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

# load dataset
ds = pd.read_csv("Ant19Data.csv")
ds.head()
```

Out[1]:

	COMP	LOC	WMC	DIT	NOC
0	org.apache.tools.ant.util.regexp.JakartaRegexp...	-0.455390	-0.644382	-0.660476	-0.14192
1	org.apache.tools.ant.taskdefs.GUnzip.java	-0.403774	-0.569526	-0.660476	-0.14192
2	org.apache.tools.ant.taskdefs.condition.Equals...	-0.201800	-0.045537	0.654215	-0.14192
3	org.apache.tools.ant.taskdefs.optional.ccm.CCM...	-0.628190	-0.794093	-0.660476	-0.14192
4	org.apache.tools.ant.listener.Log4jListener.java	-0.107546	-0.270104	0.654215	-0.14192

In [82]:
```
#ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE","NOC",
#ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE","NOC",
#ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE","NOC",
#ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE","NOC"]
#ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT","CE"]
#ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC","DIT"]
#ant19_input_features=["CA","AMC","MOA","CAM","MaX_CC"]
ant19_input_features=["CA","AMC","MOA","CAM"] #optimal
#ant19_input_features=["CA","AMC","MOA"]
#ant19_input_features=["CA","AMC","CAM"]
#ant19_input_features=["CA","MOA","CAM"]
#ant19_input_features=["AMC","MOA","CAM"]



X = ds[ant19_input_features] # Features
y = ds.Sum_Churn # Target variable
```

In [83]:
```
import statsmodels.api as sm
smlog = sm.Logit(y,sm.add_constant(X)).fit(maxiter=10000000)
smlog.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.620268
         Iterations 5
```

Out[83]:

Logit Regression Results

Dep. Variable:	Sum_Churn	**No. Observations:**	724
Model:	Logit	**Df Residuals:**	719
Method:	MLE	**Df Model:**	4
Date:	Wed, 17 Apr 2024	**Pseudo R-squ.:**	0.09592
Time:	13:35:29	**Log-Likelihood:**	-449.07
converged:	True	**LL-Null:**	-496.72
Covariance Type:	nonrobust	**LLR p-value:**	9.894e-20

	coef	std err	z	P>|z|	[0.025	0.975]
const	-0.0361	0.091	-0.395	0.693	-0.215	0.143
CA	0.7932	0.322	2.463	0.014	0.162	1.424
AMC	0.8313	0.337	2.468	0.014	0.171	1.492
MOA	0.3638	0.144	2.535	0.011	0.083	0.645
CAM	-0.5704	0.098	-5.793	0.000	-0.763	-0.377

In [84]:
```
#odds ratio
import numpy as np
np.exp(smlog.params)
```

Out[84]:
```
const    0.964558
CA       2.210560
AMC      2.296402
MOA      1.438850
CAM      0.565287
dtype: float64
```

In [85]:
```
#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	CA	1.015828
1	AMC	1.064606
2	MOA	1.272112
3	CAM	1.305850

In [86]:
```
from scipy.stats.distributions import chi2
def likelihood_ratio(reduced_ll, full_ll):
    return(-2*(reduced_ll-full_ll))

afterll=-452.41









beforell=-449.07









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



print(LR)
print(p)
```

```
6.680000000000064
0.0097500624568861
```

In [ ]:
```

```

In [ ]:

In [ ]: