In [5]:
```
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 [6]:
```
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 [7]:
```
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 [8]:
```
#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 [9]:
```
#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","C

X18 = ds18[ant18_input_features] # 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 [10]:

```
#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",
allfeatures19=["LOC","WMC","DIT","NOC","CBO","RFC","LCOM","CA","CE","NPM

X19 = ds19[allfeatures19] # 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 [11]:
```
smlog = smf.Logit(y18,smf.add_constant(X18),formula = 'Sum_Churn ~ CAM +
smlog.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.611062
         Iterations 34

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[11]:

Logit Regression Results

Dep. Variable:	Sum_Churn	**No. Observations:**	834
Model:	Logit	**Df Residuals:**	822
Method:	MLE	**Df Model:**	11
Date:	Sat, 11 May 2024	**Pseudo R-squ.:**	0.1184
Time:	10:17:07	**Log-Likelihood:**	-509.63
converged:	True	**LL-Null:**	-578.08
Covariance Type:	nonrobust	**LLR p-value:**	6.893e-24

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	23.5206	2.46e+06	9.56e-06	1.000	-4.82e+06	4.82e+06
MFA	111.1548	1.18e+07	9.45e-06	1.000	-2.3e+07	2.3e+07
CAM	-0.4356	0.117	-3.717	0.000	-0.665	-0.206
MOA	0.5899	0.160	3.693	0.000	0.277	0.903
LCOM	0.7991	0.502	1.593	0.111	-0.184	1.782
AMC	0.2606	0.330	0.790	0.430	-0.386	0.907
CE	0.1554	0.168	0.926	0.354	-0.173	0.484
DAM	-0.1205	0.085	-1.420	0.156	-0.287	0.046
NOC	-0.3963	0.302	-1.313	0.189	-0.988	0.195
CA	0.2080	0.320	0.650	0.516	-0.419	0.835
MaX_CC	0.1336	0.173	0.773	0.439	-0.205	0.472
DIT	-0.0755	0.128	-0.591	0.555	-0.326	0.175

In [12]:
```
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 [13]:
```
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 [14]:
```
# Loading the testing dataset
print("Testing Ant18 with Ant18")

AllFeaturesIn18=["MFA","CAM","MOA","LCOM","AMC","CE","DAM","NOC","CA","N

# defining the dependent and independent variables
Xtest = df18[AllFeaturesIn18] #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 Ant18 with Ant18
834
Confusion Matrix :
 [[300 117]
 [165 252]]
accuracy :  0.6618705035971223
Sensitivity :  0.60431654676259
Specificity :  0.7194244604316546
```

In [15]:
```
print("Testing Ant18 with Ant16")

ant18_input_features=["MFA","CAM","MOA","LCOM","AMC","CE","DAM","NOC","C

# defining the dependent and independent variables
Xtest = df16[ant18_input_features] #Ant16 features that are
ytest = df16['Sum_Churn']

x_matrix = smf.add_constant(Xtest)
y_pred = smlog.predict(x_matrix)
printTesingMeasurements()
```

```
Testing Ant18 with Ant16
1044
Confusion Matrix :
 [[376 146]
 [285 237]]
accuracy :  0.5871647509578544
Sensitivity :  0.4540229885057471
Specificity :  0.7203065134099617

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 [16]:

```
# Loading the testing dataset
print("Testing Ant18 with Ant19")

ant18_input_features=["MFA","CAM","MOA","LCOM","AMC","CE","DAM","NOC","C

# defining the dependent and independent variables
Xtest = df19[ant18_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)
printTesingMeasurements()
```

```
Testing Ant18 with Ant19
810
Confusion Matrix :
 [[359  46]
 [264 141]]
accuracy :  0.6172839506172839
Sensitivity :  0.34814814814814815
Specificity :  0.8864197530864197

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 [ ]: