

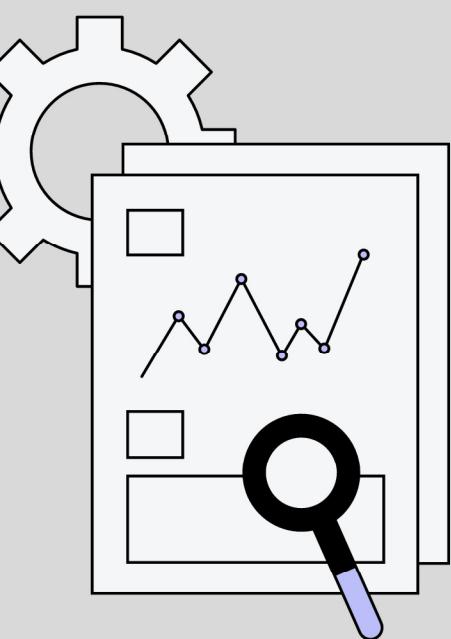
DUCT COMMENDATION EM ELOPMENT

hệ thống gợi ý sản phẩm

Anh Thư - Tú Uyên



Tổng quan nội



1

Làm sạch và khám phá dữ liệu

2

Phân tích dữ liệu & Xây dựng

- User-based Filtering
(Dựa trên khách hàng tương)
- Item-based Filtering
(Dựa trên sản phẩm tương t)
- Apriori – Market Basket Ana
(Gợi ý sản phẩm thường mua)

Đữ liệu phân tích

giao dịch **Online Retail** bao gồm tất cả
ch diễn ra từ ngày **1/4/2011 đến ngày**
của một công ty bán lẻ trực tuyến không
g, có trụ sở và đăng ký tại Vương quốc
ty này chủ yếu bán các món quà độc đáo
nhiều dịp khác nhau. Phần lớn khách
ông ty là các nhà bán sỉ.

gồm các cột sau: **InvoiceNo | StockCode**
on | Quantity | InvoiceDate | UnitPrice |
ID | Country



DATA CLEANING

làm sạch và khám phá dữ liệu



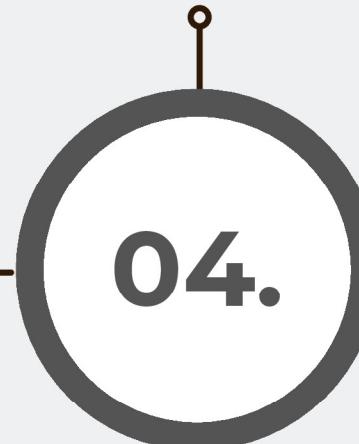
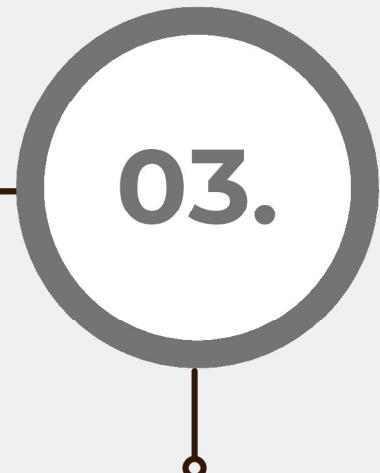
và đọc dữ

từ drive

Chuyển đổi kiểu dữ liệu (Data Types)

Loại bỏ giá trị

hợp lý



Kiểm tra thông tin tổng
quan và Lọc dữ liệu

Xử lý giá trị thiếu
(Missing Values)

Thiết kế và Load dataset từ Drive

In []:

```
import pandas as pd
import numpy as np
import scipy.stats
import matplotlib.pyplot as plt
import datetime
import seaborn as sns
from sklearn.metrics.pairwise import cosine_similarity
```

In []:

```
#Cài đặt thư viện
!pip install mlxtend
!pip install openpyxl
!pip install networkx
from mlxtend.frequent_patterns import apriori, association_rules
import networkx as nx
```

In []:

```
from google.colab import drive
drive.mount('/content/drive')
```

Tổng quan và Lọc dữ liệu

```
e.DataFrame'>
es, 0 to 541908
lumns):
   Count    Dtype
-----  -----
non-null   object
non-null   object
non-null   object
non-null   int64
non-null   datetime64[ns]
non-null   float64
non-null   float64
non-null   object
), float64(2), int64(1), object(4)
```

```
df_1['Month'] = df_1['InvoiceMonth']
df_1.head(4)
```

```
df_2 = df_1[df_1['Month'].isna()]
df_2.head(4)
```

- Dataset có 541909 hàng
- Kiểm tra Data type:
- Đa số data Type của các cột
- Xem lại các giá trị của cột
- Xem lại Data Type của cột

Tổng quan và Lọc dữ liệu

]:

```
2.describe()
```

]:

	Quantity	InvoiceDate	UnitPrice	CustomerID	More
t	178622.000000	178622	178622.000000	135106.000000	178622.000000
n	10.124324	2011-06-17 22:25:58.634882560	4.718681	15272.967825	6.074
n	-9600.000000	2011-04-01 08:22:00	-11062.060000	12347.000000	4.000
6	1.000000	2011-05-12 09:59:00	1.250000	13862.000000	5.000
6	4.000000	2011-06-19 11:10:00	2.080000	15144.000000	6.000
6	12.000000	2011-07-24 11:06:00	4.130000	16791.000000	7.000
x	4300.000000	2011-08-31 17:45:00	38970.000000	18287.000000	8.000
d	60.611790	NaN	119.918467	1718.395579	1.372

'' liệu (Data Types)

```
me.DataFrame'>
ies, 0 to 541908
olumns):
ull Count Dtype
-----  
9 non-null object
9 non-null object
5 non-null object
9 non-null int64
9 non-null datetime64[ns]
9 non-null float64
9 non-null float64
9 non-null object
1), float64(2), int64(1), object(4)
```

▶ print(df_2['CustomerID'])

→ float64

▶ print(df_2['InvoiceNo'].dtyp

object

- Có thể xuất hiện giá trị null trong cột CustomerID nên Dtypes là float64
- Các đơn hàng bị hủy thường có mã số "C" ở phía trước

Missing Values)

#Tạo function để tính tỉ lệ giá trị bị null trong các hàng

```
def calc_null_rate(df):
    newdf=df.isnull().sum().to_frame('null_count')
    newdf[['null_rate']] = newdf[['null_count']] / len(df)
    return newdf.sort_values(by=['null_rate'], ascending=False)
```

	null_count	null_rate
CustomerID	43516	0.243621
Description	692	0.003874
InvoiceNo	0	0.000000
Quantity	0	0.000000
StockCode	0	0.000000
InvoiceDate	0	0.000000
UnitPrice	0	0.000000
Country	0	0.000000
Month	0	0.000000

- Có 0,3874 % dữ liệu bị null trong (số lượng nhỏ)
- Có 24,36 % giá trị ở cột CustomerID là null
- Gán giá trị "unknow product" cho các giá trị dưới (mỗi sản phẩm tương ứng với một InvoiceNo riêng biệt)
- Kiểm tra cột CustomerID

Missing Values)

```
uct" cho những dòng thiếu Description  
= df_2['Description'].fillna('Unknown Product')
```

```
CustomerID null  
'].isnull()].head(5)
```

```
ID lại đúng Data Type  
df_2['CustomerID'].astype(int)
```

Khi có customerID null thì các giá trị ở các cột còn lại vẫn có
null sẽ không phù hợp ở các phân tích dưới
giá trị ở cột CustomerID bị null

Đổi customerID thành dạng Integer

ng hợp lê

trị Quantity < 0

0]. head(5)

các đơn hàng bị hủy

ợp lệ cuối cùng

ity'] > 0) & (df_2['UnitPrice'] > 0)]

Index	Description	Quantity	InvoiceDate	UnitPrice
0	RED RETROSPOT MINI CASES	-1	2011-04-01 10:22:00	7.95
2	CUPBOARD 3 DRAWER MA CAMPAGNE	-8	2011-04-01 10:45:00	12.75
2	DECORATIVE HANGING SHELVING UNIT	-1	2011-04-01 10:46:00	59.95
7	WHITE WOOD GARDEN PLANT LADDER	-1	2011-04-01 10:46:00	9.95
9	BOX OF 6 ASSORTED COLOUR TEASPOONS	-1	2011-04-01 10:46:00	4.25

```
#Tạo thêm cột Total Price
```

```
df['TotalPrice'] = df['Quantity']*df['UnitPrice']
df.head(2)
```

```
df = df.drop_duplicates() #Loại các dòng giống nhau ở tất cả các cột
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 130802 entries, 142083 to 320692
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
 ---  -- 
 0   InvoiceNo        130802 non-null   object 
 1   StockCode         130802 non-null   object 
 2   Description       130802 non-null   object 
 3   Quantity          130802 non-null   int64  
 4   InvoiceDate       130802 non-null   datetime64[ns]
 5   UnitPrice         130802 non-null   float64
 6   CustomerID        130802 non-null   int64  
 7   Country            130802 non-null   object 
 8   Month              130802 non-null   int32  
 9   TotalPrice         130802 non-null   float64
```

ÁM PHÁ LIỆU

nh phẩm bán chạy

ất và giá trị mua hàng trung bình



TOP SẢN PHẨM BÁN CH

TẦN SUẤT VÀ GIÁ TRỊ M
HÀNG TRUNG BÌNH

entity → Top sản phẩm bán chạy theo số lượng (xu hướng yêu thích)

```
đổng Quantity bán ra cho mỗi sản phẩm  
groupby('StockCode').agg(  
'sum') # Tính tổng Quantity cho mỗi StockCode
```

```
đến thấp và lấy top 10 sản phẩm  
by_quantity.sort_values('TotalQuantity', ascending=False).head(10)
```

Thêm thông tin về Description

```
merge(top_10_products, df[['StockCode', 'Description']].  
      drop_duplicates(), on='StockCode', how='left')
```

Sản phẩm bán chạy nhất theo Quantity

```
[[ 'StockCode', 'Description', 'TotalQuantity']]
```

	StockCode	Description
0	84077	WORLD WAR 2 GLIDERS ASSORTED
1	22197	SMALL POPCORN POPCORN
2	22197	
3	85099B	JUMBO BAG RED
4	84879	ASSORTED COLOUR BIRD
5	85123A	WHITE HANGING HEART T-LIGHT
6	15036	ASSORTED COLOURS
7	21212	PACK OF 72 RETROSPOT C
8	21977	PACK OF 60 PINK PAISLEY C
9	23203	JUMBO BAG DOILEY
10	23203	JUMBO BAG VINTAGE
11	23203	JUMBO BAG VINTAGE
12	47566	JUMBO BAG VINTAGE

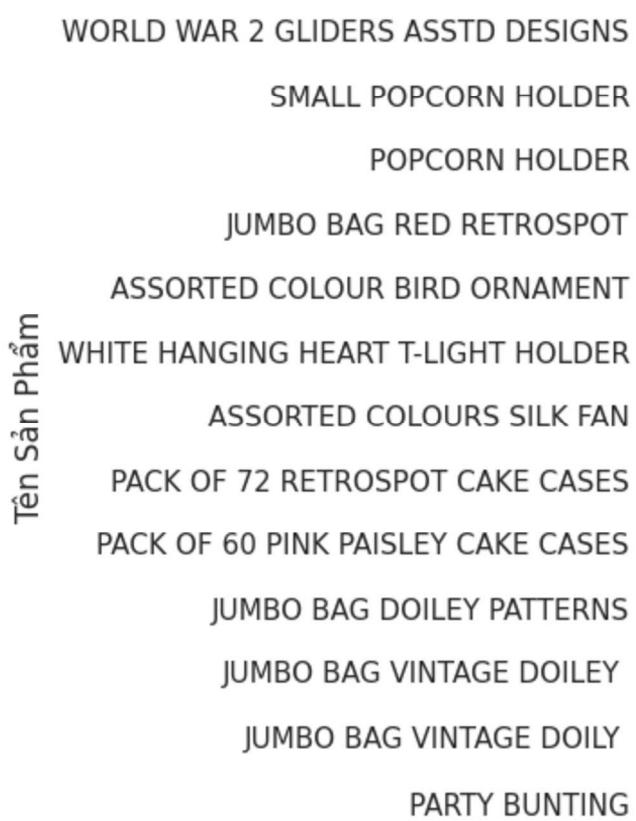
entity → Top sản phẩm bán chạy theo số lượng (xu hướng yêu thích)

a

```
s_info, x='TotalQuantity', y='Description', palette='Blues_d')
```

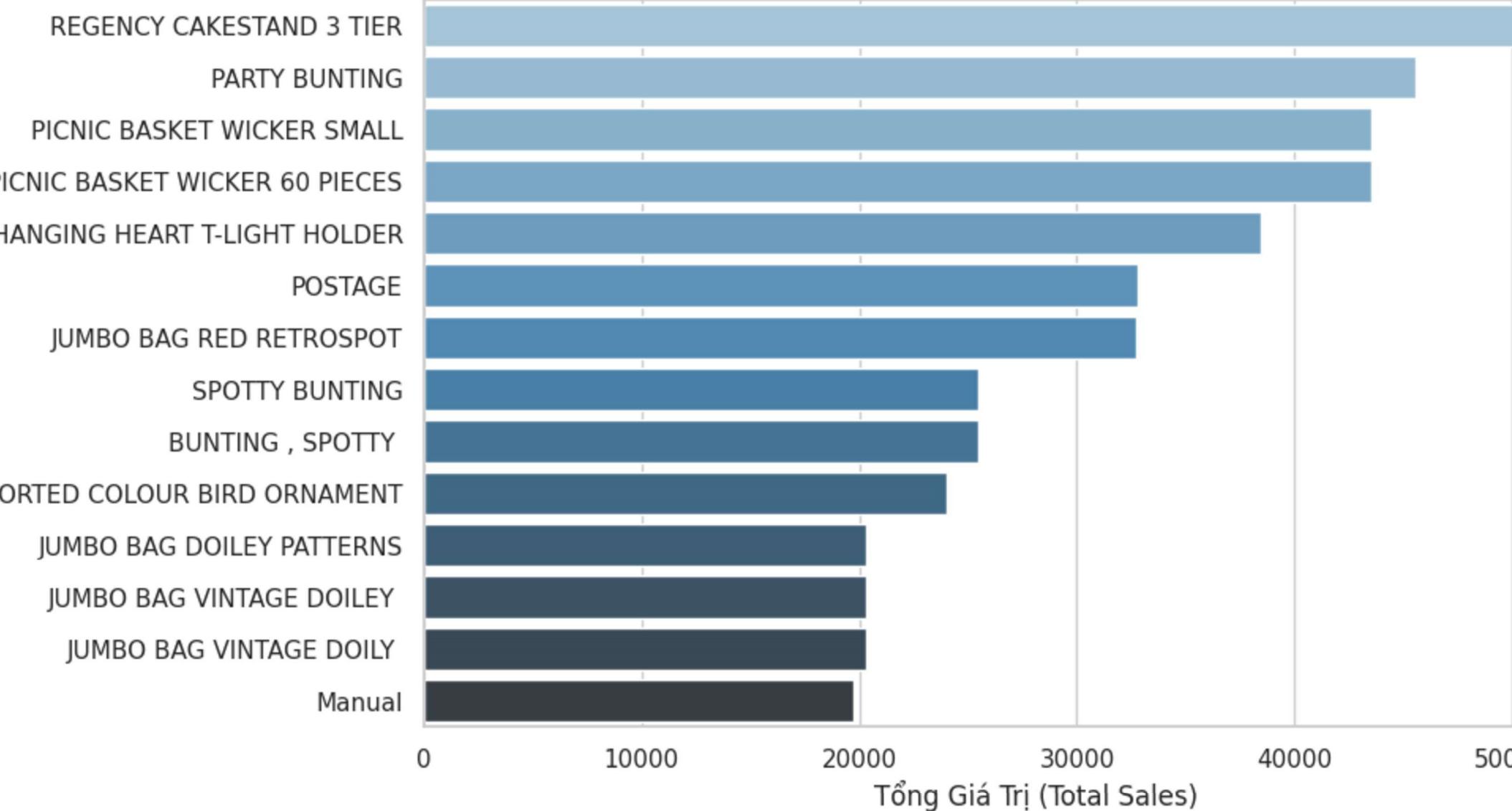
hị

```
Chạy Nhất Theo Số Lượng', fontsize=16)  
size=12)  
size=12)
```



IPrice → Top sản phẩm theo doanh thu (sản phẩm mang lại giá trị)

Top 10 Sản Phẩm Bán Chạy Nhất Theo Tổng Giá Trị



yếu tố Quantity và TotalPrices

```
ockCode và tính tổng Quantity và TotalPrice cho mỗi sản phẩm  
groupby('StockCode').agg(  
    quantity=('Quantity', 'sum'), # Tổng số lượng bán  
    total_price=('TotalPrice', 'sum')) # Tổng doanh thu  
)  
values('TotalSales', ascending=False).head(5)
```

Để lấy Description từ df

```
pd.merge(df_stock, df[['StockCode', 'Description']]).drop_duplicates(), on='StockCode', how='left')[['StockCode', 'Description', 'TotalSales', 'TotalQuantity']].sort_values('TotalSales', ascending=False)
```

Description	TotalSales	TotalQuantity
REGENCY CAKESTAND 3 TIER	53537.15	4651
PARTY BUNTING	45570.05	10127
PICNIC BASKET WICKER SMALL	43556.95	792
PICNIC BASKET WICKER 60 PIECES	43556.95	792
WHITE HANGING HEART T-LIGHT HOLDER	38477.19	14203

g tháng

```
InvoiceDate'].dt.to_period('M')  
groupby(['YearMonth', 'Description'])['TotalPrice'].sum().reset_index()
```

```
rank'] = monthly_revenue.groupby('YearMonth')['TotalPrice']\n    .rank(method='first', ascending=False)
```

```
monthly_revenue[monthly_revenue['Rank'] <= 3]
```

(20)

YearMonth		
1235	2011-04	
1606	2011-04	REGENCY CAKES
2115	2011-04	WHITE HANGING HEART T-LIC
3604	2011-05	PAR
3750	2011-05	
3854	2011-05	REGENCY CAKES
5911	2011-06	PAR
5946	2011-06	PICNIC BASKET WICKER

g tháng

))

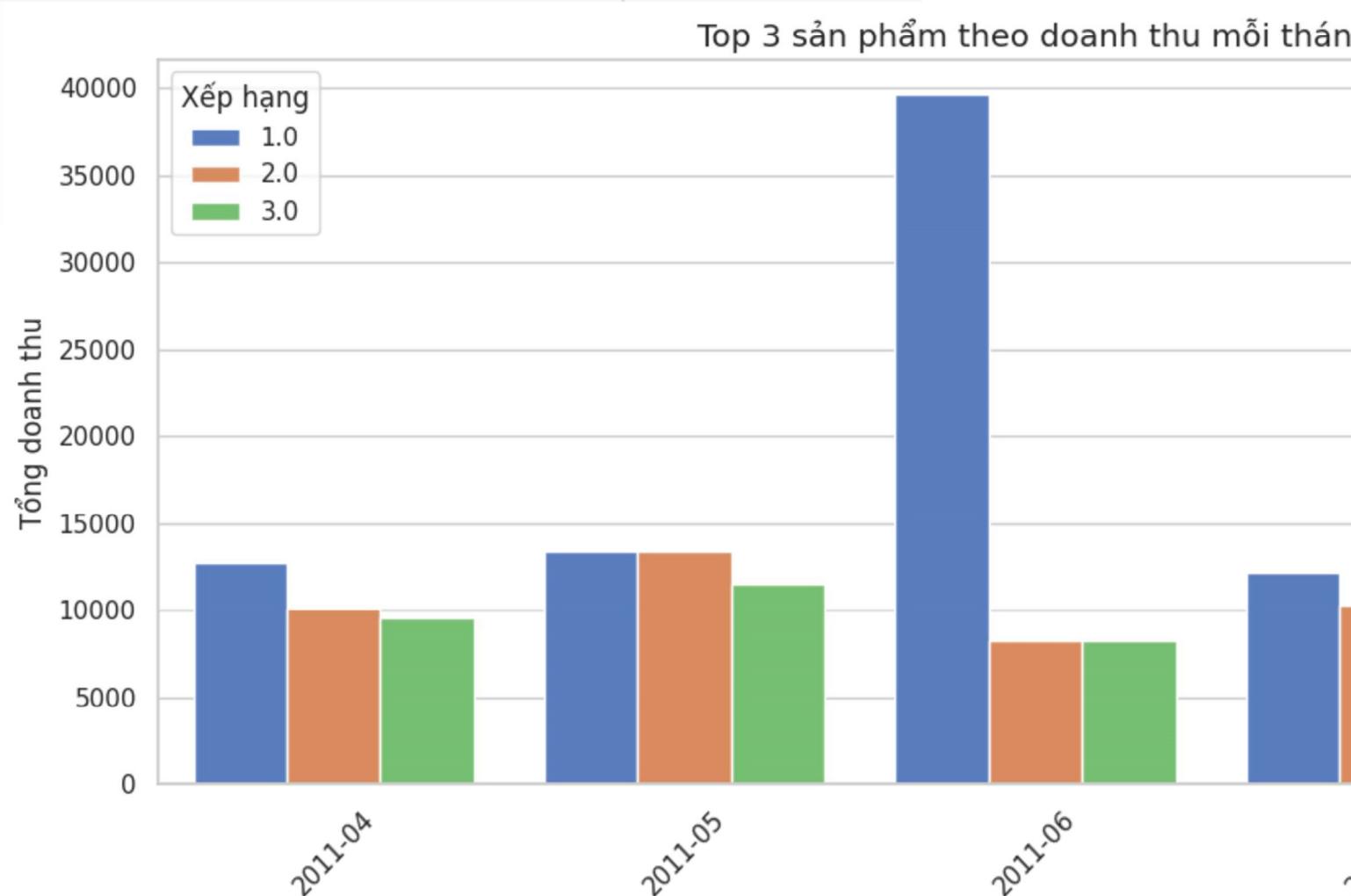
```
_month, x='YearMonth', y='TotalPrice', hue='Rank', palette='muted')
```

theo doanh thu mỗi tháng", fontsize=14)

u", fontsize=12)

ize=12)

g', loc='upper left')



Correlation giữa Quantity và Total Price

```
Quantity'].corr(df_stock['TotalSales'], method='pearson')  
n giữa Tổng Số Lượng Bán và Tổng Doanh Thu là: {correlation:.4f}")
```

Tổng Số Lượng Bán và Tổng Doanh Thu là: 0.6335

ntity và TotalSales

```
c, x='TotalQuantity', y='TotalSales', alpha=0.6)
```

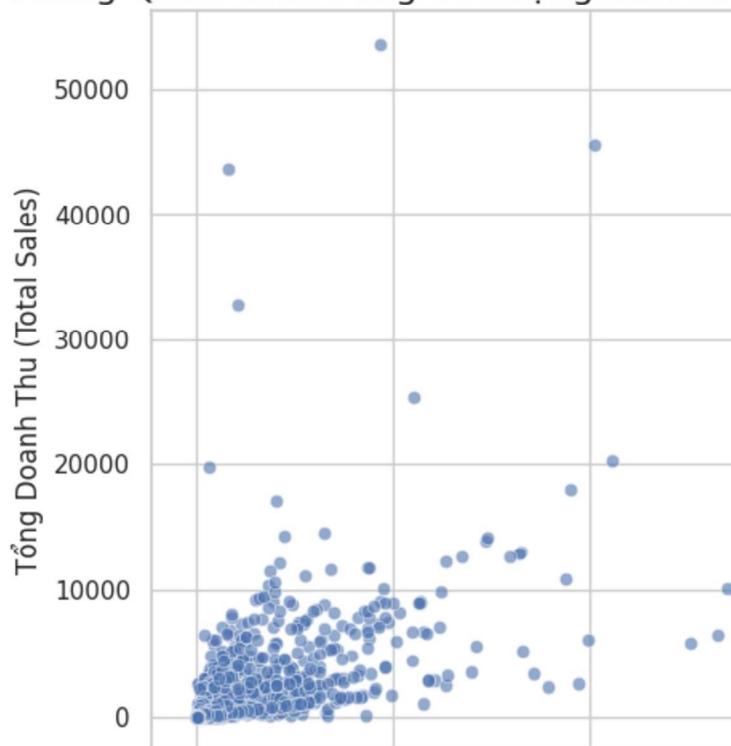
b thị

ra Tổng Số Lượng Bán và Tổng Doanh Thu (Theo StockCode)', fontsize=16)

n (Total Quantity)', fontsize=12)

Total Sales)', fontsize=12)

Mối Tương Quan Giữa Tổng Số Lượng Bán và



TOP SẢN PHẨM BÁN CH

TẦN SUẤT VÀ GIÁ TRỊ M HÀNG TRUNG BÌNH

số InvoiceNo duy nhất (unique) → Chỉ tính số lần khách hàng thực hiện tại thời gian, miễn là đơn hàng có InvoiceNo khác nhau.

```
g InvoiceNo duy nhất) và giá trị mua hàng trung bình  
('CustomerID').agg(  
'nunique'), # Đếm số lượng InvoiceNo duy nhất  
'sum')      # Tính tổng giá trị TotalPrice  
  
by='Frequency', ascending=False).head(5)
```

Frequency	TotalPrice	CustomerID	Frequency
71	7189.97	195	12748
68	49590.23	1126	14911
47	15322.70	2356	17841
39	18980.22	329	13089
35	4116.56	990	14606
Frequency	TotalPrice	CustomerID	Frequency
47	15322.70	1476	15749
39	18980.22	1202	15098
35	4116.56	33	12415
35	4116.56	29	12409
35	4116.56	112	12590

số InvoiceNo duy nhất (unique) → Chỉ tính số lần khách hàng thực hiện mua sắm trong thời gian, miễn là đơn hàng có InvoiceNo khác nhau.

Frequency và Avg_Monetary

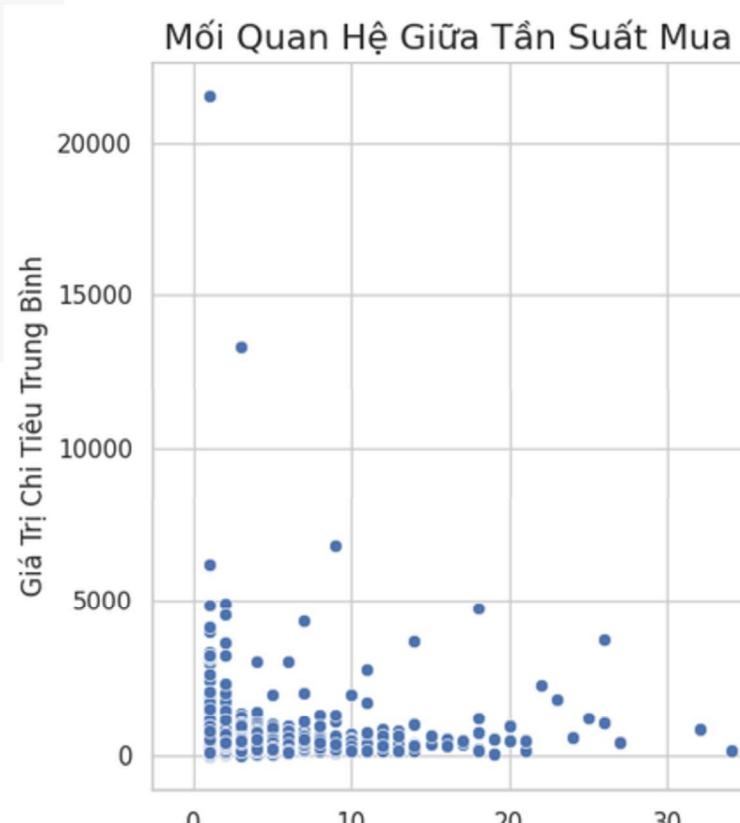
6))

```
_customer, x='Frequency', y='Avg_Monetary')
```

Giữa Tần Suất Mua Hàng và Giá Trị Chi Tiêu Trung Bình,

Mua Hàng', fontsize=12)

. Tiêu Trung Bình', fontsize=12)



Đó là ngày mà khách hàng có giao dịch Chỉ quan tâm đến ngày khác này có mua bao nhiêu đơn thì cũng chỉ tính 1 lần

```
ra ngày (bỏ phần giờ phút giây nếu có)
```

```
[InvoiceDate'].dt.date
```

```
CustomerID và đếm số ngày duy nhất
```

```
('CustomerID')['InvoiceDay'].nunique().reset_index()
```

CustomerID	Frequency_by_day
------------	------------------

1126	14911
------	-------

2356	17841
------	-------

195	12748
-----	-------

990	14606
-----	-------

1306	15311
------	-------

```
TotalPrice) theo CustomerID  
rID')['TotalPrice'].sum().reset_index()
```

```
nh giá trị mua hàng trung bình  
req, df_total, on='CustomerID')
```

```
g trung bình theo ngày  
y'] = df_summary['TotalPrice'] / df_summary['InvoiceDay']
```

```
àng hơn  
voiceDay': 'Frequency_by_day'}, inplace=True)
```

```
frequency_by_day', 'Avg_Monetary_by_day')[1].sort_values(by='Frequency_by_day', ascending=False).head()
```