In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
from sklearn.model_selection import train_test_split
from imblearn.under_sampling import RandomUnderSampler
```

```
<frozen importlib._bootstrap>:219: RuntimeWarning: numpy.ufunc size changed, may indicate binary incomp
atibility. Expected 192 from C header, got 216 from PyObject
<frozen importlib._bootstrap>:219: RuntimeWarning: numpy.ufunc size changed, may indicate binary incomp
atibility. Expected 192 from C header, got 216 from PyObject
```

In [2]:

```
#pip install imblearn
```

In [3]:

```
df = pd.read_csv("bank-full-encoded.csv")
df.head()
```

Out[3]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	40	4	1	2	0	3036	1	0	2	4	8	261	0	0	0	3	0
1	26	9	2	1	0	945	1	0	2	4	8	151	0	0	0	3	0
2	15	2	1	1	0	918	1	1	2	4	8	76	0	0	0	3	0
3	29	1	1	3	0	2420	1	0	2	4	8	92	0	0	0	3	0
4	15	11	2	3	0	917	0	0	2	4	8	198	0	0	0	3	0

In [6]:

```
df['y'].value_counts()
```

Out[6]:

```
0    39922
1     5289
Name: y, dtype: int64
```

In [ ]:

```

```

In [7]:

```
# features and target
X = df.iloc[:,:-1]
y = df.iloc[:,-1]
```

In [8]:

```
# train test splitting
X_train, X_test,y_train,y_test = train_test_split(X,y,test_size=0.30,random_state=0)
```

In [20]:

```
# before applying sampling technique
from sklearn.svm import SVC
from sklearn.metrics import classification_report,roc_auc_score
model_normal = SVC()
clf = model_normal.fit(X_train,y_train)
pred = clf.predict(X_test)
print("ROC AUC score : ", roc_auc_score(y_test,pred))
```

```
ROC AUC score :  0.5270655567635408
```

In [13]:

```
# define undersample strategy
undersample = RandomUnderSampler(sampling_strategy='majority')
# fit and transform
X_train_undersample,y_train_undersample = undersample.fit_resample(X_train,y_train)
```

In [18]:

```
from collections import Counter
print("before undersampling : ",Counter(y_train))
print("after undersampling : ",Counter(y_train_undersample))
```

```
before undersampling :  Counter({0: 27953, 1: 3694})
after undersampling :  Counter({0: 3694, 1: 3694})
```

In [19]:

```
# after applying sampling technique
model = SVC()
clf_undersample = model.fit(X_train_undersample,y_train_undersample)
pred_undersample = clf_undersample.predict(X_test)
print("ROC AUC score : ", roc_auc_score(y_test,pred_undersample))
```

```
ROC AUC score :  0.7375888233736526
```

In [ ]:

```

```

In [21]:

```
# Oversampling
from imblearn.over_sampling import SMOTE
```

In [24]:

```
# define undersample strategy
oversample = SMOTE()
# fit and transform
X_train_oversample,y_train_oversample = oversample.fit_resample(X_train,y_train)
from collections import Counter
print("before oversampling : ",Counter(y_train))
print("after oversampling : ",Counter(y_train_oversample))
# after applying sampling technique
model_oversample = SVC()
clf_oversample = model_oversample.fit(X_train_oversample,y_train_oversample)
pred_oversample = clf_oversample.predict(X_test)
print("ROC AUC score : ", roc_auc_score(y_test,pred_oversample))
```

```
before oversampling :  Counter({0: 27953, 1: 3694})
after oversampling :  Counter({0: 27953, 1: 27953})
ROC AUC score :  0.7445553573481757
```

In [ ]:

```

```