị lớn hơn 80 hoặc nhỏ hơn 10 bằng giá trị trung bình
dt.loc[condition, 'Age'] = mean_age

# Hiển thị các giá trị sau khi thay thế
print("Các giá trị sau khi thay thế:")
print(dt[condition][['Age']])
```

```
Các giá trị sau khi thay thế:
        Age
4      55.0
7      55.0
11     55.0
13     55.0
24     55.0
...     ...
14068  55.0
14072  55.0
14081  55.0
14097  55.0
14098  55.0

[2415 rows x 1 columns]
```

- Mã hóa cột ‘Frequency of Purchases’ để thuận tiện cho các quy trình tiếp theo

```
from sklearn.preprocessing import LabelEncoder
#Label Encoding cho cột 'Frequency of Purchases'
le_item = LabelEncoder()
dt['Frequency of Purchases'] = le_item.fit_transform(dt['Frequency of Purchases'])

#Xem các lớp gốc và giá trị mã hóa
print("Lớp gốc và giá trị mã hóa:")
for idx, label in enumerate(le_item.classes_):
    print(f"{label}: {idx}")

# Kiểm tra kết quả
dt
```

```
Lớp gốc và giá trị mã hóa:
Annually: 0
Bi-Weekly: 1
Every 3 Months: 2
Fortnightly: 3
Monthly: 4
Quarterly: 5
Weekly: 6
```

	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Previous Purchases	Frequency of Purchases
0	55.0	Male	Blouse	Clothing	53.0	14.0	3
1	19.0	Male	Sweater	Clothing	64.0	2.0	3
2	50.0	Female	Jeans	Clothing	73.0	23.0	6
3	21.0	Male	Sandals	Footwear	90.0	49.0	6
4	55.0	Male	Blouse	Clothing	49.0	31.0	0
...	...	...	...	...	...	...	...
14095	56.0	Female	Blouse	Clothing	59.0	17.0	3
14096	45.0	Female	Coat	Outerwear	48.0	4.0	3
14097	55.0	Female	Jewelry	Accessories	81.0	49.0	6
14098	55.0	Male	Jewelry	Clothing	71.0	48.0	2
14099	61.0	Male	Sandals	Footwear	46.0	19.0	2

13983 rows × 7 columns

- Di chuyển cột ‘Item Purchased’ xuống cuối hàng để dễ dàng tách dữ liệu

```
df = pd.DataFrame(dt)
# Giả sử df là DataFrame ban đầu
column_to_move = 'Item Purchased'

# Đưa cột 'Item Purchased' ra khỏi DataFrame và lưu lại
df = df[[col for col in df.columns if col != column_to_move] + [column_to_move]]
```

	Age	Gender	Category	Purchase Amount (USD)	Previous Purchases	Frequency of Purchases	Item Purchased
0	55.0	Male	Clothing	53.0	14.0	3	Blouse
1	19.0	Male	Clothing	64.0	2.0	3	Sweater
2	50.0	Female	Clothing	73.0	23.0	6	Jeans
3	21.0	Male	Footwear	90.0	49.0	6	Sandals
4	55.0	Male	Clothing	49.0	31.0	0	Blouse
...	...	...	...	...	...	...	...
14095	56.0	Female	Clothing	59.0	17.0	3	Blouse
14096	45.0	Female	Outerwear	48.0	4.0	3	Coat
14097	55.0	Female	Accessories	81.0	49.0	6	Jewelry
14098	55.0	Male	Clothing	71.0	48.0	2	Jewelry
14099	61.0	Male	Footwear	46.0	19.0	2	Sandals

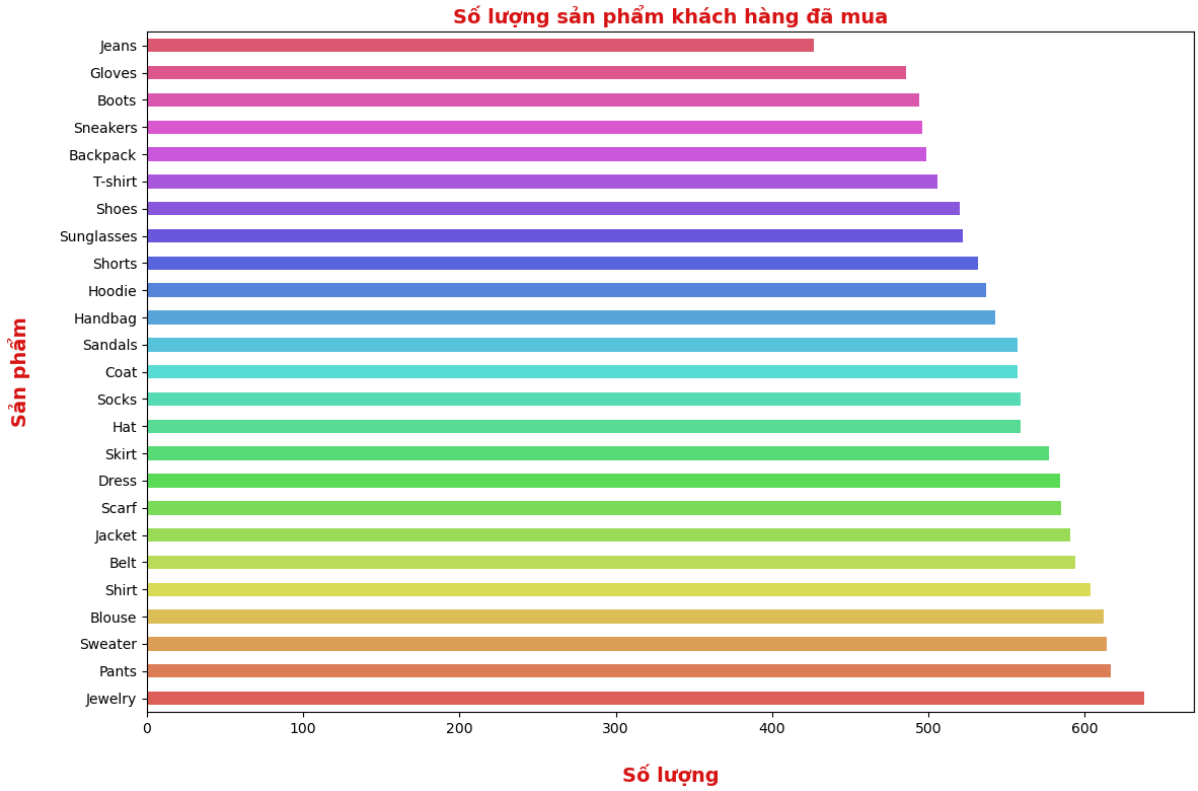
13983 rows × 7 columns

Bộ dữ liệu cuối cùng sau khi tiền xử lý dữ liệu

2.2.4. Trực quan hóa dữ liệu

- **Sử dụng biểu đồ thanh đơn nằm ngang cho dữ liệu 'Item Purchased'**

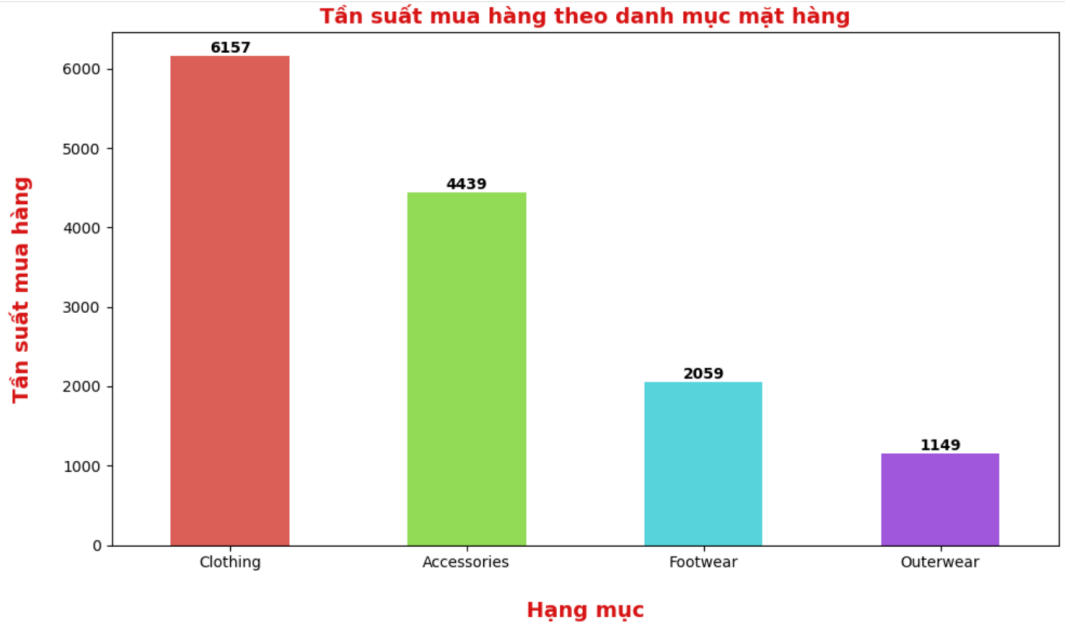
```
#Item Purchased - biểu đồ thanh đơn nằm ngang
import matplotlib.pyplot as plt
import seaborn as sns

# Tính toán số lượng mỗi sản phẩm đã mua
item_counts = dt['Item Purchased'].value_counts()
# Tạo biểu đồ thanh đơn nằm ngang
plt.figure(figsize=(12, 8))
#Tạo biểu đồ có yếu tố xen kẽ màu sắc
item_counts.plot(kind='barh', color=sns.color_palette("hls", len(item_counts)))
# Thêm tiêu đề, nhãn trục
plt.title('Số lượng sản phẩm khách hàng đã mua',weight="bold",color="#D71313", fontsize=14)
plt.xlabel('Số lượng',weight="bold", color="#D71313", fontsize=14, labelpad=20)
plt.ylabel('Sản phẩm',weight="bold", color="#D71313", fontsize=14, labelpad=20)
# Xoay nhãn trục x
plt.xticks(rotation = 0, ha = 'center')
plt.tight_layout()
# Hiển thị biểu đồ
plt.show()
```



Biểu đồ thể hiện số lượng sản phẩm khách hàng đã mua

- **Sử dụng biểu đồ cột dọc cho dữ liệu 'Category'**

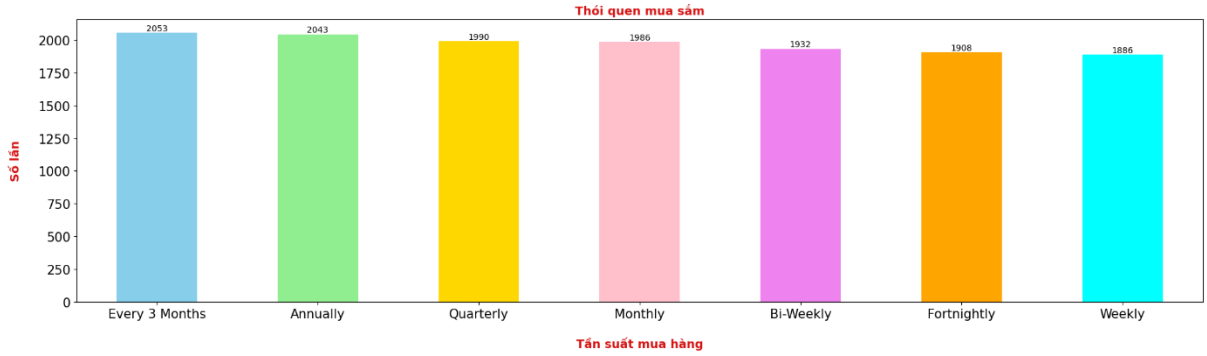
```
# Tạo biểu đồ cột dọc
plt.figure(figsize=(10, 6))
category_counts = dt['Category'].value_counts()
# Vẽ biểu đồ cột
ax = category_counts.plot(kind='bar', color=sns.color_palette("hls", len(category_counts)))
# Thêm tiêu đề và nhãn
plt.title('Tần suất mua hàng theo danh mục mặt hàng', weight="bold", color="#D71313", fontsize=14)
plt.xlabel('Hạng mục', weight="bold", color="#D71313", fontsize=14, labelpad=20)
plt.ylabel('Tần suất mua hàng', weight="bold", color="#D71313", fontsize=14, labelpad=20)
# Điều chỉnh các thuộc tính khác của trục x
plt.xticks(rotation=0, ha='center')
# Thêm số liệu lên đầu mỗi cột
for i, v in enumerate(category_counts):
    ax.text(i, v + 0.5, str(v), ha='center', va='bottom', color='black', fontweight='bold')

# Căn chỉnh và hiển thị biểu đồ
plt.tight_layout()
plt.show()
```



Biểu đồ thể hiện tần suất mua hàng theo danh mục mặt hàng

- **Sử dụng biểu đồ Histogram cho dữ liệu 'Frequency of Purchase'**

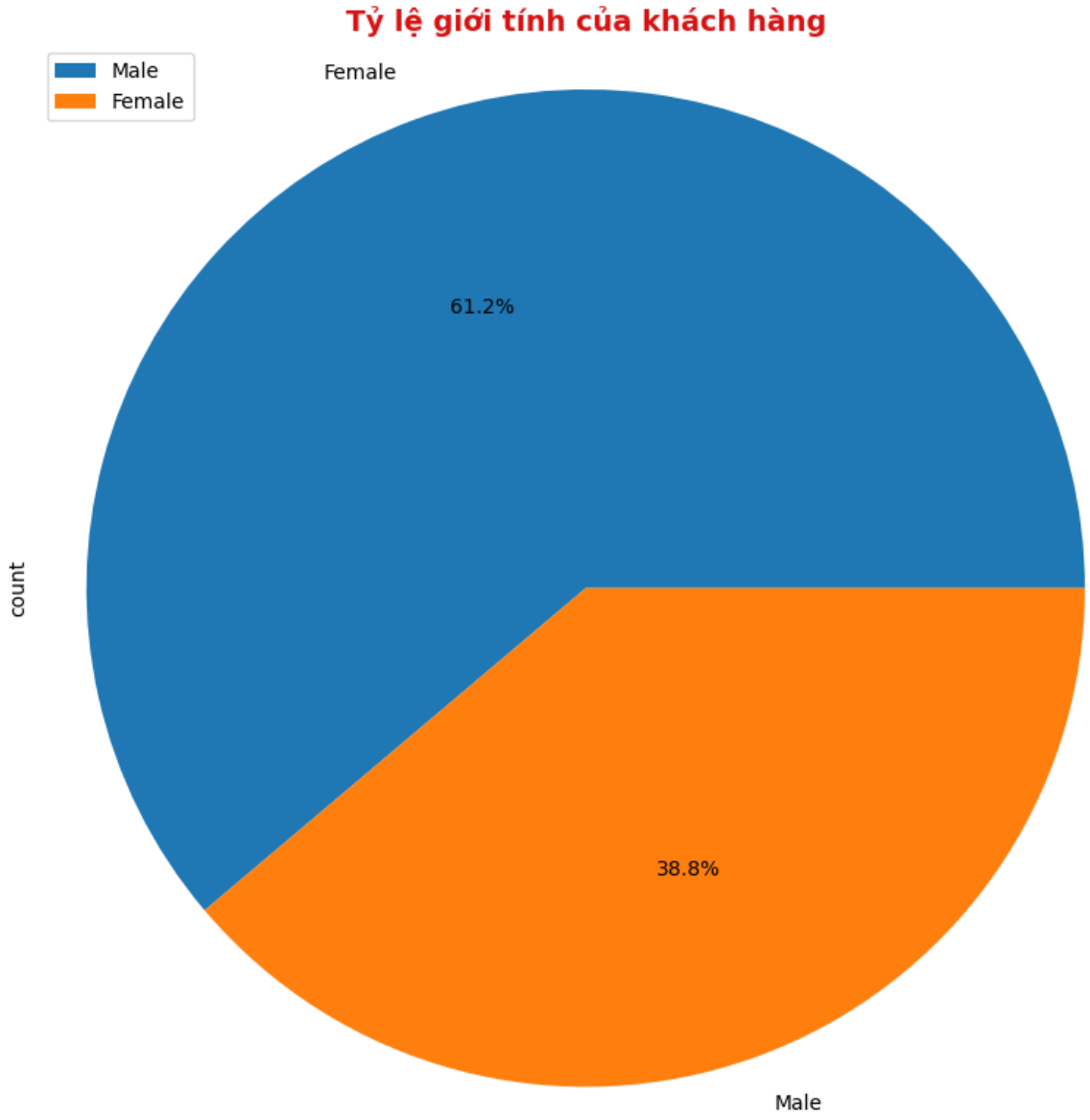
```
# Tạo biểu đồ Histogram
plt.figure(figsize=(20, 6))
colors = ['skyblue', 'lightgreen', 'gold', 'pink', 'violet', 'orange', 'cyan']
ax = dt['Frequency of Purchases'].value_counts().plot(kind='bar', color=colors, rot=0)
# Đặt nhãn cho các cột
ax.set_xticklabels(['Every 3 Months', 'Annually', 'Quarterly', 'Monthly', 'Bi-Weekly', 'Fortnightly', 'Weekly'])
# Thêm giá trị lên các cột
for p in ax.patches:
    ax.annotate(int(p.get_height()), (p.get_x() + 0.25, p.get_height() + 1), ha='center', va='bottom', color='black')
    ax.tick_params(axis='both', labelsize=15)
# Thêm tiêu đề và nhãn trục
plt.title('Thói quen mua sắm',weight="bold",color="#D71313", fontsize=14)
plt.xlabel('Tần suất mua hàng',weight="bold", color="#D71313", fontsize=14, labelpad=20)
plt.ylabel('Số lần',weight="bold", color="#D71313", fontsize=14, labelpad=20)
# Hiển thị biểu đồ
plt.tight_layout()
plt.show()
```



Biểu đồ thể hiện thói quen mua sắm của khách hàng

- **Sử dụng biểu đồ hình tròn cho dữ liệu 'Gender'**

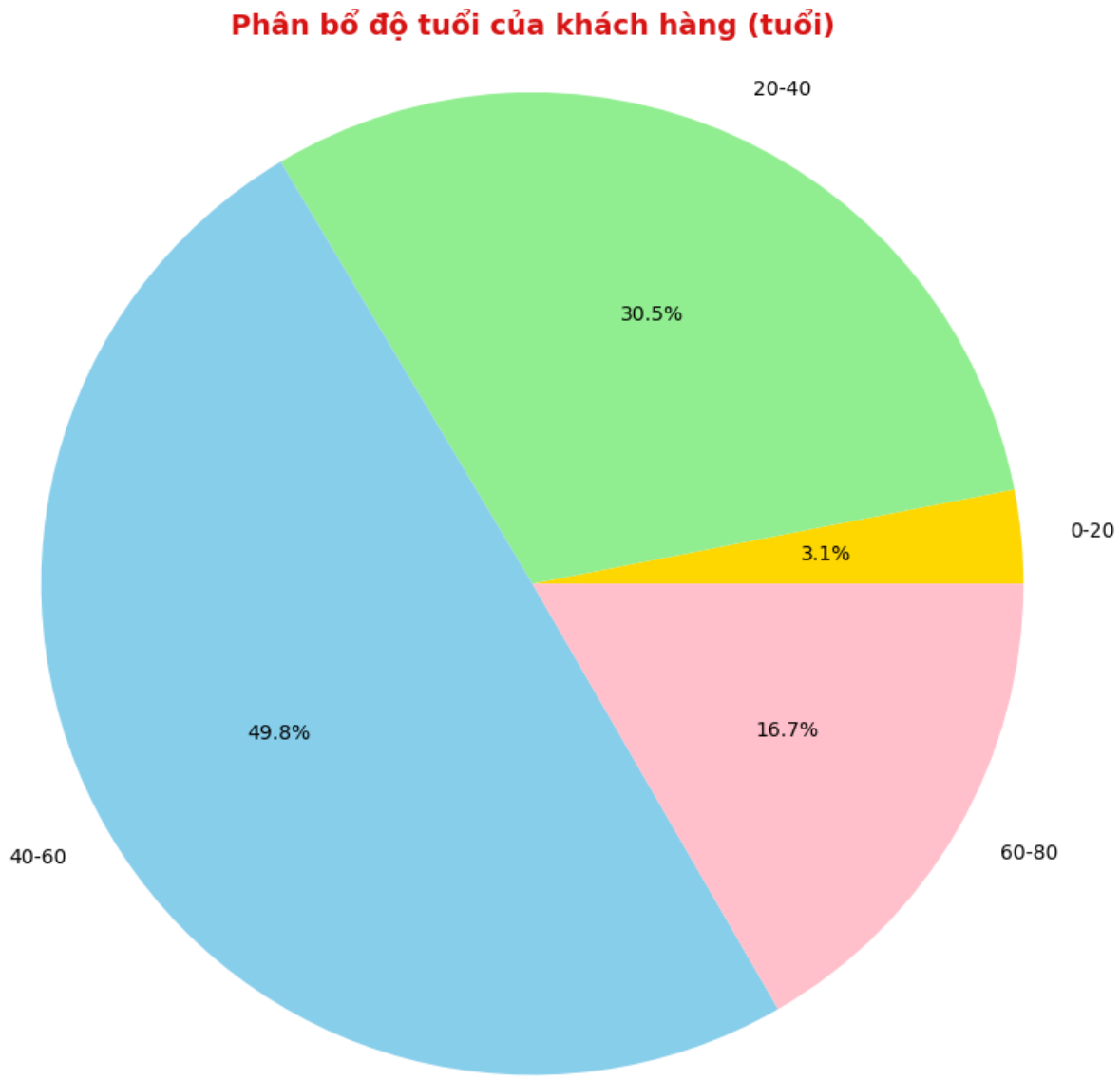
```
# Tạo biểu đồ tròn
plt.figure(figsize=(8, 8))
dt['Gender'].value_counts().plot(kind='pie', autopct='%1.1f%%')
plt.title('Tỷ lệ giới tính của khách hàng',weight="bold",color="#D71313", fontsize=14)
plt.axis('equal')
plt.legend(labels=dt['Gender'].unique(), loc='upper left')
plt.tight_layout()
plt.show()
```



Biểu đồ thể hiện tỷ lệ mua hàng giữa nam và nữ

- **Sử dụng biểu đồ hình tròn cho dữ liệu ‘Age’**

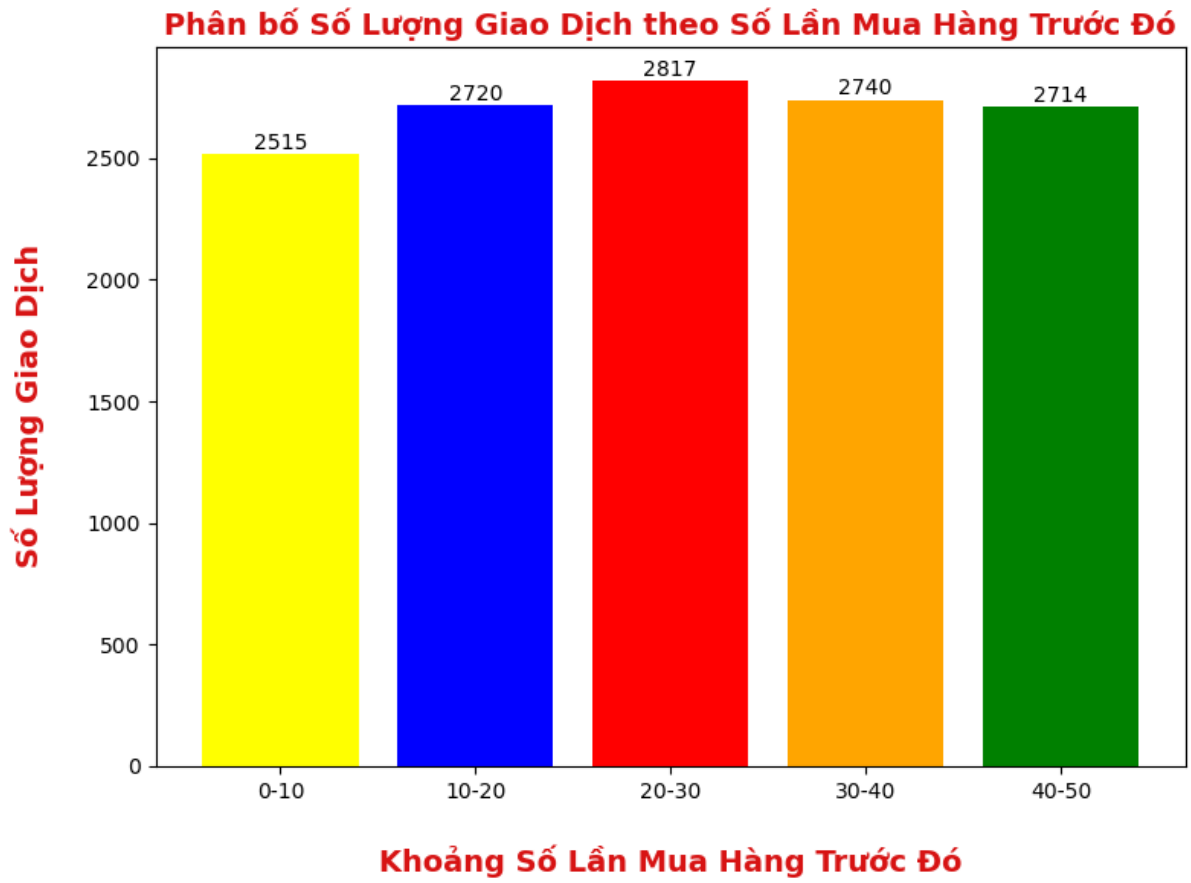
```
# Tạo biểu đồ hình tròn cho độ tuổi
plt.figure(figsize=(8, 8))
# Định nghĩa các khoảng tuổi và màu sắc tương ứng
age_ranges = [(0, 20, 'gold'), (20, 40, 'lightgreen'), (40, 60, 'skyblue'), (60, 80, 'pink')]
# Tính số lượng khách hàng trong mỗi khoảng tuổi
age_counts = []
for start, end, _ in age_ranges:
    count = dt[(dt['Age'] >= start) & (dt['Age'] < end)].shape[0]
    age_counts.append(count)
# Vẽ biểu đồ hình tròn
plt.pie(age_counts, labels=[f"{start}-{end}" for start, end, _ in age_ranges], colors=[color for _, _, color in age_ranges], autopct='%1.1f%%')
plt.title('Phân bố độ tuổi của khách hàng (tuổi)',weight="bold",color="#D71313", fontsize=14)
plt.axis('equal')
plt.tight_layout()
plt.show()
```



Biểu đồ thể hiện sự phân bổ của từng độ tuổi

- **Sử dụng biểu đồ cột dọc cho dữ liệu 'Previous Purchases'**

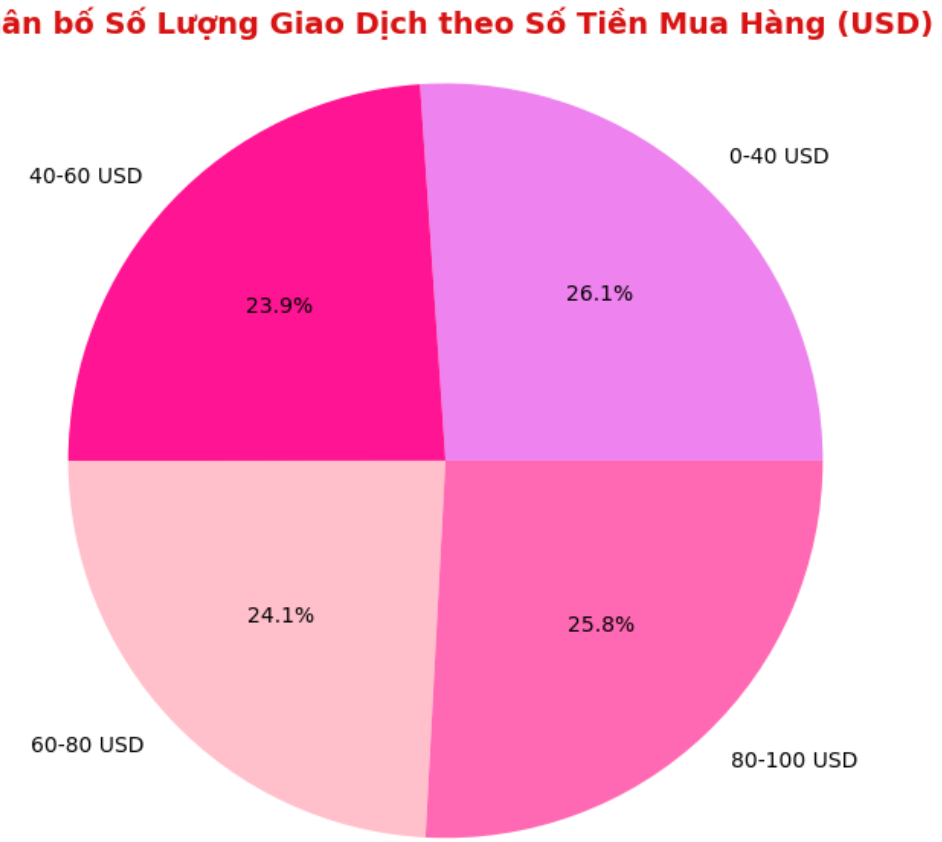
```
# Tạo biểu đồ cột dọc cho Previous Purchases
plt.figure(figsize=(8, 6))
