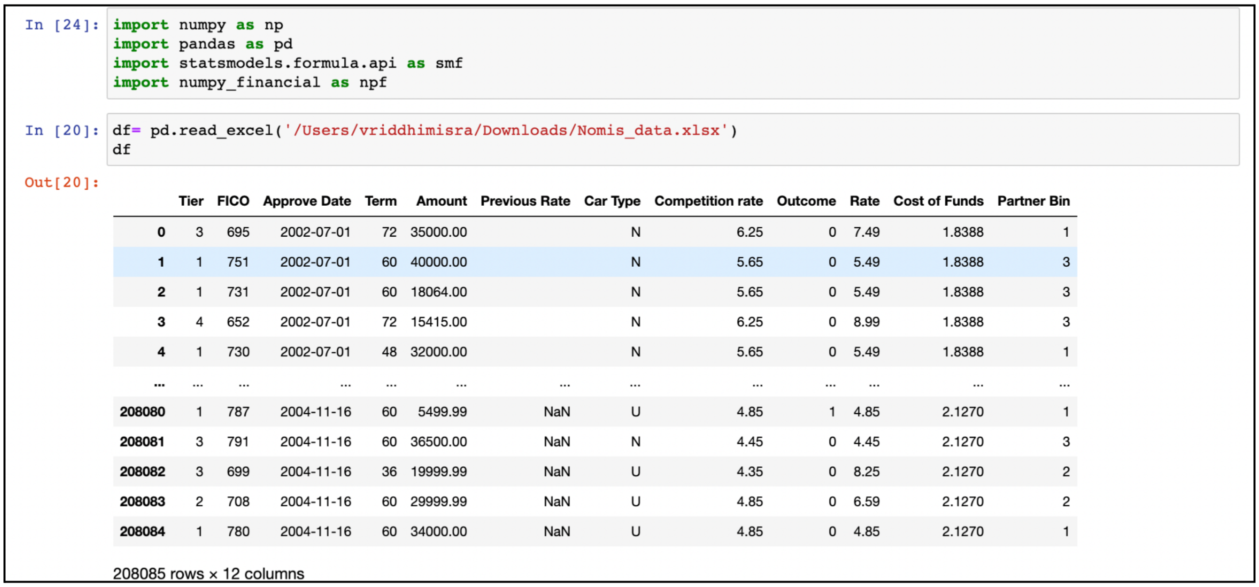
# HOMEWORK 2

**1.** Use statsmodels to fit a logistic regression on the Nomis data. Use the borrower's FICO score, the loan's APR, as well as the monthly payment as parameters. Note: the monthly payment is not found in the data and will need to be calculated as shown in class.

**Ans:** Nomis Solutions was a venture that focused on price optimization solutions. Pricing and revenue optimization includes revenue management and determining how to set and update the prices offered for a portfolio of products in order to maximize expected profitability. We are provided with the Nomis Data for e-Car, which we import as a Pandas DataFrame from the Nomis_data excel sheet.



```
In [24]: import numpy as np
         import pandas as pd
         import statsmodels.formula.api as smf
         import numpy_financial as npf
```

```
In [20]: df= pd.read_excel('/Users/vriddhimisra/Downloads/Nomis_data.xlsx')
         df
```

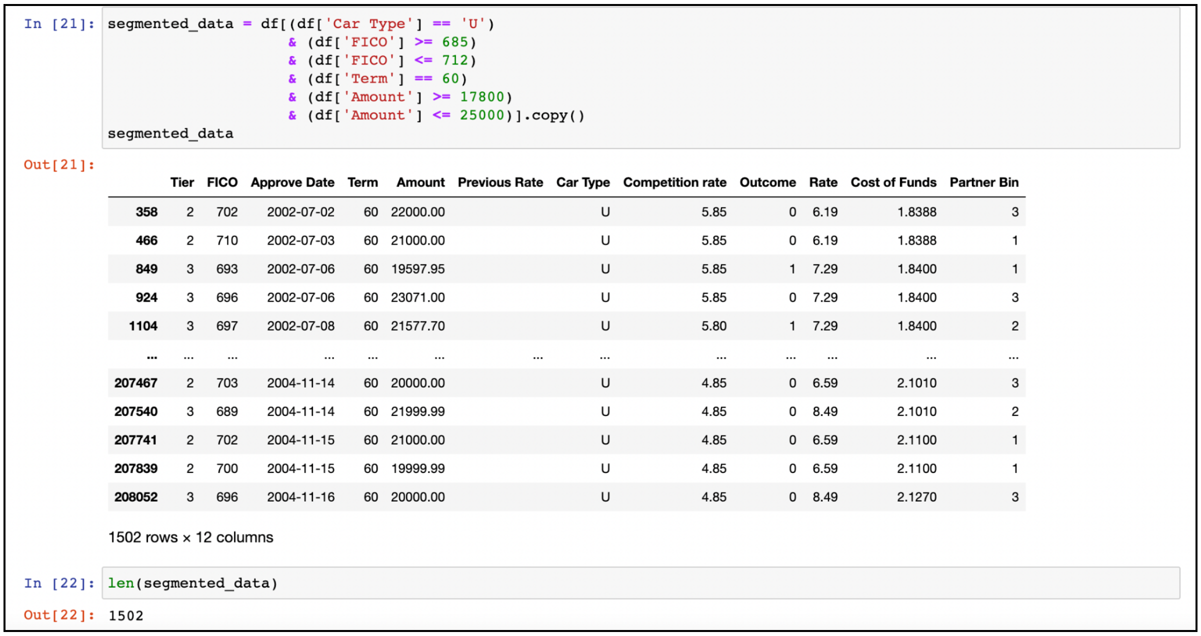
Out[20]:

| | 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 | 35000.00 | | N | 6.25 | 0 | 7.49 | 1.8388 | 1 |
| **1** | 1 | 751 | 2002-07-01 | 60 | 40000.00 | | N | 5.65 | 0 | 5.49 | 1.8388 | 3 |
| **2** | 1 | 731 | 2002-07-01 | 60 | 18064.00 | | N | 5.65 | 0 | 5.49 | 1.8388 | 3 |
| **3** | 4 | 652 | 2002-07-01 | 72 | 15415.00 | | N | 6.25 | 0 | 8.99 | 1.8388 | 3 |
| **4** | 1 | 730 | 2002-07-01 | 48 | 32000.00 | | N | 5.65 | 0 | 5.49 | 1.8388 | 1 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **208080** | 1 | 787 | 2004-11-16 | 60 | 5499.99 | NaN | U | 4.85 | 1 | 4.85 | 2.1270 | 1 |
| **208081** | 3 | 791 | 2004-11-16 | 60 | 36500.00 | NaN | N | 4.45 | 0 | 4.45 | 2.1270 | 3 |
| **208082** | 3 | 699 | 2004-11-16 | 36 | 19999.99 | NaN | U | 4.35 | 0 | 8.25 | 2.1270 | 2 |
| **208083** | 2 | 708 | 2004-11-16 | 60 | 29999.99 | NaN | U | 4.85 | 0 | 6.59 | 2.1270 | 2 |
| **208084** | 1 | 780 | 2004-11-16 | 60 | 34000.00 | NaN | U | 4.85 | 0 | 4.85 | 2.1270 | 1 |

208085 rows × 12 columns

In the data we want to focus on the loans with the following characteristics:
Car Type: 'U': Used cars
FICO: Borrowers with FICO scores between 684 and 712
Term: 60 months
Amount: Loan amounts between $17,800 and $25,000

It will be better to begin with simple, small parts of the 208085 rows of data we have. Thus, in order to segment the data, we apply the conditions for the characteristics of the data frame columns.



```
In [21]: segmented_data = df[(df['Car Type'] == 'U')
                          & (df['FICO'] >= 685)
                          & (df['FICO'] <= 712)
                          & (df['Term'] == 60)
                          & (df['Amount'] >= 17800)
                          & (df['Amount'] <= 25000)].copy()
         segmented_data
```

Out[21]:

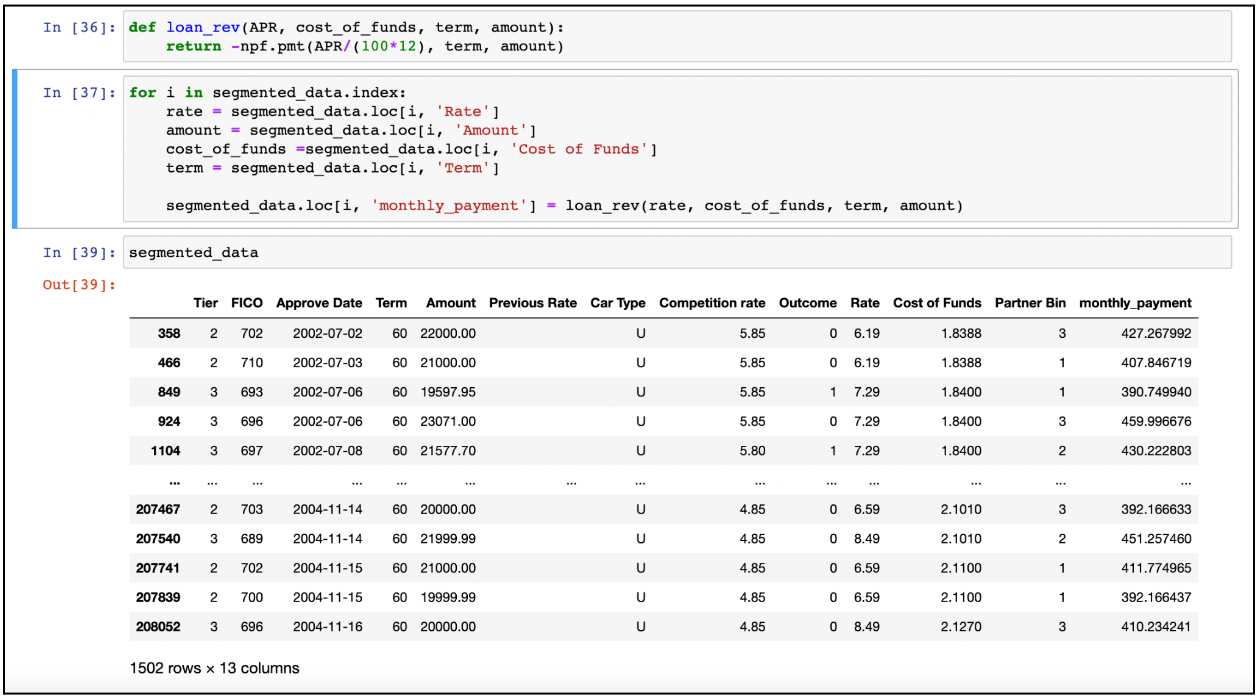
| | Tier | FICO | Approve Date | Term | Amount | Previous Rate | Car Type | Competition rate | Outcome | Rate | Cost of Funds | Partner Bin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **358** | 2 | 702 | 2002-07-02 | 60 | 22000.00 | | U | 5.85 | 0 | 6.19 | 1.8388 | 3 |
| **466** | 2 | 710 | 2002-07-03 | 60 | 21000.00 | | U | 5.85 | 0 | 6.19 | 1.8388 | 1 |
| **849** | 3 | 693 | 2002-07-06 | 60 | 19597.95 | | U | 5.85 | 1 | 7.29 | 1.8400 | 1 |
| **924** | 3 | 696 | 2002-07-06 | 60 | 23071.00 | | U | 5.85 | 0 | 7.29 | 1.8400 | 3 |
| **1104** | 3 | 697 | 2002-07-08 | 60 | 21577.70 | | U | 5.80 | 1 | 7.29 | 1.8400 | 2 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **207467** | 2 | 703 | 2004-11-14 | 60 | 20000.00 | | U | 4.85 | 0 | 6.59 | 2.1010 | 3 |
| **207540** | 3 | 689 | 2004-11-14 | 60 | 21999.99 | | U | 4.85 | 0 | 8.49 | 2.1010 | 2 |
| **207741** | 2 | 702 | 2004-11-15 | 60 | 21000.00 | | U | 4.85 | 0 | 6.59 | 2.1100 | 1 |
| **207839** | 2 | 700 | 2004-11-15 | 60 | 19999.99 | | U | 4.85 | 0 | 6.59 | 2.1100 | 1 |
| **208052** | 3 | 696 | 2004-11-16 | 60 | 20000.00 | | U | 4.85 | 0 | 8.49 | 2.1270 | 3 |

1502 rows × 12 columns

```
In [22]: len(segmented_data)

Out[22]: 1502
```

From the case study, we know that the e-Car mostly prices loans in the 6-6.5% segment. But we need to determine e-Car's revenue in each segment and need to calculate the revenue and monthly payment.

We can define a function to calculate the loan revenue on the segmented data, and then use it to calculate the monthly installment for each. We also add a new column for the monthly installments to the segemented_data data frame. We use the numpy_financial.pmt function, equivalent to the Excel PMT function.



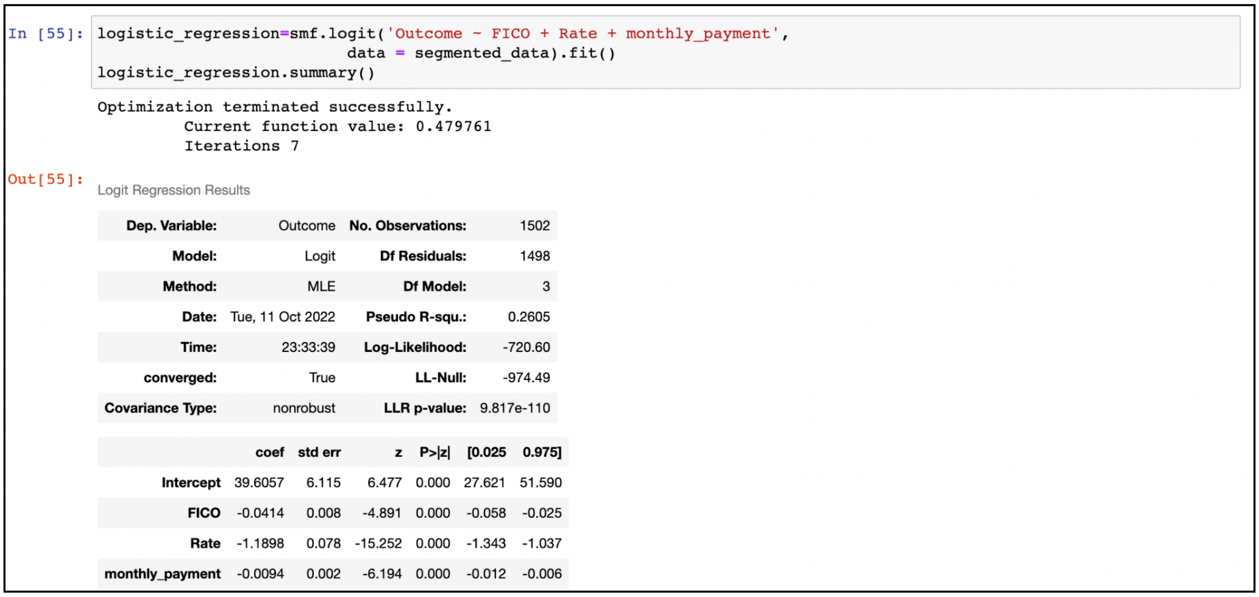
```
In [36]: def loan_rev(APR, cost_of_funds, term, amount):
             return -npf.pmt(APR/(100*12), term, amount)

In [37]: for i in segmented_data.index:
             rate = segmented_data.loc[i, 'Rate']
             amount = segmented_data.loc[i, 'Amount']
             cost_of_funds =segmented_data.loc[i, 'Cost of Funds']
             term = segmented_data.loc[i, 'Term']

             segmented_data.loc[i, 'monthly_payment'] = loan_rev(rate, cost_of_funds, term, amount)

In [39]: segmented_data
```

Out[39]:

| | Tier | FICO | Approve Date | Term | Amount | Previous Rate | Car Type | Competition rate | Outcome | Rate | Cost of Funds | Partner Bin | monthly_payment |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **358** | 2 | 702 | 2002-07-02 | 60 | 22000.00 | | U | 5.85 | 0 | 6.19 | 1.8388 | 3 | 427.267992 |
| **466** | 2 | 710 | 2002-07-03 | 60 | 21000.00 | | U | 5.85 | 0 | 6.19 | 1.8388 | 1 | 407.846719 |
| **849** | 3 | 693 | 2002-07-06 | 60 | 19597.95 | | U | 5.85 | 1 | 7.29 | 1.8400 | 1 | 390.749940 |
| **924** | 3 | 696 | 2002-07-06 | 60 | 23071.00 | | U | 5.85 | 0 | 7.29 | 1.8400 | 3 | 459.996676 |
| **1104** | 3 | 697 | 2002-07-08 | 60 | 21577.70 | | U | 5.80 | 1 | 7.29 | 1.8400 | 2 | 430.222803 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **207467** | 2 | 703 | 2004-11-14 | 60 | 20000.00 | | U | 4.85 | 0 | 6.59 | 2.1010 | 3 | 392.166633 |
| **207540** | 3 | 689 | 2004-11-14 | 60 | 21999.99 | | U | 4.85 | 0 | 8.49 | 2.1010 | 2 | 451.257460 |
| **207741** | 2 | 702 | 2004-11-15 | 60 | 21000.00 | | U | 4.85 | 0 | 6.59 | 2.1100 | 1 | 411.774965 |
| **207839** | 2 | 700 | 2004-11-15 | 60 | 19999.99 | | U | 4.85 | 0 | 6.59 | 2.1100 | 1 | 392.166437 |
| **208052** | 3 | 696 | 2004-11-16 | 60 | 20000.00 | | U | 4.85 | 0 | 8.49 | 2.1270 | 3 | 410.234241 |

1502 rows × 13 columns

We can now run a logistic regression using the .logit() function from the statsmodels library.



```
In [55]: logistic_regression=smf.logit('Outcome ~ FICO + Rate + monthly_payment',
                                      data = segmented_data).fit()
         logistic_regression.summary()

         Optimization terminated successfully.
                  Current function value: 0.479761
                  Iterations 7
Out[55]:
```

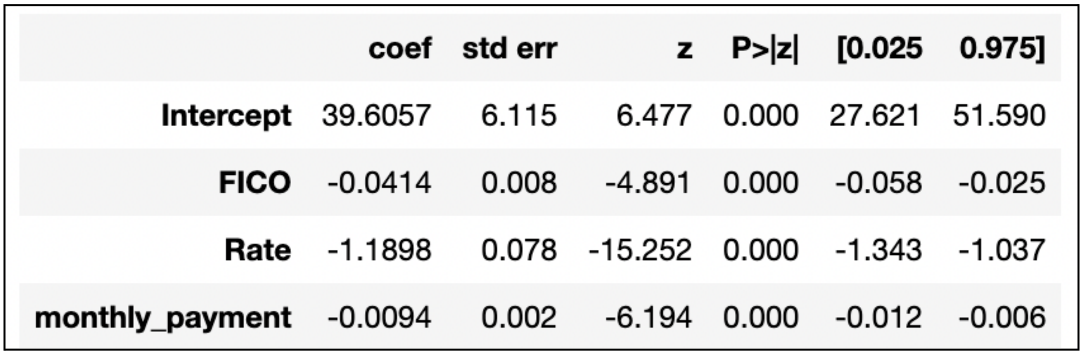
Logit Regression Results

| | | | |
|---|---|---|---|
| **Dep. Variable:** | Outcome | **No. Observations:** | 1502 |
| **Model:** | Logit | **Df Residuals:** | 1498 |
| **Method:** | MLE | **Df Model:** | 3 |
| **Date:** | Tue, 11 Oct 2022 | **Pseudo R-squ.:** | 0.2605 |
| **Time:** | 23:33:39 | **Log-Likelihood:** | -720.60 |
| **converged:** | True | **LL-Null:** | -974.49 |
| **Covariance Type:** | nonrobust | **LLR p-value:** | 9.817e-110 |

| | coef | std err | z | P>\|z\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| **Intercept** | 39.6057 | 6.115 | 6.477 | 0.000 | 27.621 | 51.590 |
| **FICO** | -0.0414 | 0.008 | -4.891 | 0.000 | -0.058 | -0.025 |
| **Rate** | -1.1898 | 0.078 | -15.252 | 0.000 | -1.343 | -1.037 |
| **monthly_payment** | -0.0094 | 0.002 | -6.194 | 0.000 | -0.012 | -0.006 |

**2.** For each of the three parameters, discuss if the coefficients found by the model are statistically significant and whether the relationship of their parameter to the outcome is positive or negative.

**Ans:**

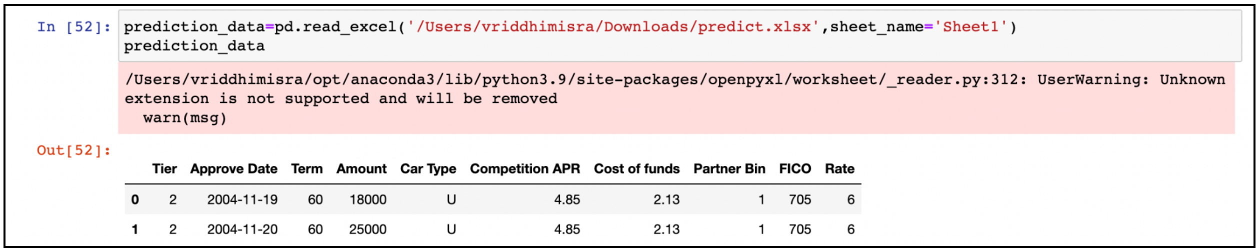


| | coef | std err | z | P>\|z\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| **Intercept** | 39.6057 | 6.115 | 6.477 | 0.000 | 27.621 | 51.590 |
| **FICO** | -0.0414 | 0.008 | -4.891 | 0.000 | -0.058 | -0.025 |
| **Rate** | -1.1898 | 0.078 | -15.252 | 0.000 | -1.343 | -1.037 |
| **monthly_payment** | -0.0094 | 0.002 | -6.194 | 0.000 | -0.012 | -0.006 |

We can see from the results of the Logistic regression that FICO, Rate, and monthly payment, all attributes have negative coefficients and thus, inversely affect the outcome, i.e chances of the loan being approved decrease as any of the parameters increase in value. Moreover, they are all statistically significant parameters as the p-values=0 for all three.

**3.** Use the logistic model you fit in Part 1 to predict if the new loans found in predict.xlsx will be accepted.

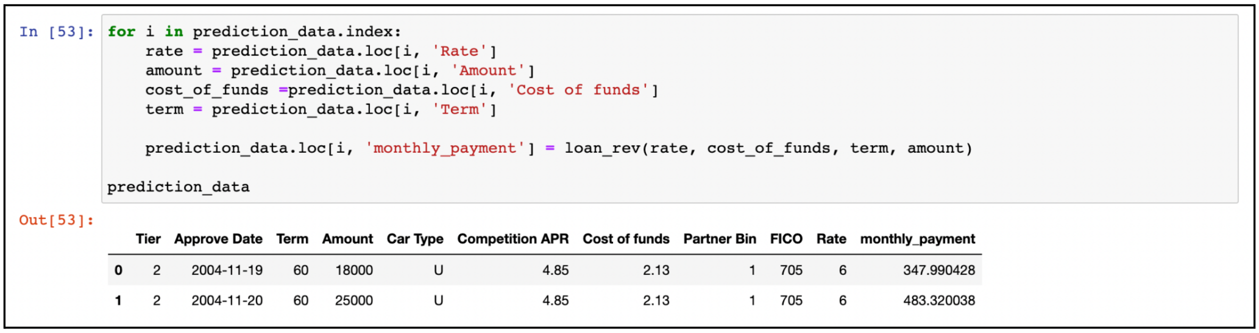
**Ans:** We import the 'predict.xlsx' file as a new data frame. We will use this data to predict if new loans will be accepted.



```
In [52]: prediction_data=pd.read_excel('/Users/vriddhimisra/Downloads/predict.xlsx',sheet_name='Sheet1')
         prediction_data

         /Users/vriddhimisra/opt/anaconda3/lib/python3.9/site-packages/openpyxl/worksheet/_reader.py:312: UserWarning: Unknown
         extension is not supported and will be removed
           warn(msg)

Out[52]:
```

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

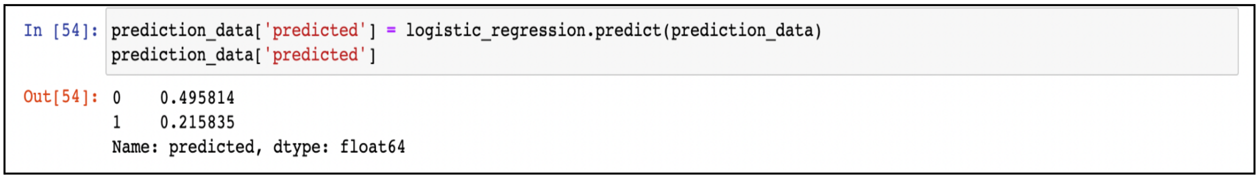
As done before, we will calculate the monthly installment using the loan_rev function but on the new(prediction) data imported.



```
In [53]: for i in prediction_data.index:
             rate = prediction_data.loc[i, 'Rate']
             amount = prediction_data.loc[i, 'Amount']
             cost_of_funds =prediction_data.loc[i, 'Cost of funds']
             term = prediction_data.loc[i, 'Term']

             prediction_data.loc[i, 'monthly_payment'] = loan_rev(rate, cost_of_funds, term, amount)

         prediction_data

Out[53]:
```

| | 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 |

Now, using the built-in .predict() function, we can predict based on the results of the logistic regression and the new prediction data.



```
In [54]: prediction_data['predicted'] = logistic_regression.predict(prediction_data)
         prediction_data['predicted']

Out[54]: 0    0.495814
         1    0.215835
         Name: predicted, dtype: float64
```

Thus, from the data provided for the two new loans, we can conclude that one of them has a probability of 0.49(49% chance of getting approved) and the other has a probability of 0.21(21% chance of getting approved).

## **Appendix:**

## **Python Notebook:**

```
import numpy as np
import pandas as pd
import statsmodels.formula.api as smf
import numpy_financial as npf
```

```
df= pd.read_excel('/Users/vriddhimisra/Downloads/Nomis_data.xlsx')
df
```

```
        Tier  FICO Approve Date  Term    Amount Previous Rate Car Type
\
0          3   695   2002-07-01    72  35000.00                      N

1          1   751   2002-07-01    60  40000.00                      N

2          1   731   2002-07-01    60  18064.00                      N

3          4   652   2002-07-01    72  15415.00                      N

4          1   730   2002-07-01    48  32000.00                      N

...      ...   ...          ...   ...       ...           ...      ...

208080     1   787   2004-11-16    60   5499.99           NaN        U

208081     3   791   2004-11-16    60  36500.00           NaN        N

208082     3   699   2004-11-16    36  19999.99           NaN        U

208083     2   708   2004-11-16    60  29999.99           NaN        U

208084     1   780   2004-11-16    60  34000.00           NaN        U


        Competition rate  Outcome  Rate  Cost of Funds  Partner Bin
0                   6.25        0  7.49         1.8388            1
1                   5.65        0  5.49         1.8388            3
2                   5.65        0  5.49         1.8388            3
3                   6.25        0  8.99         1.8388            3
4                   5.65        0  5.49         1.8388            1
...                  ...      ...   ...            ...          ...
208080              4.85        1  4.85         2.1270            1
208081              4.45        0  4.45         2.1270            3
208082              4.35        0  8.25         2.1270            2
208083              4.85        0  6.59         2.1270            2
208084              4.85        0  4.85         2.1270            1

[208085 rows x 12 columns]
```

```
segmented_data = df[(df['Car Type'] == 'U')
                    & (df['FICO'] >= 685)
                    & (df['FICO'] <= 712)
                    & (df['Term'] == 60)
                    & (df['Amount'] >= 17800)
                    & (df['Amount'] <= 25000)].copy()
segmented_data
```

```
        Tier  FICO Approve Date  Term    Amount Previous Rate Car Type
\
358        2   702   2002-07-02    60  22000.00                      U

466        2   710   2002-07-03    60  21000.00                      U

849        3   693   2002-07-06    60  19597.95                      U

924        3   696   2002-07-06    60  23071.00                      U

1104       3   697   2002-07-08    60  21577.70                      U

...      ...   ...          ...   ...       ...           ...      ...

207467     2   703   2004-11-14    60  20000.00                      U

207540     3   689   2004-11-14    60  21999.99                      U

207741     2   702   2004-11-15    60  21000.00                      U

207839     2   700   2004-11-15    60  19999.99                      U

208052     3   696   2004-11-16    60  20000.00                      U


        Competition rate  Outcome  Rate  Cost of Funds  Partner Bin
358                 5.85        0  6.19         1.8388            3
466                 5.85        0  6.19         1.8388            1
849                 5.85        1  7.29         1.8400            1
924                 5.85        0  7.29         1.8400            3
1104                5.80        1  7.29         1.8400            2
...                  ...      ...   ...            ...          ...
207467              4.85        0  6.59         2.1010            3
207540              4.85        0  8.49         2.1010            2
207741              4.85        0  6.59         2.1100            1
207839              4.85        0  6.59         2.1100            1
208052              4.85        0  8.49         2.1270            3

[1502 rows x 12 columns]
```

```
len(segmented_data)
```

```
1502
```

```python
def loan_rev(APR, cost_of_funds, term, amount):
    return -npf.pmt(APR/(100*12), term, amount)

for i in segmented_data.index:
    rate = segmented_data.loc[i, 'Rate']
    amount = segmented_data.loc[i, 'Amount']
    cost_of_funds =segmented_data.loc[i, 'Cost of Funds']
    term = segmented_data.loc[i, 'Term']

    segmented_data.loc[i, 'monthly_payment'] = loan_rev(rate, 
cost_of_funds, term, amount)

segmented_data
```

```
        Tier  FICO Approve Date  Term    Amount Previous Rate Car Type 
\
358        2   702   2002-07-02    60  22000.00                      U   

466        2   710   2002-07-03    60  21000.00                      U   

849        3   693   2002-07-06    60  19597.95                      U   

924        3   696   2002-07-06    60  23071.00                      U   

1104       3   697   2002-07-08    60  21577.70                      U   

...      ...   ...          ...   ...       ...           ...      ...   

207467     2   703   2004-11-14    60  20000.00                      U   

207540     3   689   2004-11-14    60  21999.99                      U   

207741     2   702   2004-11-15    60  21000.00                      U   

207839     2   700   2004-11-15    60  19999.99                      U   

208052     3   696   2004-11-16    60  20000.00                      U   


        Competition rate  Outcome  Rate  Cost of Funds  Partner Bin  \
358                 5.85        0  6.19         1.8388            3   
466                 5.85        0  6.19         1.8388            1   
849                 5.85        1  7.29         1.8400            1   
924                 5.85        0  7.29         1.8400            3   
1104                5.80        1  7.29         1.8400            2   
...                  ...      ...   ...            ...          ...   
207467              4.85        0  6.59         2.1010            3   
207540              4.85        0  8.49         2.1010            2   
207741              4.85        0  6.59         2.1100            1   
207839              4.85        0  6.59         2.1100            1   
```

```
208052              4.85        0  8.49         2.1270            3

        monthly_payment
358          427.267992
466          407.846719
849          390.749940
924          459.996676
1104         430.222803
...                 ...
207467       392.166633
207540       451.257460
207741       411.774965
207839       392.166437
208052       410.234241

[1502 rows x 13 columns]
```

```
logistic_regression=smf.logit('Outcome ~ FICO + Rate +
monthly_payment',
                          data = segmented_data).fit()
logistic_regression.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.479761
         Iterations 7

<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results

======================================================================
========
Dep. Variable:                Outcome   No. Observations:
1502
Model:                          Logit   Df Residuals:
1498
Method:                           MLE   Df Model:
3
Date:                Tue, 11 Oct 2022   Pseudo R-squ.:
0.2605
Time:                        23:33:39   Log-Likelihood:
-720.60
converged:                       True   LL-Null:
-974.49
Covariance Type:            nonrobust   LLR p-value:
9.817e-110
======================================================================
=============
                      coef    std err          z      P>|z|
[0.025      0.975]
```

```
------------------------------------------------------------------------
-------------
Intercept          39.6057      6.115      6.477      0.000
27.621      51.590
FICO               -0.0414      0.008     -4.891      0.000      -
0.058      -0.025
Rate               -1.1898      0.078    -15.252      0.000      -
1.343      -1.037
monthly_payment    -0.0094      0.002     -6.194      0.000      -
0.012      -0.006
========================================================================
=============
"""
```

```
prediction_data=pd.read_excel('/Users/vriddhimisra/Downloads/
predict.xlsx',sheet_name='Sheet1')
prediction_data
```

```
/Users/vriddhimisra/opt/anaconda3/lib/python3.9/site-packages/
openpyxl/worksheet/_reader.py:312: UserWarning: Unknown extension is
not supported and will be removed
  warn(msg)
```

```
   Tier Approve Date  Term  Amount Car Type  Competition APR  Cost of
funds  \
0     2   2004-11-19    60   18000        U             4.85
2.13
1     2   2004-11-20    60   25000        U             4.85
2.13

   Partner Bin  FICO  Rate
0            1   705     6
1            1   705     6
```

```
for i in prediction_data.index:
    rate = prediction_data.loc[i, 'Rate']
    amount = prediction_data.loc[i, 'Amount']
    cost_of_funds =prediction_data.loc[i, 'Cost of funds']
    term = prediction_data.loc[i, 'Term']

    prediction_data.loc[i, 'monthly_payment'] = loan_rev(rate,
cost_of_funds, term, amount)

prediction_data
```

```
   Tier Approve Date  Term  Amount Car Type  Competition APR  Cost of
funds  \
0     2   2004-11-19    60   18000        U             4.85
2.13
1     2   2004-11-20    60   25000        U             4.85
2.13
```

```
   Partner Bin  FICO  Rate  monthly_payment
0            1   705     6       347.990428
1            1   705     6       483.320038
```

```
prediction_data['predicted'] =
logistic_regression.predict(prediction_data)
prediction_data['predicted']
```

```
0    0.495814
1    0.215835
Name: predicted, dtype: float64
```