Chapter 8 - Ex3: Customer Churn Analysis

- Cho dữ liệu WA_Fn-UseC_-Telco-Customer-Churn.csv chứa thông tin khách hàng. Bộ dữ liệu này được dùng để xây dựng mô hình dự đoán một khách hàng kết thúc mối quan hệ hay hủy/không gia hạn (churn) với doanh nghiệp hay không?
- Bộ dữ liệu gồm 7043 mẫu và 21 thuộc tính

Yêu cầu:

- Đọc dữ liệu WA_Fn-UseC_-Telco-Customer-Churn.csv, tiền xử lý dữ liệu.
- Chia dữ liệu thành 2 bộ là train và test theo tỷ lệ 80-20.
- Xem xét tính cân bằng giữa hai loại mẫu ở train. Trực quan hóa. Nhận xét.
- Nếu 2 loại mẫu ở train này không cân bằng, hãy chọn một phương pháp cân bằng dữ liệu và thực hiện. Trực quan hóa kết quả.

- Tham khảo: link (https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/)
- Và link (https://www.kaggle.com/blastchar/telco-customer-churn)

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
data = pd.read_csv("WA_Fn-UseC_-Telco-Customer-Churn.csv")
```

In [4]:
```
data.head()
```

Out[4]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Ir
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	
4	9237-HQITU	Female	0	No	No	2	Yes	No	

5 rows × 21 columns

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
customerID          7043 non-null object
gender              7043 non-null object
SeniorCitizen       7043 non-null int64
Partner             7043 non-null object
Dependents          7043 non-null object
tenure              7043 non-null int64
PhoneService        7043 non-null object
MultipleLines       7043 non-null object
InternetService     7043 non-null object
OnlineSecurity      7043 non-null object
OnlineBackup        7043 non-null object
DeviceProtection    7043 non-null object
TechSupport         7043 non-null object
StreamingTV         7043 non-null object
StreamingMovies     7043 non-null object
Contract            7043 non-null object
PaperlessBilling    7043 non-null object
PaymentMethod       7043 non-null object
MonthlyCharges      7043 non-null float64
TotalCharges        7043 non-null object
Churn               7043 non-null object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

In [6]:
```
# Đếm theo loại: hiếm, phổ biến
occ = data.Churn.value_counts()
occ
```

Out[6]:
```
No     5174
Yes    1869
Name: Churn, dtype: int64
```

In [7]:
```
# Print the ratio of fraud cases
print(occ / len(data.index))
```

```
No     0.73463
Yes    0.26537
Name: Churn, dtype: float64
```

In [8]:
```
X = data.drop(["customerID", "Churn"], axis=1)
y = data.Churn
```

In [9]:
```
X.head()
```

Out[9]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService
0	Female	0	Yes	No	1	No	No phone service	DSL
1	Male	0	No	No	34	Yes	No	DSL
2	Male	0	No	No	2	Yes	No	DSL
3	Male	0	No	No	45	No	No phone service	DSL
4	Female	0	No	No	2	Yes	No	Fiber optic

In [10]:
```
X.MonthlyCharges = X.MonthlyCharges.astype('float')
```

In [11]:
```
X.TotalCharges = pd.to_numeric(X.TotalCharges, errors='coerce')
```

In [12]:
```
y.head()
```

Out[12]:
```
0     No
1     No
2    Yes
3     No
4    Yes
Name: Churn, dtype: object
```

In [13]:
```
# Chuẩn hóa y
# No: 0, Yes: 1
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
y_new = label_encoder.fit_transform(y)
y_new[:5]
```

Out[13]:
```
array([0, 0, 1, 0, 1])
```

In [14]:
```
# Chuẩn hóa X; các cột là Category
# Categorical boolean mask
categorical_feature_mask = X.dtypes==object
# filter categorical columns using mask and turn it into a list
categorical_cols = X.columns[categorical_feature_mask].tolist()
categorical_cols
```

Out[14]:
```
['gender',
 'Partner',
 'Dependents',
 'PhoneService',
 'MultipleLines',
 'InternetService',
 'OnlineSecurity',
 'OnlineBackup',
 'DeviceProtection',
 'TechSupport',
 'StreamingTV',
 'StreamingMovies',
 'Contract',
 'PaperlessBilling',
 'PaymentMethod']
```

In [15]:
```
X_new = pd.get_dummies(data=X, columns=categorical_cols, drop_first=True)
```

In [16]:
```
X_new.head()
```

Out[16]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Male	Partner_Yes	Dependents_Yes
0	0	1	29.85	29.85	0	1	0
1	0	34	56.95	1889.50	1	0	0
2	0	2	53.85	108.15	1	0	0
3	0	45	42.30	1840.75	1	0	0
4	0	2	70.70	151.65	0	0	0

5 rows × 30 columns

In [17]:
```
# Chia dữ liệu thành 2 bộ theo tỷ lệ 80:20
from sklearn.model_selection import train_test_split
```

In [18]:
```
X_train, X_test, y_train, y_test = train_test_split(X_new, y_new,
                                                    test_size = 0.2)
```

In [19]:
```
#X_train.info()
```

In [20]:
```
from collections import Counter
sorted(Counter(y_train).items())
```

Out[20]:
```
[(0, 4141), (1, 1493)]
```

In [21]:
```
# Vì lượng dữ liệu của mỗi lớp đều khá nhiều và theo tỷ lệ 3:1
# Có thể cho undersampling hoặc oversampling
```

In [22]:
```
# undersampling
from imblearn.under_sampling import RandomUnderSampler
```

In [23]:
```
X_train = X_train.fillna(X_train.mean())
X_train.head()
```

Out[23]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Male	Partner_Yes	Dependents_
5228	0	9	44.40	348.15	1	0	
1473	0	24	98.75	2407.30	0	0	
6606	0	38	105.00	4026.40	0	1	
4747	0	35	20.60	754.00	1	0	
1439	1	5	75.55	349.65	0	0	

5 rows × 30 columns

In [24]:
```
rs = RandomUnderSampler()
X_train_resampled, y_train_resampled = rs.fit_resample(X_train, y_train)
```

In [25]:
```
sorted(Counter(y_train_resampled).items())
```

Out[25]:
```
[(0, 1493), (1, 1493)]
```

In [26]:
```
# Oversampling
from imblearn.over_sampling import SMOTE
```

In [27]:
```
X_train_S, y_train_S = SMOTE().fit_resample(X_train, y_train)
```

In [28]:
```
sorted(Counter(y_train_S).items())
```

Out[28]:
```
[(0, 4141), (1, 4141)]
```

In [ ]:
```
```