BookBinders Targeting Using Logistic Regression

Professor Song Yao
Olin Business School

Customer Analytics

We use the BookBinders Book Club as an example of how to model the response to an offer with a logit model

LOGISTIC REGRESSION AT BOOKBINDERS

- Stan Lawton (marketing director) pulls a random sample of 50,000 customers from the BookBinders database
- Stan mails "The Art History of Florence" to the entire sample
- 4522 customers buy the book
- Plans to use the model to determine which customer to target from the entire database (500,000 remaining customers, excluding test group)
- Stan has information on R, F, M, and book purchases in each category from past, gender, zip code, etc.

How do different factors affect the probability of purchasing the promoted book?

RESULTS FROM LOGISTIC REGRESSION

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                  buyer   No. Observations:                50000
Model:                          Logit   Df Residuals:                    49989
Method:                           MLE   Df Model:                           10
Date:                Tue, 07 Jan 2025   Pseudo R-squ.:                  0.2053
Time:                        17:54:32   Log-Likelihood:                -12061.
converged:                       True   LL-Null:                       -15178.
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         -1.6001      0.052    -30.713      0.000      -1.702      -1.498
last          -0.0947      0.003    -33.918      0.000      -0.100      -0.089
total          0.0011      0.000      5.630      0.000       0.001       0.002
female        -0.7607      0.036    -21.272      0.000      -0.831      -0.691
child         -0.1862      0.017    -10.775      0.000      -0.220      -0.152
youth         -0.1130      0.026     -4.327      0.000      -0.164      -0.062
cook          -0.2703      0.017    -15.782      0.000      -0.304      -0.237
do_it         -0.5392      0.027    -19.994      0.000      -0.592      -0.486
refernce       0.2347      0.027      8.837      0.000       0.183       0.287
art            1.1556      0.022     52.185      0.000       1.112       1.199
geog           0.5743      0.019     30.823      0.000       0.538       0.611
==============================================================================
```

- Difficult to compare impacts because covariates have different scales.

Scaling of different variables

Standard Deviation of the Variables in the Data

```python
# Calculate and report standard deviations of all independent variables, i.e., features
# defined above
std_devs = BBB[features].std()

print("Standard Deviations of Independent Variables:")
print(std_devs)
```

```
Standard Deviations of Independent Variables:
last          8.153091
total       101.357259
female        0.471630
child         1.120153
youth         0.682996
cook          1.185432
do_it         0.765877
refernce      0.603882
art           0.680261
geog          0.841052
dtype: float64
```

The variable "total" has a much larger SD.

Scale before or after the regression

```
        Logit Marginal Effects
=====================================
Dep. Variable:                  buyer
Method:                          dydx
At:                           overall
==============================================================================
                dy/dx    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
last          -0.0064      0.000    -33.584      0.000      -0.007      -0.006
total       7.555e-05   1.34e-05      5.630      0.000    4.92e-05       0.000
female        -0.0515      0.002    -21.218      0.000      -0.056      -0.047
child         -0.0126      0.001    -10.775      0.000      -0.015      -0.010
youth         -0.0076      0.002     -4.327      0.000      -0.011      -0.004
cook          -0.0183      0.001    -15.785      0.000      -0.021      -0.016
do_it         -0.0365      0.002    -19.987      0.000      -0.040      -0.033
refernce       0.0159      0.002      8.839      0.000       0.012       0.019
art            0.0782      0.001     53.462      0.000       0.075       0.081
geog           0.0389      0.001     30.972      0.000       0.036       0.041
==============================================================================
```

AME of total * SD = 7.555e-05 * 101.36 = 0.0077

Scale before the regression

```python
# Scale the 'total' variable using z-score
BBB['total_scaled'] = (BBB['total'] - BBB['total'].mean()) / BBB['total'].std()

# Update the features list with the scaled 'total' variable
features = ['last', 'total_scaled', 'female', 'child', 'youth', 'cook', 'do_it', 'refernce', 'art', 'geog']

# Prepare the features (X) and target variable (y)
X = BBB[features]
y = BBB['buyer']

# Add a constant term to the features (required for statsmodels)
X = sm.add_constant(X)

# Fit the logistic regression model with scaled 'total'
logit_model_scaled = sm.Logit(y, X)
results_scaled = logit_model_scaled.fit()
```

```python
# Calculate Average Marginal Effects (AME)
ame = results_scaled.get_margeff(at='overall', method='dydx')
```

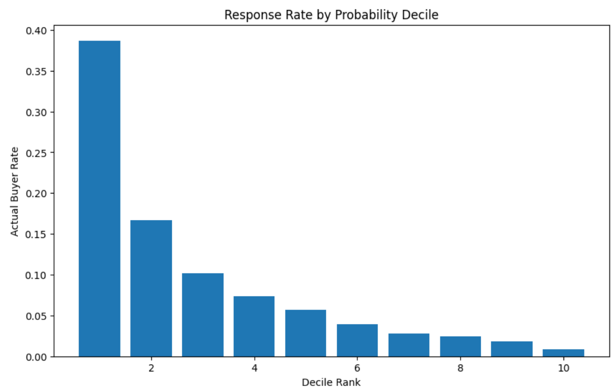
```
================================================================================
                   dy/dx    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
last             -0.0064      0.000    -33.584      0.000      -0.007      -0.006
total_scaled      0.0077      0.001      5.630      0.000       0.005       0.010
female           -0.0515      0.002    -21.218      0.000      -0.056      -0.047
child            -0.0126      0.001    -10.775      0.000      -0.015      -0.010
youth            -0.0076      0.002     -4.327      0.000      -0.011      -0.004
cook             -0.0183      0.001    -15.785      0.000      -0.021      -0.016
do_it            -0.0365      0.002    -19.987      0.000      -0.040      -0.033
refernce          0.0159      0.002      8.839      0.000       0.012       0.019
art               0.0782      0.001     53.462      0.000       0.075       0.081
geog              0.0389      0.001     30.972      0.000       0.036       0.041
================================================================================
```

We can generate individual purchase probabilities and then view the response rates by decile of predicted purchase probabilities

DECILE ANALYSIS OF PURCHASE PROBABILITIES

Prediction vs. Actual Buyer Rate by Decile:

Decile_Rank	Avg_Predicted_Prob	Actual_Buyer_Rate
1	0.3856	0.3870
2	0.1638	0.1672
3	0.1049	0.1022
4	0.0741	0.0736
5	0.0556	0.0568
6	0.0423	0.0392
7	0.0321	0.0278
8	0.0237	0.0242
9	0.0157	0.0180
10	0.0065	0.0084



Number of customers, buyers, and response rate by decile

DECILE ANALYSIS OF PURCHASE PROBABILITIES

pred_prob_logit_decile	Customer_Count	Buyer_Count	Resp_Rate
1	5000	1935	0.3870000
2	5000	836	0.1672000
3	5000	511	0.1022000
4	5000	368	0.0736000
5	5000	284	0.0568000
6	5000	196	0.0392000
7	4998	139	0.0278111
8	5002	121	0.0241903
9	5000	90	0.0180000
10	5000	42	0.0084000

Why do the two regressions have such different estimates?

Original Model

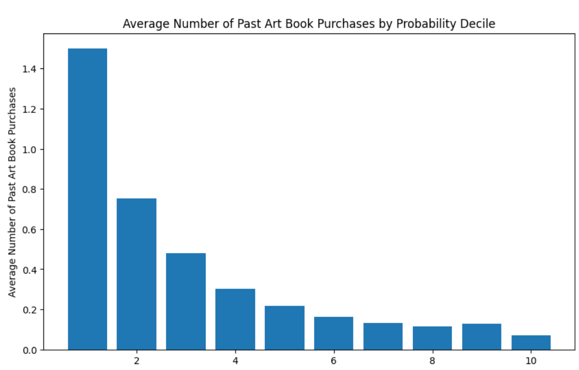
	coef	std err	z	P>|z|	[0.025	0.975]
const	-1.3676	0.043	-31.457	0.000	-1.453	-1.282
last	-0.0947	0.003	-33.918	0.000	-0.100	-0.089
total_scaled	0.1131	0.020	5.630	0.000	0.074	0.152
female	-0.7607	0.036	-21.272	0.000	-0.831	-0.691
child	-0.1862	0.017	-10.775	0.000	-0.220	-0.152
youth	-0.1130	0.026	-4.327	0.000	-0.164	-0.062
cook	-0.2703	0.017	-15.782	0.000	-0.304	-0.237
do_it	-0.5392	0.027	-19.994	0.000	-0.592	-0.486
refernce	0.2347	0.027	8.837	0.000	0.183	0.287
art	1.1556	0.022	52.185	0.000	1.112	1.199
geog	0.5743	0.019	30.823	0.000	0.538	0.611

Child only Model

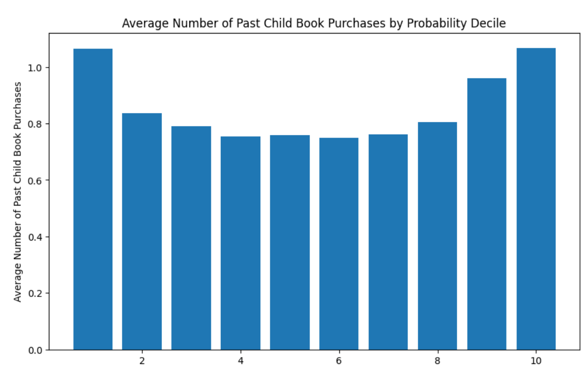
	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.3744	0.020	-119.412	0.000	-2.413	-2.335
child	0.0741	0.013	5.606	0.000	0.048	0.100

Why do the two regressions have such different estimates?

DECILE ANALYSIS OF INDEPENDENT VARIABLES

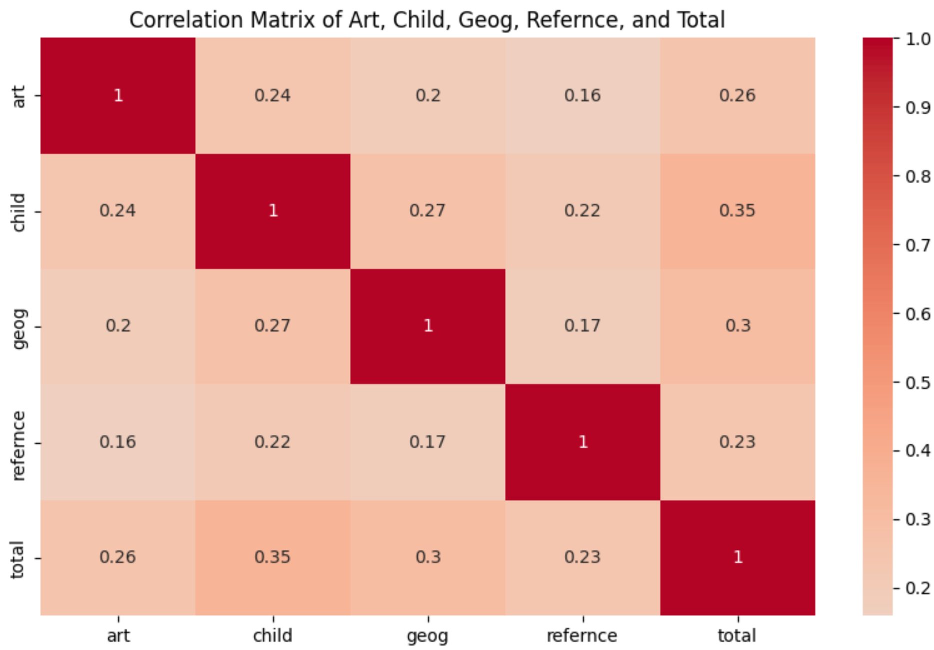


Art's Coefficient is 1.156 in the Original Model

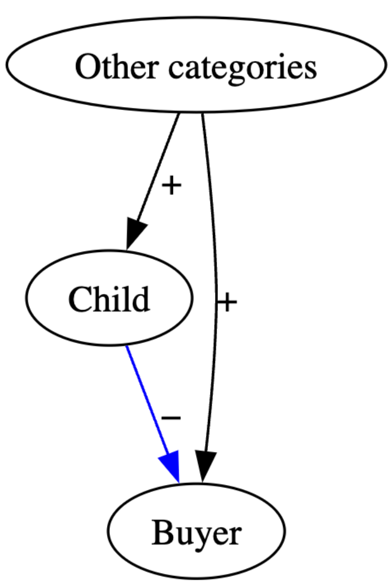


Child's Coefficient is –0.186 in the Original Model

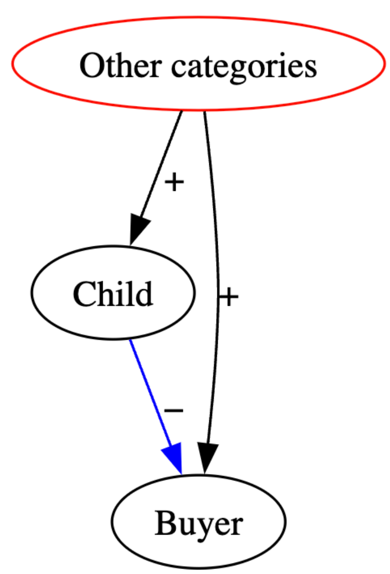
The correlations of "Child" with other "positive" variables are all significant—what does it imply?



DAG Analysis



What type of bias is this?
And how to mitigate?



Why do the two regressions have such different estimates?

Original Model

	coef	std err	z	P>|z|	[0.025	0.975]
const	-1.3676	0.043	-31.457	0.000	-1.453	-1.282
last	-0.0947	0.003	-33.918	0.000	-0.100	-0.089
total_scaled	0.1131	0.020	5.630	0.000	0.074	0.152
female	-0.7607	0.036	-21.272	0.000	-0.831	-0.691
child	-0.1862	0.017	-10.775	0.000	-0.220	-0.152
youth	-0.1130	0.026	-4.327	0.000	-0.164	-0.062
cook	-0.2703	0.017	-15.782	0.000	-0.304	-0.237
do_it	-0.5392	0.027	-19.994	0.000	-0.592	-0.486
refernce	0.2347	0.027	8.837	0.000	0.183	0.287
art	1.1556	0.022	52.185	0.000	1.112	1.199
geog	0.5743	0.019	30.823	0.000	0.538	0.611

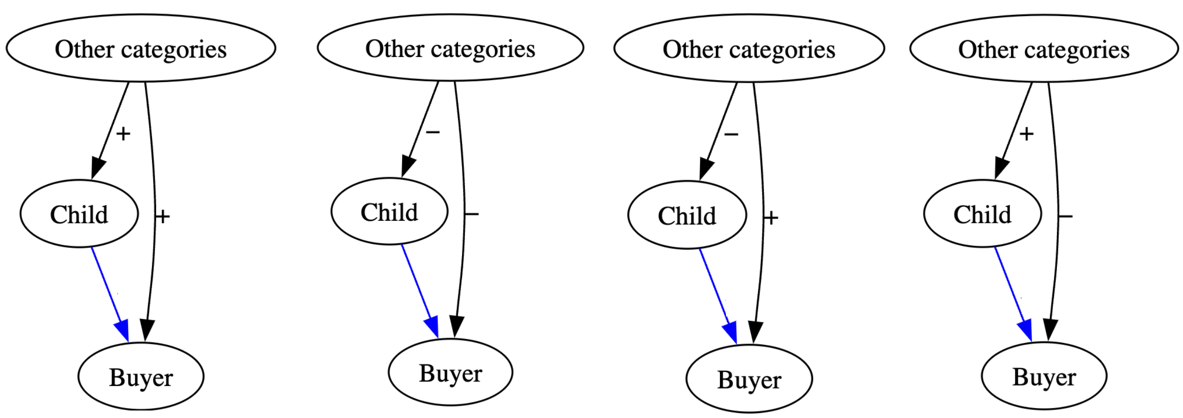
Child only Model

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.3744	0.020	-119.412	0.000	-2.413	-2.335
child	0.0741	0.013	5.606	0.000	0.048	0.100

Child and Art Model

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.8095	0.023	-119.575	0.000	-2.856	-2.763
child	-0.1351	0.015	-8.929	0.000	-0.165	-0.105
art	1.0407	0.019	54.289	0.000	1.003	1.078

Direction of the confounding bias: Over or under-estimate?



Two positives:
effect upward bias
(overestimate)

Two negatives:
effect upward bias
(overestimate)

One positive and
one negative?

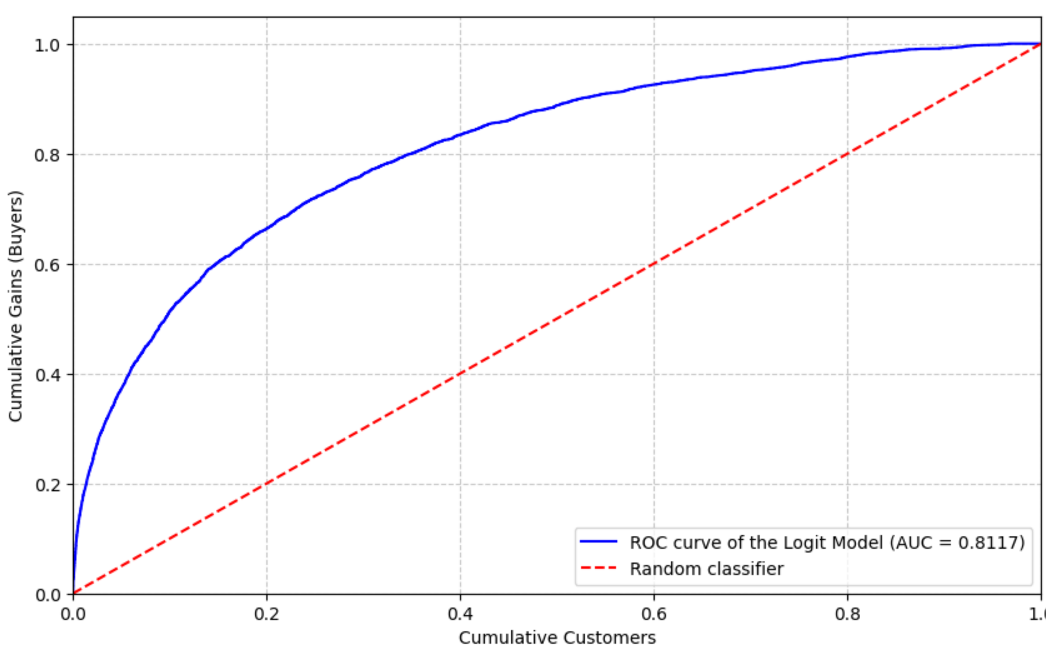
Gains Table and Chart (I wrote a function—see demo code on Canvas)

```
gain_table = create_gains_table(BBB['buyer'], BBB['predicted_prob'])
print(gain_table)
```

```
   Bucket  Customers  Cum_Customers  Buyers  Cum_Buyers  Resp_Rate   Gains  \
0       1       5000           5000    1935        1935     0.3870  0.4279
1       2       5000          10000     836        2771     0.1672  0.1849
2       3       5000          15000     511        3282     0.1022  0.1130
3       4       5000          20000     368        3650     0.0736  0.0814
4       5       5000          25000     284        3934     0.0568  0.0628
5       6       5000          30000     196        4130     0.0392  0.0433
6       7       4998          34998     139        4269     0.0278  0.0307
7       8       5002          40000     121        4390     0.0242  0.0268
8       9       5000          45000      90        4480     0.0180  0.0199
9      10       5000          50000      42        4522     0.0084  0.0093

   Cum_Gains    Lift  Cum_Lift
0     0.4279  4.2791    4.2791
1     0.6128  1.8487    3.0639
2     0.7258  1.1300    2.4193
3     0.8072  0.8138    2.0179
4     0.8700  0.6280    1.7399
5     0.9133  0.4334    1.5222
6     0.9441  0.3075    1.3487
7     0.9708  0.2675    1.2135
8     0.9907  0.1990    1.1008
9     1.0000  0.0929    1.0000
```


Gains Table and Chart



The break-even response rate tells us to which cells to extend the offer

BREAK EVEN RESPONSE RATE

- Cost of mailing an offer = $0.50
- Selling price (includes shipping) = $18
- Wholesale price paid by Bookbinders = $9
- Shipping costs = $3
- Break-even = Cost to mail/net revenue per sale = .5/(18-9-3)= **8.3%**

Using the logit model, we target fewer customers but with a higher response rate

PROFITABILITY (LOGIT MODEL)

```
break_even = 0.5 / 6
print(f"The breakeven response rate is {break_even:.4f}")

# Generate a "targeted" variable based on the breakeven response rate
BBB['targeted'] = (BBB['predicted_prob'] >= break_even).astype(int)

# Calculate and print the number of customers targeted
num_targeted = BBB['targeted'].sum()
print(f"\nNumber of customers targeted: {num_targeted}")

# Calculate and print the percentage of customers targeted
percent_targeted = (num_targeted / len(BBB)) * 100
print(f"Percentage of customers targeted: {percent_targeted:.2f}%")

# Calculate the number of targeted customers ending up buying
num_targeted_buyers = BBB[BBB['targeted'] == 1]['buyer'].sum()
print(f"Number of buyers targeted: {num_targeted_buyers}")

# Calculate the percentage of buyers among the targeted customers
percent_targeted_buyers = (num_targeted_buyers / num_targeted) * 100
print(f"Percentage of buyers targeted: {percent_targeted_buyers:.2f}%")
```

```
Number of customers targeted: 15560
Percentage of customers targeted: 31.12%
Number of buyers targeted: 3323
Percentage of buyers targeted: 21.36%
```

Mail to 31.1% of sample:
500,000 * 31.1%=155,500

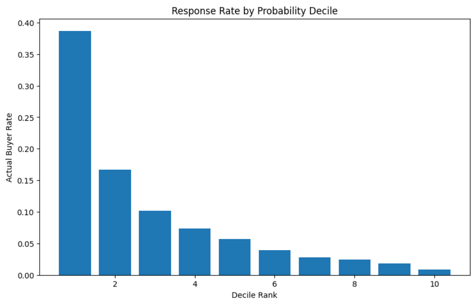
Average response rate: 21.4%
Expected number of buyers:
21.4% * 155,500 = **33,277**

- Gross profit = ($18 -9 -3)* 33,277 - 0.5*155,500
  = **$ 121,912**
- Return on marketing expenditure
  = $121,912/ ($0.5*155,500) = **156.8%**

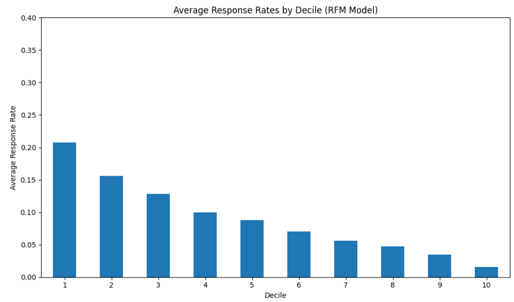
We can start comparing the logit model with the RFM analysis by comparing predicted response rates by decile

COMPARISON OF PREDICTED RESPONSE RATES

Analysis with logistic regression



Analysis with RFM



Analysis with logistic regression using only RFM variables