In [4]:

```
%matplotlib inline
import pandas as pd
import matplotlib.pyplot as plt
import calendar
import seaborn as sns
from category_encoders import TargetEncoder
from imblearn.over_sampling import SMOTENC
from collections import Counter
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
# pip install git+https://github.com/scikit-learn-contrib/category_encoders.git#egg=httpie
# pip install imblearn

sns.set()
plt.rcParams['font.family']= ['Microsoft JhengHei']
```

```
Using TensorFlow backend.
```

手動刪除部分欄位

- 刪除POLICY_HOLDER, Policy_RK, INJURED_RK, Claim_RK, illness_desc, INSURED_RK, MATURITY_BENEFICIARY_RK, DEATH_BENEFICIARY_RK, 初次理賠時間, 結案後120天, 結案後180天, 結案後360天, CUST_RK, VIP_CLASS, VIP, CLIENT_MARITAL, CLIENT_INCOME, TOTAL_AUM
- 產生AGE
- 存為'理賠再購屬性合併_before_encoding.xlsx'

刪除沒有要保人資料的列

In [5]:

```
df = pd.read_excel('理賠再購屬性合併_before_encoding.xlsx')
df = df.dropna()
df.to_excel('理賠再購屬性合併_before_encoding.xlsx',index=False)
```

匯入資料

In [6]:

```
df = pd.read_excel('理賠再購屬性合併_before_encoding.xlsx')
```

In [7]:

```
df
```

Out[7]:

	BundleSubtype2	illness_code	DiagnosisCode_DESC	claim_settle_dt	REIMBURSED_YR_1
0	5.N疾病醫療	C18	02.腫瘤	2015-03-25	46987
1	5.N疾病醫療	C18	02.腫瘤	2015-08-15	7087
2	5.N疾病醫療	C18	02.腫瘤	2015-08-15	30712
3	5.N疾病醫療	C18	02.腫瘤	2015-08-15	7087
4	5.N疾病醫療	C18	02.腫瘤	2015-08-15	7087
...	...	...	...	...	
210784	6.N意外醫療	Y99.8	99.不知道不想猜	2017-10-15	1050
210785	6.N意外醫療	V23	20.疾病和死亡的外因	2017-02-19	813
210786	6.N意外醫療	Y99.8	99.不知道不想猜	2017-08-23	1627
210787	5.N疾病醫療	D36	02.腫瘤	2017-08-11	546
210788	4.C重大疾病	I25.1	09.循環系統疾病	2017-04-29	16825

210789 rows × 34 columns

In [8]:

```
X = df.iloc[:,:31]
y = df.iloc[:,31]
```

In [9]:

```
X
```

Out[9]:

	BundleSubtype2	illness_code	DiagnosisCode_DESC	claim_settle_dt	REIMBURSED_YR_1
0	5.N疾病醫療	C18	02.腫瘤	2015-03-25	46987
1	5.N疾病醫療	C18	02.腫瘤	2015-08-15	7087
2	5.N疾病醫療	C18	02.腫瘤	2015-08-15	30712
3	5.N疾病醫療	C18	02.腫瘤	2015-08-15	7087
4	5.N疾病醫療	C18	02.腫瘤	2015-08-15	7087
...	...	...	...	...	
210784	6.N意外醫療	Y99.8	99.不知道不想猜	2017-10-15	1050
210785	6.N意外醫療	V23	20.疾病和死亡的外因	2017-02-19	813
210786	6.N意外醫療	Y99.8	99.不知道不想猜	2017-08-23	1627
210787	5.N疾病醫療	D36	02.腫瘤	2017-08-11	546
210788	4.C重大疾病	I25.1	09.循環系統疾病	2017-04-29	16825

210789 rows × 31 columns

In [10]:

```
y
```

Out[10]:

```
0         0
1         0
2         0
3         0
4         0
         ..
210784    1
210785    0
210786    0
210787    1
210788    0
Name: 再購(120天), Length: 210789, dtype: int64
```

處理資料不平衡

不平衡資料的二元分類 2：利用抽樣改善模型品質 (https://taweihuang.hpd.io/2018/12/30/imbalanced-data-sampling-techniques/)
Oversampling: SMOTE for binary and categorical data in Python (https://stackoverflow.com/questions/47655813/oversampling-smote-for-binary-and-categorical-data-in-python)

In [11]:

```
cate = [0,1,2,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27] # 這些是類別資料，使用smotenc前要先標出來
```

In [12]:

```
sm = SMOTENC(categorical_features = cate,random_state=0)
X = X.drop(columns=['claim_settle_dt','INSURED_DOB']) # 找不到處理timestamp的資料，決定刪除
X_res, y_res = sm.fit_resample(X, y)
```

In [13]:

```
df1 = X_res
df1['y'] = y_res
```

In [14]:

```
df1.to_excel('理賠再購屬性合併balanced_before_encoding.xlsx')
```

In [15]:

```
df1
```

Out[15]:

	BundleSubtype2	illness_code	DiagnosisCode_DESC	REIMBURSED_YR_TW	累積理賠金
0	5.N疾病醫療	C18	02.腫瘤	46987.500000	46987.5000
1	5.N疾病醫療	C18	02.腫瘤	7087.500000	54075.0000
2	5.N疾病醫療	C18	02.腫瘤	30712.500000	84787.5000
3	5.N疾病醫療	C18	02.腫瘤	7087.500000	91875.0000
4	5.N疾病醫療	C18	02.腫瘤	7087.500000	98962.5000
...	...	...	...	...	
400161	5.N疾病醫療	O75	15.妊娠、分娩和產褥期	96394.841317	96394.8413
400162	6.N意外醫療	W18	20.疾病和死亡的外因	10500.000000	10500.0000
400163	6.N意外醫療	Y93.7	20.疾病和死亡的外因	2650.423746	30129.7687
400164	5.N疾病醫療	O75	15.妊娠、分娩和產褥期	89299.035951	182784.2357
400165	6.N意外醫療	Y99.8	99.不知道不想猜	2625.000000	2625.0000

400166 rows × 30 columns

encoding

In [16]:

```
## 要先分訓練跟測試，才能target encoding
X_train, X_test, y_train, y_test = train_test_split(X_res, y_res, test_size=0.33, random_s
tate=42)
enc = TargetEncoder(cols=['BundleSubtype2', 'illness_code', 'DiagnosisCode_DESC','WEALTH_L
EVEL','stick_level2', 'cust_group2'])
training_numeric_dataset = enc.fit_transform(X_train, y_train)
testing_numeric_dataset = enc.transform(X_test)
```

min_max

In [17]:

```
scaler = MinMaxScaler()
scaler.fit(training_numeric_dataset[['REIMBURSED_YR_TW', '累積理賠金額', '累積理賠次數', 'te
rnure_m', 'recency_m','AGE']])
training_numeric_dataset[['REIMBURSED_YR_TW', '累積理賠金額', '累積理賠次數', 'ternure_m',
'recency_m','AGE']] = scaler.transform(training_numeric_dataset[['REIMBURSED_YR_TW', '累積
理賠金額', '累積理賠次數', 'ternure_m', 'recency_m','AGE']])
testing_numeric_dataset[['REIMBURSED_YR_TW', '累積理賠金額', '累積理賠次數', 'ternure_m', 'r
ecency_m','AGE']] = scaler.transform(testing_numeric_dataset[['REIMBURSED_YR_TW', '累積理賠
金額', '累積理賠次數', 'ternure_m', 'recency_m','AGE']])
```

儲存最後training, testing data

In [18]:

```
training_numeric_dataset.to_excel('training_data.xlsx')
testing_numeric_dataset.to_excel('testing_data.xlsx')
```

In [ ]:

```

```