In [2]:
```
import random
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from imblearn.over_sampling import RandomOverSampler
from collections import Counter
np.random.seed(1000)  # any value to remove randomeness in prediction
```

In [3]:
```
allFeatures16=["LOC","WMC","DIT","NOC","CBO","RFC","LCOM","CA","CE","NPM
allfeatures18=["LOC","WMC","DIT","NOC","CBO","RFC","LCOM","CA","CE","NPM
allfeatures19=["LOC","WMC","DIT","NOC","CBO","RFC","LCOM","CA","CE","NPM
```

In [4]:
```
import pandas as pd

# Load dataset
ds16 = pd.read_csv("Ant16Data.csv") #training
ds18 = pd.read_csv("Ant18Data.csv") #training
ds19 = pd.read_csv("Ant19Data.csv") #training
```

In [5]:
```
#Ant 16
import statsmodels.api as smf
from statsmodels.datasets.longley import load_pandas

#ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA","CBM","DAM","CA"
allFeatures16=["LOC","WMC","DIT","NOC","CBO","RFC","LCOM","CA","CE","NPM


X16 = ds16[allFeatures16] # Features  X_train
y16 = ds16.Sum_Churn # Target variable     y_train

#resample ant 16
from imblearn.over_sampling import SMOTE
smote = SMOTE()
X16, y16 = smote.fit_resample(X16, y16)

df16 = pd.concat([pd.DataFrame(X16), pd.DataFrame(y16)], axis=1) #sample
```

In [6]:
```
#Ant 18
import statsmodels.api as smf
from statsmodels.datasets.longley import load_pandas

#ant18_input_features=["MFA","CAM","MOA","LCOM","AMC","CE","DAM","NOC",'
allfeatures18=["LOC","WMC","DIT","NOC","CBO","RFC","LCOM","CA","CE","NPM


X18 = ds18[allfeatures18] # Features  X_train
y18 = ds18.Sum_Churn # Target variable     y_train

#resample ant 18
from imblearn.over_sampling import SMOTE
smote = SMOTE()
X18, y18 = smote.fit_resample(X18, y18)

df18 = pd.concat([pd.DataFrame(X18), pd.DataFrame(y18)], axis=1) #sample
```

In [7]:
```
#Ant 19
import statsmodels.api as smf
from statsmodels.datasets.longley import load_pandas

ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE","NOC",'

X19 = ds19[ant19_input_features] # Features  X_train
y19 = ds19.Sum_Churn # Target variable     y_train

#resample ant 18
from imblearn.over_sampling import SMOTE
smote = SMOTE()
X19, y19 = smote.fit_resample(X19, y19)

df19 = pd.concat([pd.DataFrame(X19), pd.DataFrame(y19)], axis=1) #sample
```

In [8]:
```
from sklearn.metrics import confusion_matrix
import numpy as np
def printCM():
    print(y_pred.size)  #predicted probabilities using the final model.
    threshold=0.5
    predicted_class1=np.zeros(y_pred.shape)
    predicted_class1[y_pred>threshold]=1
    cm1 = confusion_matrix(ytest,predicted_class1)
    print('Confusion Matrix : \n', cm1)
    return cm1
```

In [9]:
```
#training Ant 16
smlog = smf.Logit(y19,smf.add_constant(X19),formula = 'Sum_Churn ~ CA +
smlog.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.618050
         Iterations 22

C:\Users\Haneen\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2
629: FutureWarning: Method .ptp is deprecated and will be removed in a
future version. Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

Out[9]:

Logit Regression Results

Dep. Variable:	Sum_Churn	**No. Observations:**	810
Model:	Logit	**Df Residuals:**	798
Method:	MLE	**Df Model:**	11
Date:	Sat, 11 May 2024	**Pseudo R-squ.:**	0.1083
Time:	10:20:23	**Log-Likelihood:**	-500.62
converged:	True	**LL-Null:**	-561.45
Covariance Type:	nonrobust	**LLR p-value:**	8.414e-21

	coef	std err	z	P>|z|	[0.025	0.975]
const	-5.4207	1.9e+06	-2.85e-06	1.000	-3.73e+06	3.73e+06
CA	0.9015	0.350	2.573	0.010	0.215	1.588
AMC	0.7563	0.331	2.284	0.022	0.107	1.405
MOA	0.3240	0.152	2.129	0.033	0.026	0.622
CAM	-0.4705	0.115	-4.095	0.000	-0.696	-0.245
MaX_CC	0.1949	0.167	1.164	0.245	-0.133	0.523
DIT	-0.1861	0.131	-1.419	0.156	-0.443	0.071
CE	0.1266	0.152	0.832	0.405	-0.172	0.425
NOC	-0.0662	0.307	-0.216	0.829	-0.667	0.535
LCOM	0.1922	0.414	0.464	0.642	-0.619	1.003
DAM	-0.0472	0.088	-0.538	0.591	-0.219	0.125
MFA	-26.3716	8.89e+06	-2.97e-06	1.000	-1.74e+07	1.74e+07

In [10]:
```
def printTesingMeasurements():
    cm1= printCM()
    accuracy=(cm1[0,0]+cm1[1,1])/(cm1[0,0]+cm1[0,1]+cm1[1,0]+cm1[1,1])
    print('accuracy : ', accuracy )
    sensitivity = cm1[1,1]/(cm1[1,0]+cm1[1,1])
    print('Sensitivity : ', sensitivity)
    specificity = cm1[0,0]/(cm1[0,0]+cm1[0,1])
    print('Specificity : ', specificity )
```

In [11]:
```
print("Testing Ant19 with Ant19")

ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE","NOC",'

# defining the dependent and independent variables
Xtest = df19[ant19_input_features] #Ant16 features that are
ytest = df19['Sum_Churn']

#x_pred = np.linspace(x.min(), x.max(), 50)
# put the X matrix in 'standard' form, i.e. with a column of ones.
x_matrix = smf.add_constant(Xtest)
y_pred = smlog.predict(x_matrix)   # y_pred is now bound to an array of
printTesingMeasurements()
```

```
Testing Ant19 with Ant19
810
Confusion Matrix :
 [[273 132]
 [151 254]]
accuracy :  0.6506172839506172
Sensitivity :  0.6271604938271605
Specificity :  0.674074074074074
```

In [12]:
```
print("Testing Ant19 with Ant16")

ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE","NOC",'


# defining the dependent and independent variables
Xtest = df16[ant19_input_features] #Ant16 features that are
ytest = df16['Sum_Churn']


#x_pred = np.linspace(x.min(), x.max(), 50)
# put the X matrix in 'standard' form, i.e. with a column of ones.
x_matrix = smf.add_constant(Xtest)
y_pred = smlog.predict(x_matrix)
printTesingMeasurements()
```

```
Testing Ant19 with Ant16
1044
Confusion Matrix :
 [[147 375]
 [237 285]]
accuracy :  0.41379310344827586
Sensitivity :  0.5459770114942529
Specificity :  0.28160919540229884

C:\Users\Haneen\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2
629: FutureWarning: Method .ptp is deprecated and will be removed in a
future version. Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

In [13]:
```
print("Testing Ant19 with Ant18")

ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE","NOC",'

# defining the dependent and independent variables
Xtest = df18[ant19_input_features] #Ant16 features that are
ytest = df18['Sum_Churn']

#x_pred = np.linspace(x.min(), x.max(), 50)
# put the X matrix in 'standard' form, i.e. with a column of ones.
x_matrix = smf.add_constant(Xtest)
y_pred = smlog.predict(x_matrix)
printTesingMeasurements()
```

```
Testing Ant19 with Ant18
834
Confusion Matrix :
 [[304 113]
 [169 248]]
accuracy :  0.6618705035971223
Sensitivity :  0.5947242206235012
Specificity :  0.7290167865707434

C:\Users\Haneen\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2
629: FutureWarning: Method .ptp is deprecated and will be removed in a
future version. Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

In [ ]:

In [ ]:

In [ ]: