Customer Segmentation & Prediction using Python

{Customer Behavior Analysis with Python}

Prepared by:Manar Naji

Icons to include:

Python • Jupyter Notebook • Pandas • Seaborn • Scikit-learn

.....

This project focuses on analyzing customer behavior in a .simulated e-commerce store using Python

The goal is to understand purchasing patterns through RFM analysis (Recency, Frequency, Monetary) and segment customers into actionable groups such as .Champions, At Risk, and Potential Loyalists

In addition, machine learning models (Linear Regression & Random Forest) were used to predict .customer spending based on their purchasing behavior

Some pictures

In [1]:

```python
import pandas as pd
import numpy as np
from datetime import datetime, timedelta

# لضبط العشوائية للتكرار
np.random.seed(42)

# إعدادات حجم البيانات
num_customers = 300
num_products = 50
num_orders = 1000

# بيانات العملاء
cities = ['الرياض', 'جدة', 'الدمام', 'المدينة', 'أبها']
genders = ['ذكر', 'أنثى']

customers = pd.DataFrame({
    'customer_id': range(1, num_customers + 1),
    'gender': np.random.choice(genders, size=num_customers),
    'city': np.random.choice(cities, size=num_customers),
    'join_date': [datetime(2022, 1, 1) + timedelta(days=int(np.random.rand()*730)) for _ in range(num_customers)]
})

# بيانات المنتجات
categories = ['ملابس', 'إلكترونيات', 'أدوات منزلية', 'مكياج', 'كتب']

products = pd.DataFrame({
    'product_id': range(1, num_products + 1),
    'category': np.random.choice(categories, size=num_products),
    'price': np.random.randint(50, 1000, size=num_products),
    'on_sale': np.random.choice([True, False], size=num_products),
    'rating': np.round(np.random.uniform(2.5, 5.0, size=num_products), 1)
})

# بيانات الطلبات
orders = pd.DataFrame({
    'order_id': range(1, num_orders + 1),
    'order_date': [datetime(2023, 1, 1) + timedelta(days=int(np.random.rand()*580)) for _ in range(num_orders)],
    'customer_id': np.random.choice(customers['customer_id'], size=num_orders),
    'product_id': np.random.choice(products['product_id'], size=num_orders),
    'quantity': np.random.randint(1, 4, size=num_orders)
})

# إضافة سعر المنتج لكل طلب وحساب السعر الكلي
orders = orders.merge(products[['product_id', 'price']], on='product_id')
orders['total_price'] = orders['price'] * orders['quantity']

print("✅ تم توليد البيانات بنجاح")
```

```
✅ تم توليد البيانات بنجاح
```

```python
import matplotlib.pyplot as plt
import seaborn as sns

sns.set(style="whitegrid")
plt.rcParams['font.family'] = 'Arial'  # لضمان عرض الخطوط العربية بشكل جيد

# عرض أول 5 صفوف من الجداول
print("Customers Table:")
display(customers.head())

print("Products Table:")
display(products.head())

print("Orders Table:")
display(orders.head())

# استخراج الشهر من تاريخ الطلب
orders['order_month'] = orders['order_date'].dt.to_period('M').astype(str)

# المبيعات الشهرية
monthly_sales = orders.groupby('order_month')['total_price'].sum().reset_index()

plt.figure(figsize=(12, 6))
sns.lineplot(data=monthly_sales, x='order_month', y='total_price', marker='o', color='blue')
plt.title('Monthly Sales')
plt.xlabel('Month')
plt.ylabel('Total Sales (SAR)')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()

# دمج الطلبات مع العملاء لأخذ المدينة
orders_with_city = orders.merge(customers[['customer_id', 'city']], on='customer_id')

city_sales = orders_with_city.groupby('city')['total_price'].sum().sort_values(ascending=False)

plt.figure(figsize=(8, 5))
sns.barplot(x=city_sales.values, y=city_sales.index, palette='Blues_d')
plt.title('Sales by City')
plt.xlabel('Total Sales')
plt.ylabel('City')
plt.show()

# دمج الطلبات مع المنتجات لأخذ الفئة
orders_with_category = orders.merge(products[['product_id', 'category']], on='product_id')

category_sales = orders_with_category.groupby('category')['total_price'].sum().sort_values(ascending=False)

plt.figure(figsize=(8, 5))
sns.barplot(x=category_sales.values, y=category_sales.index, palette='viridis')
plt.title('Sales by Product Category')
plt.xlabel('Total Sales')
plt.ylabel('Category')
```

Customers Table:

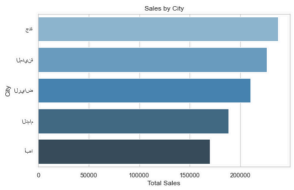
	customer_id	gender	city	join_date
0	1	ذكر	الرياض	2023-06-19
1	2	أنثى	الرياض	2022-04-04
2	3	ذكر	الدمام	2022-07-02
3	4	ذكر	جدة	2023-02-28
4	5	ذكر	أبها	2023-09-25

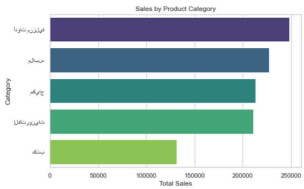
Products Table:

	product_id	category	price	on_sale	rating
0	1	مكياج	319	True	3.9
1	2	أدوات منزلية	877	True	4.3
2	3	مكياج	847	True	4.0
3	4	ملابس	340	False	4.6
4	5	مكياج	774	False	4.9

Orders Table:

	order_id	order_date	customer_id	product_id	quantity	price	total_price
0	1	2023-01-20	258	28	2	914	1828
1	6	2023-01-20	288	28	1	914	914
2	49	2023-11-16	283	28	2	914	1828
3	66	2023-08-24	281	28	2	914	1828
4	86	2024-05-14	234	28	3	914	2742






```python
import datetime as dt

# نفترض أن اليوم الأخير في البيانات هو تاريخ اليوم الذي نحتسب منه recency
reference_date = orders['order_date'].max() + pd.Timedelta(days=1)

# تجميع البيانات لإنشاء جدول RFM
rfm = orders.groupby('customer_id').agg({
    'order_date': lambda x: (reference_date - x.max()).days,  # Recency: الفرق بالأيام
    'order_id': 'count',                                      # Frequency: عدد الطلبات
    'total_price': 'sum'                                      # Monetary: مجموع الإنفاق
}).reset_index()

# إعادة تسمية الأعمدة
rfm.columns = ['customer_id', 'Recency', 'Frequency', 'Monetary']

# عرض أول 5 صفوف
display(rfm.head())
```

	customer_id	Recency	Frequency	Monetary
0	1	68	3	3066
1	2	447	3	2544
2	3	52	4	2164
3	4	156	2	632
4	5	87	3	689

In [4]:

```python
# تصنيف Recency: كلما كان أقل (أحدث) يحصل على درجة أعلى (5 أفضل)
rfm['R_Score'] = pd.qcut(rfm['Recency'], 5, labels=range(5, 0, -1))

# تصنيف Frequency: كلما كان أعلى يحصل على درجة أعلى (5 أفضل)
rfm['F_Score'] = pd.qcut(rfm['Frequency'].rank(method='first'), 5, labels=range(1, 6))

# تصنيف Monetary: كلما كان أعلى يحصل على درجة أعلى (5 أفضل)
rfm['M_Score'] = pd.qcut(rfm['Monetary'], 5, labels=range(1, 6))

# دمج الدرجات في عمود واحد للتسهيل
rfm['RFM_Score'] = rfm['R_Score'].astype(str) + rfm['F_Score'].astype(str) + rfm['M_Score'].astype(str)

# عرض أول 10 عملاء مع تصنيف RFM
display(rfm.head(10))
```

	customer_id	Recency	Frequency	Monetary	R_Score	F_Score	M_Score	RFM_Score
0	1	68	3	3066	4	2	3	423
1	2	447	3	2544	1	2	2	122
2	3	52	4	2164	4	3	2	432

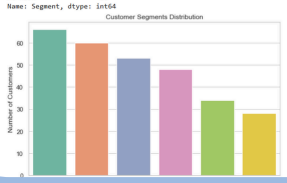
```python
def rfm_segment(row):
    if row['R_Score'] >= 4 and row['F_Score'] >= 4 and row['M_Score'] >= 4:
        return 'Champions'
    elif row['R_Score'] >= 3 and row['F_Score'] >= 3 and row['M_Score'] >= 3:
        return 'Loyal Customers'
    elif row['R_Score'] >= 4 and row['F_Score'] <= 2:
        return 'Potential Loyalists'
    elif row['R_Score'] <= 2 and row['F_Score'] >= 4:
        return 'At Risk'
    elif row['R_Score'] <= 2 and row['F_Score'] <= 2:
        return 'Lost'
    else:
        return 'Others'

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

# عرض عدد العملاء في كل فئة
segment_counts = rfm['Segment'].value_counts()
print(segment_counts)

# رسم بياني للتوزيع الفئات
plt.figure(figsize=(8,5))
sns.barplot(x=segment_counts.index, y=segment_counts.values, palette='Set2')
plt.title('Customer Segments Distribution')
plt.xlabel('Segment')
plt.ylabel('Number of Customers')
plt.show()
```

```
Lost                 66
Loyal Customers      60
Others               53
Champions            48
Potential Loyalists  34
At Risk              28
Name: Segment, dtype: int64
```



Step What Was Done

1. Data Generation

Created synthetic data for 300 customers, 50 products, and 1000 orders using Faker

2. EDA (Exploratory Data Analysis)

Analyzed monthly sales, top customers, cities, and product categories

3. RFM Analysis

Calculated Recency, Frequency, and Monetary metrics for each customer

4. Customer Segmentation

Grouped customers into segments: Champions, At Risk, Loyalists, etc.

5. Predictive Modeling

Built Linear Regression and Random Forest models to predict customer spending

6. Model Comparison

Compared both models using MSE and R^2 metrics

✅ Results:

- RFM analysis helped reveal valuable customer behavior insights.
- Customer segmentation provided clear targets for personalized marketing.
- Random Forest outperformed linear regression in accuracy and prediction power.

📝 Conclusion:

This project demonstrates the practical use of Python and data science techniques to analyze customer behavior, extract insights, and build predictive models.

It showcases key skills in data wrangling, analysis, segmentation, and machine learning, making it a strong addition to any data portfolio.

✅ Thank you for viewing!

🔗 Project link on Google Drive

🎯 Prepared by: Manar Naji