Lorenzo Rega

UNI: lr3103

Homework 2

1.

To run the logistic regression, we read the excel file and select only the loans we are interested in.

In [2]:
```
import pandas as pd

df_nomis = pd.read_excel('Nomis_data.xlsx')

df_nomis.head()
```

Out[2]:

	Tier	FICO	Approve Date	Term	Amount	Previous Rate	Car Type	Competition rate	Outcome	Rate	Cost of Funds	Partner Bin
0	3	695	2002-07-01	72	35000.0		N	6.25	0	7.49	1.8388	1
1	1	751	2002-07-01	60	40000.0		N	5.65	0	5.49	1.8388	3
2	1	731	2002-07-01	60	18064.0		N	5.65	0	5.49	1.8388	3
3	4	652	2002-07-01	72	15415.0		N	6.25	0	8.99	1.8388	3
4	1	730	2002-07-01	48	32000.0		N	5.65	0	5.49	1.8388	1

In [3]:
```
df_segment = df_nomis[(df_nomis['Car Type'] == 'U')
                      & (df_nomis['FICO'] >= 684)
                      & (df_nomis['FICO'] <= 712)
                      & (df_nomis['Term'] == 60)
                      & (df_nomis['Amount'] >= 17800)
                      & (df_nomis['Amount'] <= 25000)].copy()
```

We then have to add a column to our data: the Monthly Payment.

In [5]:
```
!pip install numpy_financial
```

```
Collecting numpy_financial
  Downloading numpy_financial-1.0.0-py3-none-any.whl (14 kB)
Requirement already satisfied: numpy>=1.15 in c:\users\lorenzo\anaconda3\lib\site-packages (from numpy_financial) (1.21.5)
Installing collected packages: numpy-financial
Successfully installed numpy-financial-1.0.0
```

In [7]:
```
import numpy_financial as npf


df_segment["Monthly_Payment"]=-npf.pmt(df_segment["Rate"]/(100*12), df_segment["Term"], df_segment["Amount"])
```

Finally, we run the logistic regression:

In [6]:
```
import statsmodels.formula.api as smf
logistic_reg = smf.logit('Outcome ~ FICO+ Rate+ Monthly_Payment', data=df_segment).fit()

logistic_reg.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.487058
         Iterations 7
```

Out[6]:

Logit Regression Results

Dep. Variable:	Outcome	**No. Observations:**	1540
Model:	Logit	**Df Residuals:**	1536
Method:	MLE	**Df Model:**	3
Date:	Thu, 13 Oct 2022	**Pseudo R-squ.:**	0.2500
Time:	20:36:41	**Log-Likelihood:**	-750.07
converged:	True	**LL-Null:**	-1000.1
Covariance Type:	nonrobust	**LLR p-value:**	4.506e-108

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	39.4574	5.854	6.740	0.000	27.983	50.931
FICO	-0.0417	0.008	-5.136	0.000	-0.058	-0.026
Rate	-1.1269	0.074	-15.179	0.000	-1.272	-0.981
Monthly_Payment	-0.0095	0.001	-6.391	0.000	-0.012	-0.007

We are using FICO, Rate, and Monthly_Payment as our independent variables and the Outcome as our dependent variable. We must use logistic regression and not the linear one because the outcome can only be defined between zero and one. Using linear regression, we could have had outcomes<0 or outcomes>1.

2.

The coefficients of the three parameters are all statistically significant. We can see that from their p-values which are all zeros. If the p-value is zero, it means we cannot reject the null hypothesis which states that the coefficient is equal to zero. The coefficients are all negative, which means that the relation of each variable with the outcome is negative. The more these variables get higher, the more the outcome decreases.

3.

We read the data that we want to use to predict the outcome. We must add the “Monthly_Payment” column just like we did in the previous case:

In [21]:
```
df_test = pd.read_excel('predict.xlsx', sheet_name = 'Sheet1')

df_test["Monthly_Payment"]=-npf.pmt(df_test["Rate"]/(100*12), df_test["Term"], df_test["Amount"])

df_test
```

Out[21]:

	Tier	Approve Date	Term	Amount	Car Type	Competition APR	Cost of funds	Partner Bin	FICO	Rate	Monthly_Payment
0	2	2004-11-19	60	18000	U	4.85	2.13	1	705	6	347.990428
1	2	2004-11-20	60	25000	U	4.85	2.13	1	705	6	483.320038

Finally, we use the predict function from statsmodels in order to predict whether these loans will be accepted or not.

```
In [22]: result=logistic_reg.predict(df_test)
         result

Out[22]: 0    0.499952
         1    0.216838
         dtype: float64
```

According to our model, there is almost a 50% chance for the first loan and a 21.7% for the second one to be accepted.