



Chapter 2 - Ex3: Predicting Customer Spend

Part1

- Cho dữ liệu retail_transactions.csv, gồm các thông tin như sau: Original
- Hãy chuẩn hóa dữ liệu này và lưu vào tập tin wrangled_transactions.csv, gồm các thông tin như sau: Pre-pro

▼ 1. Đọc dữ liệu. tìm hiểu thông tin chung từ dữ liệu.

```
import pandas as pd
```

```
df = pd.read_csv('retail_transactions.csv')
df.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	C
0	546729	22775	PURPLE DRAWERKNOB ACRYLIC EDWARDIAN	12	2011-03-16 11:36:00	1.25	18231.0	K
1	559898	21868	POTTING SHED TEA MUG	6	2011-07-13 12:18:00	1.25	16225.0	K
			HANGING .IAM		-----			

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 397884 entries, 0 to 397883
Data columns (total 8 columns):
InvoiceNo      397884 non-null int64
StockCode      397884 non-null object
Description     397884 non-null object
Quantity       397884 non-null int64
InvoiceDate    397884 non-null object
UnitPrice      397884 non-null float64
CustomerID     397884 non-null float64
Country        397884 non-null object
dtypes: float64(2), int64(2), object(4)
memory usage: 24.3+ MB
```

2. Chuyển dữ liệu cột 'InvoiceDate' thành định dạng datetime

```
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
```

3. Tạo cột 'revenue' = 'UnitPrice' * 'Quantity'

```
df['revenue'] = df['UnitPrice']*df['Quantity']
```

```
df.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	C
0	546729	22775	PURPLE DRAWERKNOB ACRYLIC EDWARDIAN	12	2011-03-16 11:36:00	1.25	18231.0	K
1	559898	21868	POTTING SHED TEA MUG	6	2011-07-13 12:18:00	1.25	16225.0	K
			HANGING .IAM		-----			

4. Quan sát thấy rằng mỗi hóa đơn được trải đều trên nhiều hàng, một hàng là một loại sản phẩm được mua => Kết hợp sao cho dữ liệu của mỗi giao dịch nằm trên một hàng bằng cách thực hiện thao tác nhóm trên InvoiceNo, với các chỉ định cách kết hợp các hàng được nhóm lại với nhau. revenue: sum, InvoiceDate: first, CustomerID: first.

```
operations = {'revenue':'sum',
              'InvoiceDate':'first',
              'CustomerID':'first'
            }
df = df.groupby('InvoiceNo').agg(operations)

df.head()
```

	revenue	InvoiceDate	CustomerID
InvoiceNo			
536365	139.12	2010-12-01 08:26:00	17850.0
536366	22.20	2010-12-01 08:28:00	17850.0
536367	278.73	2010-12-01 08:34:00	13047.0
536368	70.05	2010-12-01 08:34:00	13047.0

5. Vì sẽ sử dụng năm để quyết định hàng nào đang được sử dụng để dự đoán => hãy tạo một cột riêng có tên 'year' cho năm

```
df['year'] = df['InvoiceDate'].apply(lambda x: x.year)
```

```
df.head()
```

	revenue	InvoiceDate	CustomerID	year
InvoiceNo				
536365	139.12	2010-12-01 08:26:00	17850.0	2010
536366	22.20	2010-12-01 08:28:00	17850.0	2010
536367	278.73	2010-12-01 08:34:00	13047.0	2010
536368	70.05	2010-12-01 08:34:00	13047.0	2010
536369	17.85	2010-12-01 08:35:00	13047.0	2010

6. Ngày thực hiện các giao dịch cũng có thể là feature quan trọng. Những ngày kể từ khi giao dịch cuối cùng của khách hàng vào cuối năm, hoặc giao dịch đầu tiên đầu tiên vào đầu năm có thể cho chúng ta biết một phần về lịch sử mua hàng của khách hàng. Do đó, đối với mỗi giao dịch => tính toán có

bao nhiêu ngày chênh lệch giữa ngày cuối cùng của năm

```
df['days_since'] = (pd.datetime(year=2010, month=12, day=31) -
                    df['InvoiceDate']).apply(lambda x: x.days)
```

7. Hiện tại, dữ liệu đã được nhóm theo InvoiceNo, nhưng chúng ta cần dữ liệu được nhóm theo khách hàng bằng cách xác định một tập hợp các hàm tổng hợp cho từng biến và áp dụng chúng bằng cách sử dụng nhóm: revenue:sum, với days_since, tính số ngày tối đa và tối thiểu (là các feature cho biết khách hàng đã hoạt động bao lâu trong năm 2010 và gần đây), cũng như số lượng giá trị duy nhất (cho biết bao nhiêu ngày duy nhất khách hàng đã mua hàng). Chúng ta sẽ chỉ áp dụng các hàm này cho dữ liệu từ năm 2010 và lưu trữ trong một dataframe X

```
operations = {'revenue':'sum',
              'days_since':['max','min', 'nunique'],
              }
```

```
X = df[df['year'] == 2010].groupby('CustomerID').agg(operations)
```

```
X.head()
```

	revenue		days_since		
	sum	max	min	nunique	
CustomerID					
12347.0	711.79	23	23	1	
12348.0	892.80	14	14	1	
12370.0	1868.02	16	13	2	
12377.0	1001.52	10	10	1	
12383.0	600.72	8	8	1	

```
X.columns = [' '.join(col).strip() for col in X.columns.values]
```

```
X.head()
```

	revenue sum	days_since max	days_since min	days_since nunique
CustomerID				
12347.0	711.79	23	23	1
12348.0	892.80	14	14	1
12370.0	1868.02	16	13	2
12377.0	1001.52	10	10	1
12383.0	600.72	8	8	1

8. Tạo thêm một feature mới 'avg_order_cost' (chi tiêu trung bình cho mỗi đơn hàng) bằng cách chia 'revenue sum' (tổng doanh thu) cho 'days_since nunique' (là chi tiêu trung bình mỗi ngày, không phải cho mỗi đơn hàng, với giả định rằng nếu hai đơn hàng được đưa vào cùng một ngày thì có thể coi chúng là một phần của cùng order)

```
X['avg_order_cost'] = X['revenue sum']/X['days_since nunique']
```

```
X.head()
```

	revenue sum	days_since max	days_since min	days_since nunique	avg_order_cost
CustomerID					
12347.0	711.79	23	23	1	711.79
12348.0	892.80	14	14	1	892.80
12370.0	1868.02	16	13	2	934.01
12377.0	1001.52	10	10	1	1001.52
12383.0	600.72	8	8	1	600.72

9. Bây giờ đã có các feature inputs, chúng ta cần kết quả output, đó là tổng doanh thu cho năm 2011 (revenue_2011) bằng cách tính toán với một nhóm lưu trữ các giá trị trong biến y,

```
y = df[df['year'] == 2011].groupby('CustomerID')['revenue'].sum()
```

10. Ráp dữ liệu inputs X và output y thành một dataframe duy nhất.

```
wrangled_df = pd.concat([X,y], axis=1)
wrangled_df.columns = ['2010 revenue',
                        'days_since_first_purchase',
                        'days_since_last_purchase',
                        'number_of_purchases',
                        'avg_order_cost',
                        '2011 revenue']
```

```
wrangled_df.head()
```

	2010 revenue	days_since_first_purchase	days_since_last_purchase	number_of_pur
CustomerID				
12346.0	NaN	NaN	NaN	
12347.0	711.79	23.0	23.0	
12348.0	892.80	14.0	14.0	
12349.0	NaN	NaN	NaN	
12350.0	NaN	NaN	NaN	

11. Quan sát thấy nhiều giá trị trong dataframe là NaN. Nguyên nhân là do có một số khách hàng chỉ hoạt động trong năm 2010 hoặc chỉ hoạt động trong năm 2011, do đó

- không có dữ liệu cho năm khác. Do đó, cần xóa tất cả các khách hàng không hoạt động trong năm 2010 và không hoạt động trong năm 2011.

- Chú ý: Điều này có nghĩa là mô hình sẽ dự đoán chi tiêu của khách hàng trong năm tới với giả định rằng họ vẫn là khách hàng hoạt động.

```
wrangled_df = wrangled_df[~wrangled_df['2010 revenue'].isnull()]
wrangled_df = wrangled_df[~wrangled_df['2011 revenue'].isnull()]
```

12. Có thể chúng ta nên loại bỏ các ngoại lệ (outlier) dựa trên một định nghĩa: theo IQR, hay theo phân phối chuẩn. Theo "standard definition" thì ngoại lệ là bất kỳ điểm dữ liệu nào

- lớn hơn ba độ lệch chuẩn (std) trên trung vị (mean), vì vậy, thử áp dụng để loại bỏ các khách hàng vượt trội về doanh thu năm 2010 hoặc 2011.

```
wrangled_df = wrangled_df[wrangled_df['2011 revenue'] < ((wrangled_df['2011 revenue'].median(
wrangled_df = wrangled_df[wrangled_df['2010 revenue'] < ((wrangled_df['2010 revenue'].median(
```

```
wrangled_df.head()
```

	2010 revenue	days_since_first_purchase	days_since_last_purchase	number_of_pur
CustomerID				
12347.0	711.79	23.0	23.0	
12348.0	892.80	14.0	14.0	
12370.0	1868.02	16.0	13.0	
12377.0	1001.52	10.0	10.0	
12383.0	600.72	8.0	8.0	

13. Lưu dữ liệu đã xử lý vào tập tin

▼ 'wrangled_transactions.csv' để dùng cho Part2: Dự đoán.

```
wrangled_df.to_csv('wrangled_transactions.csv')
```

	2010 revenue	days_since_first_purchase	days_since_last_purchase	number_of_pur
CustomerID				
12347.0	711.79	23.0	23.0	
12348.0	892.80	14.0	14.0	
12370.0	1868.02	16.0	13.0	
12377.0	1001.52	10.0	10.0	
12383.0	600.72	8.0	8.0	