Homework 6 - Grant Jackson

October 16, 2024

0.0.1 HW6: Calculate the marginal effect of the student status on default probability, holding income and balance at their means, using the formula on page 8 of the lecture note 6

- Note that the student variable is binary

[1]:
```python
import os
os.chdir('C:\\Users\gmoor\Documents\Economic Analytics 1\Data')

import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt

raw0 = pd.read_csv('Default.csv')

# drop the observations that contain missing values
raw0.dropna()

raw0.head()
```

[1]:

	Unnamed: 0	default	student	balance	income
0	1	No	No	729.526495	44361.625074
1	2	No	Yes	817.180407	12106.134700
2	3	No	No	1073.549164	31767.138947
3	4	No	No	529.250605	35704.493935
4	5	No	No	785.655883	38463.495879

[2]:
```python
raw0.describe(include = 'all')
```

[2]:

	Unnamed: 0	default	student	balance	income
count	10000.00000	10000	10000	10000.000000	10000.000000
unique	NaN	2	2	NaN	NaN
top	NaN	No	No	NaN	NaN
freq	NaN	9667	7056	NaN	NaN
mean	5000.50000	NaN	NaN	835.374886	33516.981876
std	2886.89568	NaN	NaN	483.714985	13336.639563
min	1.00000	NaN	NaN	0.000000	771.967729
25%	2500.75000	NaN	NaN	481.731105	21340.462903

```
50%       5000.50000      NaN      NaN     823.636973  34552.644802
75%       7500.25000      NaN      NaN    1166.308386  43807.729272
max      10000.00000      NaN      NaN    2654.322576  73554.233495
```

[3]:
```python
raw0.default=(raw0.default=='Yes')*1
raw0.student=(raw0.student=='Yes')*1

raw0.head()
```

[3]:
```
   Unnamed: 0  default  student      balance        income
0           1        0        0   729.526495  44361.625074
1           2        0        1   817.180407  12106.134700
2           3        0        0  1073.549164  31767.138947
3           4        0        0   529.250605  35704.493935
4           5        0        0   785.655883  38463.495879
```

[4]:
```python
# Run a logistic regression
import statsmodels.api as sm # Regular api -> Logit(Y,X)
import statsmodels.formula.api as smf # Formula api -> logit(default ~ student␣
 ↪+... ) (lower-case l)
# SKlearn -> LogitRegression(X,Y)

Y = raw0.default
X = raw0.iloc[:,2:]
X = sm.add_constant(X)
```

[5]:
```python
X
```

[5]:
```
      const  student      balance        income
0       1.0        0   729.526495  44361.625074
1       1.0        1   817.180407  12106.134700
2       1.0        0  1073.549164  31767.138947
3       1.0        0   529.250605  35704.493935
4       1.0        0   785.655883  38463.495879
…       …      …           …             …
9995    1.0        0   711.555020  52992.378914
9996    1.0        0   757.962918  19660.721768
9997    1.0        0   845.411989  58636.156984
9998    1.0        0  1569.009053  36669.112365
9999    1.0        1   200.922183  16862.952321

[10000 rows x 4 columns]
```

[6]:
```python
Y
```

[6]:
```
0       0
1       0
```

```
2       0
3       0
4       0
       ..
9995    0
9996    0
9997    0
9998    0
9999    0
Name: default, Length: 10000, dtype: int32
```

[7]:
```python
logitres=sm.Logit(Y,X).fit() # Include Y first; case sensitive: Logit (o)␣
↪logit(x)

print(logitres.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.078577
         Iterations 10
                           Logit Regression Results
==============================================================================
Dep. Variable:                default   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9996
Method:                           MLE   Df Model:                            3
Date:                Wed, 16 Oct 2024   Pseudo R-squ.:                  0.4619
Time:                        21:51:10   Log-Likelihood:                -785.77
converged:                       True   LL-Null:                       -1460.3
Covariance Type:            nonrobust   LLR p-value:                3.257e-292
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const        -10.8690      0.492    -22.079      0.000     -11.834      -9.904
student       -0.6468      0.236     -2.738      0.006      -1.110      -0.184
balance        0.0057      0.000     24.737      0.000       0.005       0.006
income      3.033e-06    8.2e-06      0.370      0.712    -1.3e-05    1.91e-05
==============================================================================

Possibly complete quasi-separation: A fraction 0.15 of observations can be
perfectly predicted. This might indicate that there is complete
quasi-separation. In this case some parameters will not be identified.
```

[8]:
```python
# Extract coefficients from the initial logistic model
beta_0 = logitres.params.iloc[0] # Intercept
beta_student = logitres.params['student']
beta_balance = logitres.params['balance']
beta_income = logitres.params['income']
```

```python
print('\nInterecept:', beta_0)
print('\nStudent coefficent:', beta_student)
print('\nBalance coefficent: ', beta_balance)
print('\nIncome coefficent:', beta_income)
```

```
Interecept: -10.869045212744663

Student coefficent: -0.646775808244028

Balance coefficent:  0.005736505265799081

Income coefficent: 3.0334501193335614e-06
```

[9]:
```python
# Define the logistic function
def logistic_function(x):
    return 1 / (1 + np.exp(-x))

# Means of balance and income
mean_balance = raw0['balance'].mean()
mean_income = raw0['income'].mean()

# Predicted probabilities when student = 1
P_student_1 = logistic_function(beta_0 + beta_student * 1 + beta_balance *␣
↪mean_balance + beta_income * mean_income)

# Predicted probabilties when student = 0
P_student_0 = logistic_function(beta_0 + beta_student * 0 + beta_balance *␣
↪mean_balance + beta_income * mean_income)

# Calculate marginal effect
marginal_effect = P_student_1 - P_student_0

print(f"Marginal effect of being a student on default probabilities is:␣
↪{marginal_effect:.6f}")
```

```
Marginal effect of being a student on default probabilities is: -0.001205
```

[10]:
```python
# Using the predict function
marginal_effect = logitres.predict([1,1,mean_balance,mean_income]) - logitres.
↪predict([1,0,mean_balance,mean_income])

print(f"Marginal effect of being a student on default probabillities is:␣
↪{marginal_effect}")
```

```
Marginal effect of being a student on default probabillities is: [-0.00120547]
```

[ ]: