Lecture 11: Logistic regression

Madeleine Udell and Josh Grossman
Stanford University

Linear regression

Real-valued outcomes modeled as a linear combination of covariates.

$$y_i = \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i \qquad \epsilon_i \sim N(0, \sigma^2)$$

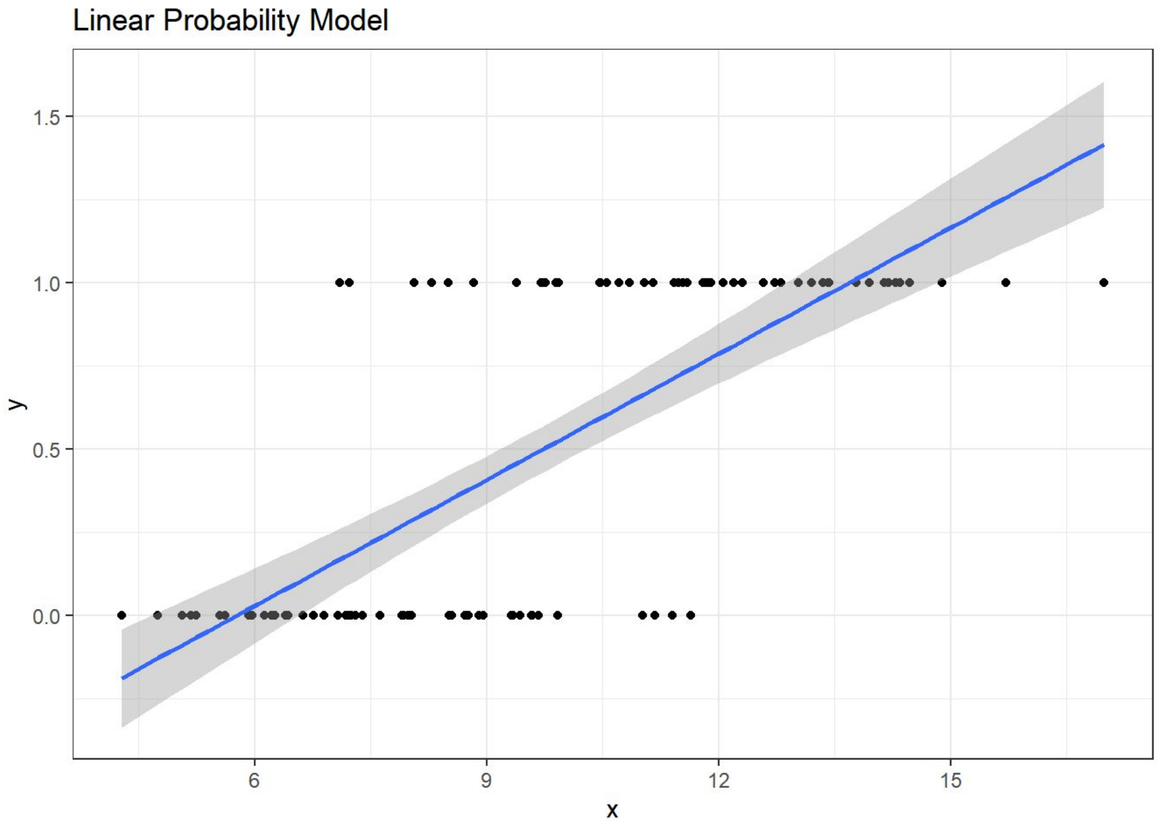
0-1 outcomes

What if outcomes are binary?

$$Y_i \in \{\text{rain}, \text{no rain}\}$$

$$Y_i \in \{\text{democrat}, \text{republican}\}$$

$$Y_i \in \{\text{spam}, \text{not spam}\}$$


Linear Probability Model
1.5
1.0
0.5
0.0
y
6
9
12
15
x

Linear regression for binary outcomes

You can do it, but the prediction is not guaranteed to be in the interval [0,1]. [Linear probability model.]