# Định nghĩa các khoảng giá trị và màu sắc tương ứng
amount_ranges = [(0, 10, 'yellow'), (10, 20, 'blue'), (20, 30, 'red'), (30, 40, 'orange'), (40, 50, 'green')]
# Tính số lượng giao dịch trong mỗi khoảng giá trị
amount_counts = []
for start, end, _ in amount_ranges:
    count = dt[(dt['Previous Purchases'] >= start) & (dt['Previous Purchases'] < end)].shape[0]
    amount_counts.append(count)
# Vẽ biểu đồ cột dọc
bars = plt.bar(range(len(amount_counts)), amount_counts, color=[color for _, _, color in amount_ranges])
# Thiết lập tiêu đề và phụ đề
plt.title('Phân bố Số Lượng Giao Dịch theo Số Lần Mua Hàng Trước Đó', weight="bold", color="#D71313", fontsize=14)
plt.xlabel('Khoảng Số Lần Mua Hàng Trước Đó', weight="bold", color="#D71313", fontsize=14, labelpad=20)
plt.ylabel('Số Lượng Giao Dịch', weight="bold", color="#D71313", fontsize=14, labelpad=20)
# Đặt nhãn cho trục x
plt.xticks(range(len(amount_ranges)), [f"{start}-{end}" for start, end, _ in amount_ranges])
# Thêm số liệu lên đầu các cột
for i, bar in enumerate(bars):
    yval = bar.get_height()  # Lấy chiều cao của cột (tương ứng với số lượng giao dịch)
    plt.text(bar.get_x() + bar.get_width() / 2, yval + 0.5, str(yval), ha='center', va='bottom', color='black')
# Hiển thị biểu đồ
plt.tight_layout()
plt.show()
```



Biểu đồ phân bổ số lượng giao dịch theo số lần mua hàng trước đó

- **Sử dụng biểu đồ hình tròn cho dữ liệu 'Purchased Amount (USD)'**

```
# Tạo biểu đồ hình tròn cho Purchased Amount
plt.figure(figsize=(8, 6))

# Định nghĩa các khoảng giá trị và màu sắc tương ứng
amount_ranges = [ (0, 40, 'violet'), (40, 60, 'deeppink'), (60, 80, 'pink'), (80, 100, 'hotpink')]

# Tính số lượng giao dịch trong mỗi khoảng giá trị
amount_counts = []
for start, end, _ in amount_ranges:
    count = dt[(dt['Purchase Amount (USD)'] >= start) & (dt['Purchase Amount (USD)'] < end)].shape[0]
    amount_counts.append(count)

# Vẽ biểu đồ hình tròn
plt.pie(amount_counts, labels=[f"{start}-{end} USD" for start, end, _ in amount_ranges], colors=[color for _, _, color in amount_ranges],
        autopct='%1.1f%%')

# Thiết lập tiêu đề và phụ đề
plt.title('Phân bố Số Lượng Giao Dịch theo Số Tiền Mua Hàng (USD)',weight="bold",color="#D71313", fontsize=14)
plt.axis('equal')
plt.tight_layout()
plt.show()
```



Biểu đồ phân bổ số lượng giao dịch theo số tiền mua hàng

2.2.5. Tìm mô hình phù hợp với bộ dữ liệu

Tiến hành import các thư viện cần thiết

```
import pandas as pd
import numpy as np
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import LabelEncoder
```

Tách dữ liệu thành 2 tập X (Các biến độc lập) và Y (Biến phụ thuộc)

```
# Tạo tập dữ liệu Y
Y = df.iloc[:,-1].values
Y
```

```
array(['Blouse', 'Sweater', 'Jeans', ..., 'Belt', 'Shoes', 'Handbag'],
      dtype=object)
```

```
#Tạo tập dữ liệu X
X = df.iloc[:, :-1].values
X
```

```
array([[55.0, 'Male', 'Clothing', 53.0, 14.0, 3],
       [19.0, 'Male', 'Clothing', 64.0, 2.0, 3],
       [50.0, 'Male', 'Clothing', 73.0, 23.0, 6],
       ...,
       [46.0, 'Female', 'Accessories', 33.0, 24.0, 5],
       [44.0, 'Female', 'Footwear', 77.0, 24.0, 6],
       [52.0, 'Female', 'Accessories', 81.0, 33.0, 5]], dtype=object)
```

Mã hóa dữ liệu chữ về dạng số

Mã hóa Gender và Category của tập X

```
ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [1,2])], remainder = "passthrough")
X = ct.fit_transform(X)
X
```

```
array([[0.0, 1.0, 0.0, ..., 53.0, 14.0, 3],
       [0.0, 1.0, 0.0, ..., 64.0, 2.0, 3],
       [0.0, 1.0, 0.0, ..., 73.0, 23.0, 6],
       ...,
       [1.0, 0.0, 1.0, ..., 33.0, 24.0, 5],
       [1.0, 0.0, 0.0, ..., 77.0, 24.0, 6],
       [1.0, 0.0, 1.0, ..., 81.0, 33.0, 5]], dtype=object)
```

Mã hóa Item Purchased của tập Y

```
le = LabelEncoder()
Y = le.fit_transform(Y)
Y
```

```
array([ 2, 23, 11, ...,  1, 17,  7], dtype=int64)
```

2.2.6. Huấn luyện mô hình

Tách dữ liệu thành 2 tập train và test

Chọn tỷ lệ 70% train và 30% test

Tách dữ liệu thành Training Dataset và Testing Dataset

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.3)
```

Chuyển dữ liệu về dạng 0->1 để đồng bộ hóa dữ liệu trước khi đưa vào mô hình

FEATURE SCALING

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train[:,6:] = sc.fit_transform(X_train[:,6:])
```

```
X_train
```

```
array([[0.0, 1.0, 0.0, ..., -0.3707721546962222, -0.36740363157148065,
        -0.47795717536390464],
       [0.0, 1.0, 0.0, ..., 1.409120466508251, -1.5460310277996148,
        0.5196139934002081],
       [0.0, 1.0, 0.0, ..., 0.18014698996230522, 0.3952376248114296,
        0.020828409018151733],
       ...,
       [0.0, 1.0, 1.0, ..., 1.0700933005645419, 0.8805547879641907,
        -0.47795717536390464],
       [0.0, 1.0, 0.0, ..., -1.006448090840677, 0.256575578196355,
        1.5171851621643209],
       [1.0, 0.0, 0.0, ..., 0.22252538570526886, 1.643196044347101,
        -1.4755283441280174]], dtype=object)
```

```
X_test[:,6:] = sc.transform(X_test[:,6:])
```

```
X_test
```

```
array([[0.0, 1.0, 0.0, ..., -0.49790734192511316, 0.6032306947340415,
        -1.4755283441280174],
       [0.0, 1.0, 0.0, ..., 1.1548500920504692, 0.18724455488881772,
        -1.4755283441280174],
       [0.0, 1.0, 0.0, ..., -0.5402857376680769, 1.2965409278094144,
        1.0183995777822645],
       ...,
       [0.0, 1.0, 0.0, ..., -1.4302320482703135, 0.32590660150389233,
        1.0183995777822645],
       [1.0, 0.0, 0.0, ..., -0.031744988752513026, -0.5753967014940926,
        1.0183995777822645],
       [0.0, 1.0, 0.0, ..., -0.3707721546962222, 0.3952376248114296,
        -0.976742759745961]], dtype=object)
```

Tìm mô hình phù hợp với bộ dữ liệu

```
#import các thư viện cần thiết
from sklearn.metrics import accuracy_score, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from imblearn.over_sampling import SMOTE
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
# Danh sách các mô hình hồi quy
models = {
    "Linear Regression": LinearRegression(),
    "Random Forest": RandomForestRegressor(random_state=42),
    "Decision Tree": DecisionTreeRegressor(random_state=42),
    'Logistic Regression': LogisticRegression(),
    'KNN': KNeighborsRegressor(n_neighbors=5),
    'Gaussion NB': GaussianNB(),
}

# Khởi tạo danh sách để lưu kết quả
results = []

# Lặp qua từng mô hình
for name, model in models.items():
    # Huấn luyện mô hình
    model.fit(X_train, Y_train)

    # Dự đoán trên tập kiểm tra
    y_pred = model.predict(X_test)

    # Tính toán các chỉ số đánh giá
    mae = mean_absolute_error(Y_test, y_pred)
    mse = mean_squared_error(Y_test, y_pred)
    rmse = np.sqrt(mse)
    r2 = r2_score(Y_test, y_pred)

    # Lưu kết quả
    results.append({
        "Model": name,
        "MAE": mae,
        "MSE": mse,
        "RMSE": rmse,
        "R²": r2
    })
```

results

```
[{'Model': 'Linear Regression',
  'MAE': 5.400104234638694,
  'MSE': 41.5971052734356,
  'RMSE': 6.449581790584224,
  'R²': 0.16679319979475216},
 {'Model': 'Random Forest',
  'MAE': 2.2721492964748826,
  'MSE': 12.463978482311322,
  'RMSE': 3.53043601872507,
  'R²': 0.7503414826390407},
 {'Model': 'Decision Tree',
  'MAE': 1.7247568649885583,
  'MSE': 19.134504910373757,
  'RMSE': 4.374300505266386,
  'R²': 0.6167281471851478},
 {'Model': 'Logistic Regression',
  'MAE': 6.946224256292906,
  'MSE': 82.32379862700229,
  'RMSE': 9.073246311381737,
  'R²': -0.6489788985040235},
 {'Model': 'KNN',
  'MAE': 4.8473112128146445,
  'MSE': 38.680972540045765,
  'RMSE': 6.219402908643704,
  'R²': 0.22520451490405602},
 {'Model': 'Gaussion NB',
  'MAE': 6.824656750572083,
  'MSE': 79.63529748283753,
  'RMSE': 8.923861130857961,
  'R²': -0.5951271359606283}]
```

Danh sách kết quả của từng loại mô hình

Nhận thấy mô hình Random Forest:

Hệ số xác định R^2 = 75,03%: Điều này cho thấy mô hình Random Forest đã giải thích được 75% sự biến động của biến mục tiêu. Đây là một kết quả khá tốt, cho thấy mô hình có khả năng dự đoán khá tốt.

Hệ số MAE, MSE, RMSE: Các chỉ số này khá thấp, đặc biệt là khi so sánh với phạm vi giá trị của biến mục tiêu. Điều này cho thấy các dự đoán của mô hình khá gần với giá trị thực tế.

Dùng Random Forest để train mô hình

```
rf_model = RandomForestRegressor(random_state=1)
rf_model.fit(X_train, Y_train)
```

```
RandomForestRegressor(random_state=1)
```

```
rf_val_preds = rf_model.predict(X_test)
```

```
rf_val_preds[:5]
```

```
array([12.52      , 10.71      , 15.11333333, 16.19      ,  6.98      ])
```

Tạo một DataFrame so sánh giá trị Y_test và giá trị Y_pred (giá trị mô hình dự đoán)

```
pd.DataFrame({'y':Y_test, 'y_preds':rf_val_preds}).head(20)
```

	y	y_preds
0	13	12.520000
1	9	10.710000
2	13	15.113333
3	20	16.190000
4	7	6.980000
5	18	16.240000
6	20	19.540000
7	13	14.250000
8	18	17.114167
9	12	10.830000
10	8	9.323825
11	6	11.380000
12	23	21.470000
13	0	0.890000
14	14	16.920000
15	16	14.230000

CHƯƠNG 3: KẾT LUẬN

3.1. Ý nghĩa của mô hình

1. **Hỗ trợ ra quyết định kinh doanh:**
   - Mô hình cung cấp cái nhìn sâu sắc về hành vi mua sắm của khách hàng, dựa trên các yếu tố như độ tuổi, giới tính, tần suất mua hàng và giá trị đơn hàng.
   - Là cơ sở dữ liệu đáng tin cậy để doanh nghiệp xây dựng chiến lược kinh doanh phù hợp với từng nhóm khách hàng.
2. **Tối ưu hóa chiến lược marketing:**
   - Hỗ trợ phân khúc khách hàng theo đặc điểm và hành vi mua sắm.
   - Cho phép cá nhân hóa các chiến dịch marketing nhằm gia tăng khả năng tiếp cận và hiệu quả quảng bá.
   - Nâng cao hiệu quả các chương trình khuyến mãi nhờ việc nhắm mục tiêu chính xác hơn.
3. **Cải thiện trải nghiệm khách hàng:**
   - Dự đoán nhu cầu khách hàng để chuẩn bị nguồn hàng một cách chủ động và hiệu quả.
   - Đưa ra các đề xuất sản phẩm phù hợp với sở thích và thói quen mua sắm của từng khách hàng.
   - Tối ưu hóa quy trình bán hàng thông qua việc phân tích hành vi khách hàng.
4. **Quản lý kho hàng hiệu quả:**
   - Dự báo chính xác nhu cầu mua sắm, giúp tối ưu hóa dự trữ hàng hóa.
   - Giảm thiểu tình trạng thiếu hàng hoặc tồn kho vượt mức, tiết kiệm chi phí vận hành.
   - Tăng cường hiệu quả trong quản lý kho và nguồn lực.

3.2. Những hạn chế của mô hình

1. Hạn chế về dữ liệu:

- Dữ liệu mẫu từ Kaggle có thể không phản ánh toàn diện thực tế thị trường.
- Thiếu các yếu tố môi trường và bối cảnh thị trường như mùa vụ, xu hướng thời trang, và tình hình kinh tế.
- Chưa tích hợp dữ liệu về hành vi duyệt web và tương tác khách hàng trên các kênh số.

2. Hạn chế về kỹ thuật:

- Chỉ phân tích các yếu tố cơ bản, chưa bao quát các yếu tố phức tạp như tâm lý và cảm xúc khách hàng.
- Thiếu các phương pháp phân tích phản hồi và cảm xúc từ khách hàng.
- Khả năng dự đoán có thể bị ảnh hưởng bởi những thay đổi đột ngột từ thị trường.

3. Hạn chế về ứng dụng thực tế:

- Đòi hỏi thời gian để thu thập và cập nhật dữ liệu mới thường xuyên.
- Yêu cầu nguồn lực duy trì, vận hành, và cải tiến mô hình liên tục.

- Có thể gặp khó khăn trong việc tích hợp với các hệ thống quản lý hiện tại của doanh nghiệp.

4. Hạn chế về khả năng mở rộng:

- Mô hình cần điều chỉnh đáng kể nếu áp dụng cho các ngành hàng hoặc thị trường khác nhau.
- Chưa có khả năng học tự động và cập nhật từ dữ liệu mới.
- Hạn chế trong việc mở rộng để bao quát thêm các yếu tố mới ảnh hưởng đến hành vi mua sắm.