In [1]:
```
import pandas as pd

# Load dataset
ds = pd.read_csv("Ant16Data.csv")
ds.head()
```

Out[1]:

		COMP	LOC	WMC	NOC	CB
0	org.apache.tools.ant.taskdefs.XSLTLoggerAware....	-0.730244	-0.896416	-0.146396	-0.3156(
1	org.apache.tools.ant.taskdefs.Recorder.java	-0.235684	-0.205915	-0.146396	-0.1041⁴
2	org.apache.tools.ant.util.facade.Implementatio...	-0.674519	-0.723791	0.241780	-0.2098⁷
3	org.apache.tools.bzip2.CRC.java	1.779710	-0.378540	-0.146396	-0.3508⁵
4	org.apache.tools.ant.taskdefs.optional.depend....	-0.477159	-0.378540	-0.146396	-0.2803⁶

5 rows × 21 columns

In [76]:
```
#ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA","CBM","DAM","CA'
#ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA","CBM","DAM","CA'
#ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA","CBM","DAM","CA'
#ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA","CBM","DAM","CA'
#ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA","CBM","DAM"]
#ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA","CBM"]
#ant16_input_features=["MaX_CC","CE","MOA","LCOM","MFA"]
#ant16_input_features=["MaX_CC","CE","MOA","LCOM"]
#ant16_input_features=["MaX_CC","CE","MOA"]
ant16_input_features=["MaX_CC","CE"]#optimal
#ant16_input_features=["MaX_CC"]
#ant16_input_features=["CE"]









X = ds[ant16_input_features] # Features
y = ds.Sum_Churn # Target variable
```

In [77]:
```
import statsmodels.api as sm
smlog = sm.Logit(y,sm.add_constant(X)).fit(maxiter=10000000)
smlog.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.240138
         Iterations 8
```

Out[77]:

Logit Regression Results

Dep. Variable:	Sum_Churn	**No. Observations:**	565
Model:	Logit	**Df Residuals:**	562
Method:	MLE	**Df Model:**	2
Date:	Wed, 08 May 2024	**Pseudo R-squ.:**	0.1078
Time:	12:21:32	**Log-Likelihood:**	-135.68
converged:	True	**LL-Null:**	-152.07
Covariance Type:	nonrobust	**LLR p-value:**	7.584e-08

	coef	std err	z	P>|z|	[0.025	0.975]
const	3.4822	0.358	9.717	0.000	2.780	4.185
MaX_CC	1.3992	0.620	2.256	0.024	0.184	2.615
CE	1.1209	0.347	3.233	0.001	0.441	1.800

In [78]:
```
#odds ratio
import numpy as np
np.exp(smlog.params)
```

Out[78]:
```
const     32.531432
MaX_CC     4.051933
CE         3.067607
dtype: float64
```

In [79]:
```
#calculate Variance Inflation Factor
from statsmodels.stats.outliers_influence import variance_inflation_fact
vif_scores = pd.DataFrame()
vif_scores["Attribute"] = X.columns

# calculating VIF for each feature
vif_scores["VIF Scores"] = [variance_inflation_factor(X.values, i) for i

display(vif_scores)
```

	Attribute	VIF Scores
0	MaX_CC	1.290843
1	CE	1.290843

In [75]:
```
from scipy.stats.distributions import chi2
def likelihood_ratio(reduced_ll, full_ll):
    return(-2*(reduced_ll-full_ll))

afterll=-139.17



beforell=-135.68









LR = likelihood_ratio(afterll,beforell)
p = chi2.sf(LR, 1) # 1 DoF coz diff between variable in model



print(LR)
print(p)
```

```
6.979999999999961
0.008242560884641844
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: