In [25]:
```
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 [26]:
```
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 [27]:
```
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 [28]:
```
#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",

X16 = ds16[ant16_input_features] # 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 [29]:
```
#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 [30]:
```
#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 [31]:
```
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 [32]:
```
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 [33]:
```
#training Ant 16
smlog = smf.Logit(y16,smf.add_constant(X16),formula = 'Sum_Churn ~ MaX_C
smlog.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.523231
         Iterations 8

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[33]:

Logit Regression Results

Dep. Variable:	Sum_Churn	**No. Observations:**	1044
Model:	Logit	**Df Residuals:**	1032
Method:	MLE	**Df Model:**	11
Date:	Wed, 05 Jun 2024	**Pseudo R-squ.:**	0.2451
Time:	20:56:49	**Log-Likelihood:**	-546.25
converged:	True	**LL-Null:**	-723.65
Covariance Type:	nonrobust	**LLR p-value:**	2.354e-69

	coef	std err	z	P>|z|	[0.025	0.975]
const	2.6883	0.334	8.055	0.000	2.034	3.342
MaX_CC	1.0750	0.316	3.407	0.001	0.456	1.694
CE	1.0142	0.182	5.564	0.000	0.657	1.371
MOA	1.5032	0.333	4.514	0.000	0.851	2.156
LCOM	4.0184	1.203	3.339	0.001	1.660	6.377
MFA	-0.0251	0.090	-0.278	0.781	-0.202	0.151
CBM	0.5521	0.200	2.756	0.006	0.160	0.945
DAM	-0.1404	0.080	-1.754	0.079	-0.297	0.016
CA	0.3425	0.445	0.770	0.441	-0.529	1.214
NOC	0.3776	0.409	0.923	0.356	-0.424	1.179
CAM	0.1066	0.114	0.939	0.348	-0.116	0.329
AMC	1.2505	0.432	2.893	0.004	0.403	2.098

In [34]:
```
print("Testing Ant16 with Ant16")
ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA","CBM","DAM","CA",

# defining the dependent and independent variables
Xtest = df16[ant16_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)
#import numpy as np

#x_matrix=np.c_[np.ones((565,1)),x_matrix] #add col of ones or any other
y_pred = smlog.predict(x_matrix)

printTesingMeasurements()
```

```
Testing Ant16 with Ant16
1044
Confusion Matrix :
 [[428  94]
 [190 332]]
accuracy :  0.7279693486590039
Sensitivity :  0.6360153256704981
Specificity :  0.8199233716475096
```

In [35]:
```
# loading the testing dataset
print("Testing Ant16 with Ant18")

ant16_input_featuresEXC_CBM=["MaX_CC","CE","MOA","LCOM","MFA","DAM","CA"

# defining the dependent and independent variables
Xtest = df18[ant16_input_featuresEXC_CBM] #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)

import numpy as np
x_matrix=np.c_[np.ones((834,1)),x_matrix] #add col of ones or any other
y_pred = smlog.predict(x_matrix)
printTesingMeasurements()
```

```
Testing Ant16 with Ant18
834
Confusion Matrix :
 [[121 296]
 [ 52 365]]
accuracy :  0.5827338129496403
Sensitivity :  0.8752997601918465
Specificity :  0.290167865707434

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 [36]:
```
# loading the testing dataset
print("Testing Ant16 with Ant19")

ant16_input_featuresEXC_CBM=["MaX_CC","CE","MOA","LCOM","MFA","DAM","CA'

# defining the dependent and independent variables
Xtest = df19[ant16_input_featuresEXC_CBM] #Ant16 features that are CBM
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)
#print(x_matrix)
y_pred = smlog.predict(x_matrix)


printTesingMeasurements()
```

```
Testing Ant16 with Ant19
810
Confusion Matrix :
 [[159 246]
 [ 85 320]]
accuracy :  0.591358024691358
Sensitivity :  0.7901234567901234
Specificity :  0.3925925925925926

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 [ ]: