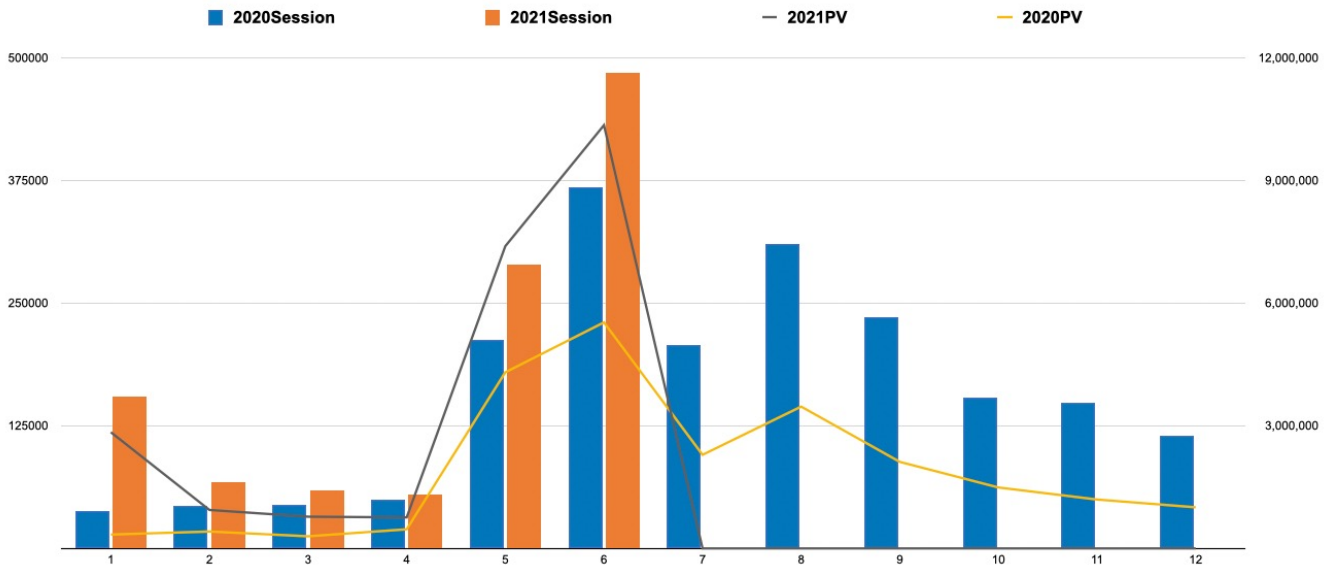
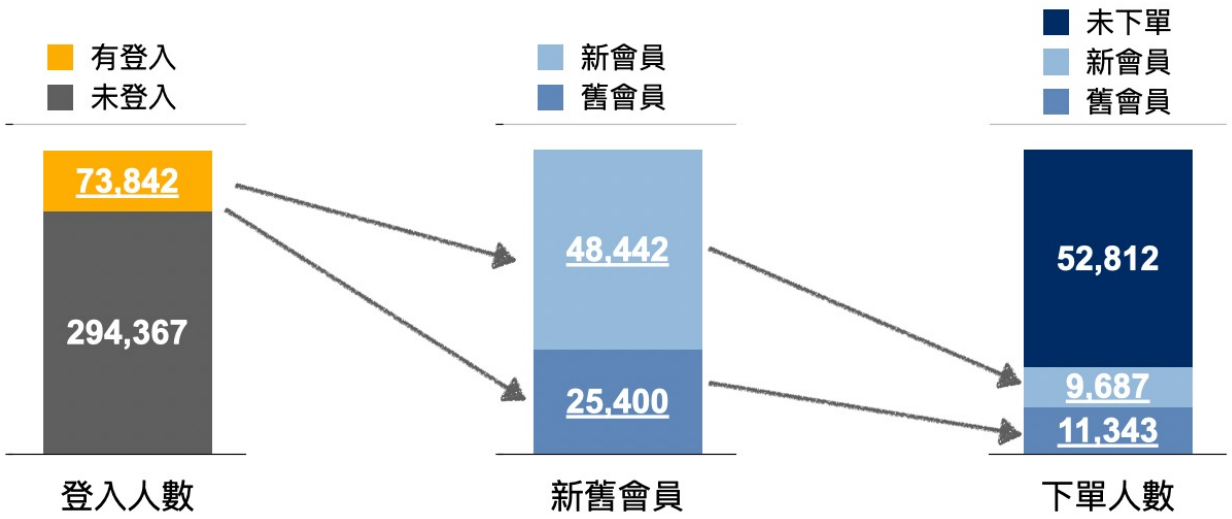
會員輪廓分析

流量與會員概況





70% 的流量來自於5&6兩個月

去年6月後流量逐月平均下降47%

-37% 對比去年5-6月 直接+搜尋流量衰退

Display與Email較去年比成長800%

20% 會員於今年有登入過網站

其中66%為今年新註冊會員

34%為今年前註冊的舊會員

5.7% 會員於今年有下過訂單

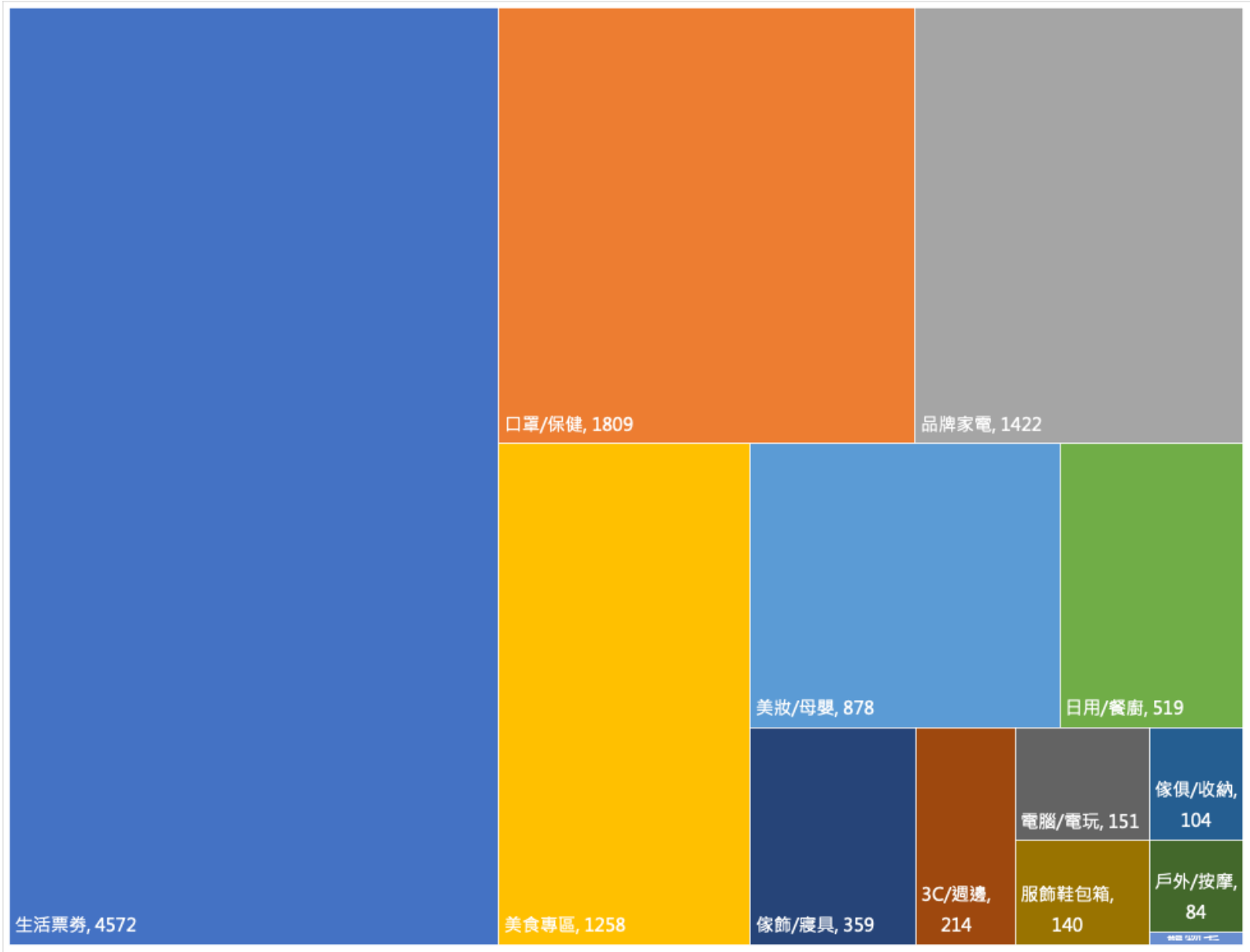
佔有登入的會員的28%

其中46%為新會員

54%為舊會員

新註冊會員銷售狀況

分類	銷量	營業額	商品數
生活票券	4,572	3,383,060	28
口罩/保健	1,809	1,507,004	207
品牌家電	1,422	3,066,429	306
美食專區	1,258	1,018,644	337
美妝/母嬰	878	1,620,862	195
日用/餐廚	519	399,950	110
傢飾/寢具	359	275,352	59
3C/週邊	214	1,476,696	76
電腦/電玩	151	655,029	72
服飾鞋包箱	140	68,912	23
傢俱/收納	104	128,549	30
戶外/按摩	84	147,340	29
寵物毛孩	12	6,070	6



商品偏好比較

有下單新會員的銷售數據

分類	銷量	營業額	商品數
生活票券	4,572	3,383,060	28
口罩/保健	1,809	1,507,004	207
品牌家電	1,422	3,066,429	306
美食專區	1,258	1,018,644	337
美妝/母嬰	878	1,620,862	195
日用/餐廚	519	399,950	110
傢飾/寢具	359	275,352	59
3C/週邊	214	1,476,696	76
電腦/電玩	151	655,029	72
服飾鞋包箱	140	68,912	23
傢俱/收納	104	128,549	30
戶外/按摩	84	147,340	29
寵物毛孩	12	6,070	6

未下單新會員的購物車清單

分類	下架	上架	總計
品牌家電	84	543	627
美食專區	101	207	308
夏普旗艦館	50	159	209
美妝/母嬰	33	149	182
口罩/保健	63	115	178
3C/週邊	24	128	152
電腦/電玩	54	83	137
日用/餐廚	30	87	117
傢俱/收納	16	71	87
傢飾/寢具	7	52	59
戶外/按摩	8	43	51
生活票券	3	46	49
服飾鞋包箱	4	35	39
無分類	10	21	31
寵物毛孩	2	9	11

1. 未消費新會員會員中**有6,000位將商品加入購物車**,另有65%新會員未消費且未加商品進購物車

2. 兩者需求結構相似
主要在**品牌家電**,**美食**,**美妝母嬰**分類

3. 在未下單會員中在**3C/周邊**,**電腦/電玩**
品牌家電關注度較高

4. **日用/餐廚**在兩者間保持平穩,可當作穩定回購的剛性需求品類

會員RFM指標分群

分群	上次訂購時間	平均下單次數	平均訂單金額	客戶類型	人數	說明
1	~5/24	1.2	18,472	重要喚回客戶	28	下單次數大於1 平均訂單金額大於平均
	5/24~	1.3	23,565	重要價值客戶	528	
2	~5/24	2.3	2,802	一般維持客戶	156	下單次數大於2 平均訂單金額小於平均
	5/24~	2.2	3,899	潛力客戶	970	
3	~5/24	5.3	23,082	重要喚回客戶	16	下單次數大於2 平均訂單金額大於平均
	5/24~	4.3	33,518	重要價值客戶	165	
4	~5/24	1.0	6,808	重要挽留客戶	44	下單次數為1 平均訂單金額大於平均
	5/24~	1.0	7,168	重要深耕客戶	615	
5	~5/24	1.0	1,396	流失客戶	1112	下單次數為1 平均訂單金額小於平均
	5/24~	1.0	2,050	新客戶	4273	

結論與行動

結論:

1. 未下單的新會員在**商品偏好**與有下單的新會員相似
2. 3C周邊及電腦/筆電這兩個分類需檢視購買與在購物車的品項有哪些,提高商品銷售率
3. 針對新會員有下單的RFM指標可以發現有**44%是願意回購2次以上**且訂單**金額在2~3,000**,有**26%會下單1次**且訂單**金額在6~7,000**,有**22%會下單超過1次**且訂單**金額在15,000~30,000**

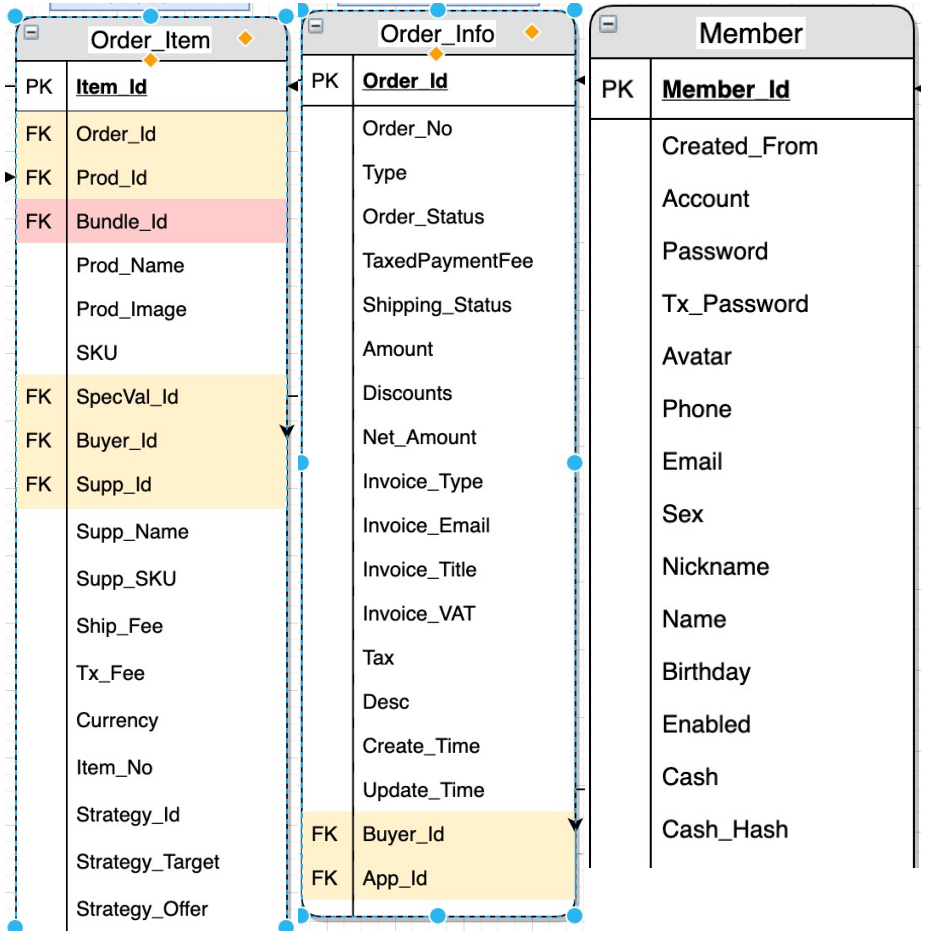
行動:

1. 針對有加入購物車但未購買的會員發送客製化的EDM(隱藏賣場不鎖身份),商品可參考美安或已下單的歷史數據,架構可參考會員輪廓與商品偏好
2. 利用網站瀏覽數據進行會員進行推播

Kmeans 會員分群

Data Extraction | Data Cleaning | Data Wrangling | Analysis | Action

Tables



Mysql Query

```sql
select member.Member_Id,sum(order_item.Net_Amount) as Order_Sales,
count(distinct(order_info.Order_Id)) as Order_count ,
date_format(max(order_info.Create_Time),'%Y%m%d') as last_order
from (order_item
join order_info on order_info.order_id = order_item.order_id)
join member on member.Member_Id = order_info.Buyer_Id
where date_format(date_add(order_item.create_time,interval 8 hour),'%Y%m%d') between 20210101 and 20210630
and date_format(date_add(member.create_time,interval 8 hour),'%Y%m%d') between 20210101 and 20210630
#第一層資料過濾(針對訂單狀態篩選已成立的訂單)
and order_item.Order_Status in (1,2)
#第二層資料過濾(商品卡廠商)
and order_item.Supp_Id not in (910,772)
#第三層資料過濾(去除團購會員)
and order_item.Buyer_Id not in (
select distinct(buyer_id) from order_item where Prod_Name like '%《團》%'
and order_status in (1,2))
group by member.Member_Id;
```

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

```
import datetime
```

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

```
df = pd.read_csv('member.csv')
```

```
df = df.rename(columns = {'Order_count':'Order_Count','last_order':'Last_Order'})
```

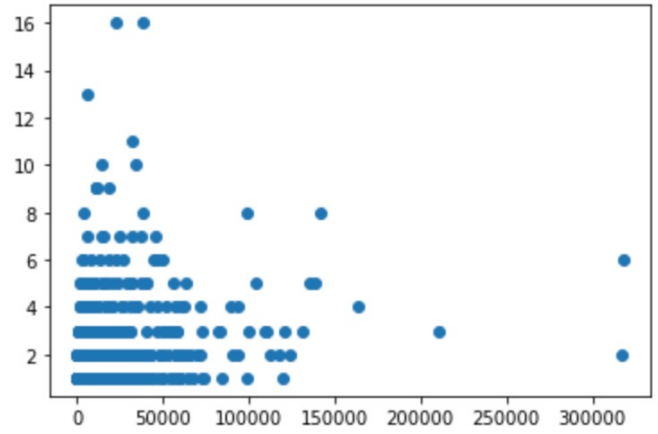
```
df['Last_Order'] = pd.to_datetime(df['Last_Order'].astype(str), format='%Y%m%d')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7960 entries, 1 to 7960
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype         
---  ------       --------------  -----         
 0   Member_Id    7960 non-null   int64         
 1   Order_Sales  7960 non-null   int64         
 2   Order_Count  7960 non-null   int64         
 3   Last_Order   7960 non-null   datetime64[ns]
dtypes: datetime64[ns](1), int64(3)
memory usage: 310.9 KB
```

```
plt.scatter(df['Order_Sales'], df['Order_Count'])
```

```
<matplotlib.collections.PathCollection at 0x7fc50cb82f90>
```




```
pivot_Order_Count = pd.pivot_table(df,index = 'Order_Count',values = 'Member_Id',aggfunc = 'count')
pivot_Order_Count
```

Order_Count	Member_Id
1	6448
2	1109
3	265
4	73
5	33
6	11
7	7
8	4
9	4
10	2

```python
def Last_Level(l):
    if l < pd.Timestamp('2021-05-25'):
        return 0
    else:
        return 1
```

```python
def Order_Count(o):
    if o >= 7:
        return 0
    elif o > 3 and o <= 6:
        return 1
    elif o == 3:
        return 2
    elif o == 2:
        return 3
    else:
        return 4

def Order_Sales(s):
    if s > 150000:
        return 0
    elif s >= 70000 and s < 150000:
        return 1
    elif s >= 50000 and s < 70000:
        return 2
    elif s >= 30000 and s < 50000:
        return 3
    elif s >= 10000 and s < 30000:
        return 4
    elif s >= 5000 and s < 10000:
        return 5
    else:
        return 6
```

```python
df['Last_Order_Level'] = df['Last_Order'].apply(Last_Level)
```

```python
df['Order_Count_Level'] = df['Order_Count'].apply(Order_Count)
```

```python
df['Order_Sales_Level'] = df['Order_Sales'].apply(Order_Sales)
```

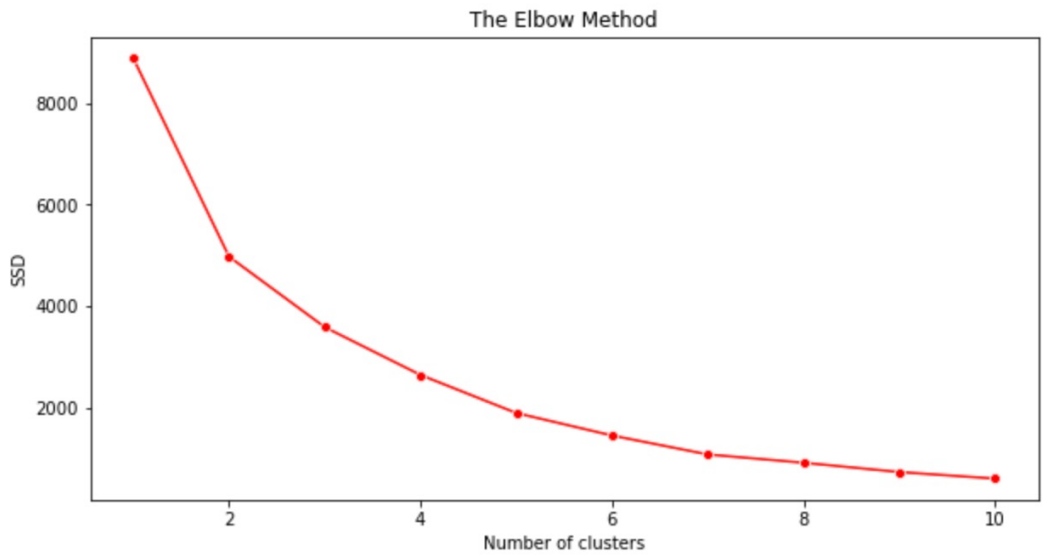
```python
df
```

	Member_Id	Order_Sales	Order_Count	Last_Order	Last_Order_Level	Order_Count_Level	Order_Sales_Level
1	942651	599	1	2021-01-01	0	4	6
2	942653	2945	1	2021-01-01	0	4	6
3	942655	378	1	2021-01-01	0	4	6
4	942657	447	1	2021-01-03	0	4	6
5	942659	189	1	2021-01-01	0	4	6
...	...	...	...	...	...	...	...
7956	1024210	1380	1	2021-06-30	1	4	6
7957	1024232	13780	1	2021-06-30	1	4	4
7958	1024240	2700	1	2021-06-30	1	4	6
7959	1024247	2700	1	2021-06-30	1	4	6
7960	1024249	1380	1	2021-06-30	1	4	6

```
from sklearn.cluster import KMeans
ssd = []
for k in range(1,11):
    model = KMeans(n_clusters = k)
    model.fit(df_kmeans)
    ssd.append(model.inertia_)
```

```
import os
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10,5))
sns.lineplot(range(1, 11), ssd,marker='o',color='red')
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('SSD')
plt.show()
```

```
/Users/wangxiang/opt/anaconda3/lib/python3.7/site-packages/seaborn/_decora
ng variables as keyword args: x, y. From version 0.12, the only valid posi
other arguments without an explicit keyword will result in an error or mis
  FutureWarning
```




```
pd.Series(ssd).diff()
```

```
0             NaN
1   -3911.150306
2   -1390.219521
3    -942.897303
4    -749.648402
5    -443.753783
6    -373.250576
7    -163.138126
8    -183.920880
9    -129.094694
dtype: float64
```

```
kmeans = KMeans(n_clusters = 6, init = 'k-means++', random_state = 42)
clusters = kmeans.fit_predict(df_kmeans)
```

```
df['Clusters'] = clusters
```

**這裡應該選擇分5群就好 因為5到6群的SSD差異為最小 這裡有誤判

分群	上次訂購時間	平均下單次數	平均訂單金額	客戶類型	人數	說明
1	~5/24	1.2	18,472	重要喚回客戶	28	下單次數大於1 平均訂單金額大於平均
	5/24~	1.3	23,565	重要價值客戶	528	
2	~5/24	2.3	2,802	一般維持客戶	156	下單次數大於2 平均訂單金額小於平均
	5/24~	2.2	3,899	潛力客戶	970	
3	~5/24	5.3	23,082	重要喚回客戶	16	下單次數大於2 平均訂單金額大於平均
	5/24~	4.3	33,518	重要價值客戶	165	
4	~5/24	1.0	6,808	重要挽留客戶	44	下單次數為1 平均訂單金額大於平均
	5/24~	1.0	7,168	重要深耕客戶	615	
5	~5/24	1.0	1,396	流失客戶	1112	下單次數為1 平均訂單金額小於平均
	5/24~	1.0	2,050	新客戶	4273	

**這兩群當時是分開的 因誤判為6群