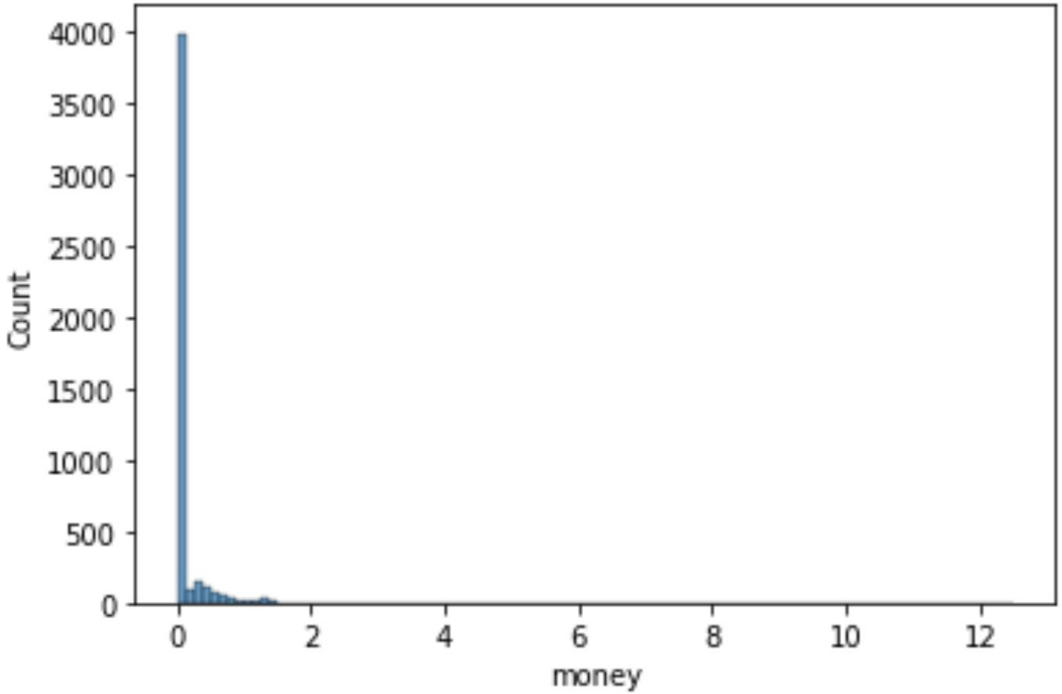
$$y_i = \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

```python
print(spam.columns)
```

```
Index(['make', 'address', 'all', 'num3d', 'our', 'over', 'remove', 'internet',
       'order', 'mail', 'receive', 'will', 'people', 'report', 'addresses',
       'free', 'business', 'email', 'you', 'credit', 'your', 'font', 'num000',
       'money', 'hp', 'hpl', 'george', 'num650', 'lab', 'labs', 'telnet',
       'num857', 'data', 'num415', 'num85', 'technology', 'num1999', 'parts',
       'pm', 'direct', 'cs', 'meeting', 'original', 'project', 're', 'edu',
       'table', 'conference', 'char_semicolon', 'char_left_paren',
       'char_left_bracket', 'char_exclamation', 'char_dollar', 'char_pound',
       'capital_avg', 'capital_long', 'capital_total', 'is_spam'],
      dtype='object')
```

```python
# 1: email is spam
# 0: email is not spam
spam['is_spam'].value_counts()
```

```
0    2788
1    1813
Name: is_spam, dtype: int64
```


4000
3500
3000
2500
2000
1500
1000
500
0
Count
0
2
4
6
8
10
12
money

```python
# 0% to 12.5% of the words in each of the 4601 emails is "money"
print(spam['money'].head())
spam['money'].describe()
```

```
0    0.00
1    0.43
2    0.06
3    0.00
4    0.00
Name: money, dtype: float64
count    4601.000000
mean        0.094269
std         0.442636
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max        12.500000
Name: money, dtype: float64
```

```python
# 1. char_dollar: % of characters in the email that are '$'
# 2. credit: % of words in the email that are 'credit'
# 3. money: % of words in the email that are 'money'
# 4. re: % of words in the email that are 're' (as in the subject line 're: hello')
formula = 'is_spam ~ 1 + char_dollar + credit + money + re'
model = smf.ols(formula=formula, data=spam).fit()
model.summary()
```

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.3346	0.007	45.696	0.000	0.320	0.349
char_dollar	0.5855	0.027	21.889	0.000	0.533	0.638
credit	0.1575	0.013	12.255	0.000	0.132	0.183
money	0.1879	0.015	12.635	0.000	0.159	0.217
re	-0.0536	0.006	-8.271	0.000	-0.066	-0.041

How do you interpret the intercept?

How do you interpret the `char_dollar` coefficient?
[Discuss with neighbors] [Poll]

How do you interpret the intercept?

A) 0.3346% of sampled emails are spam
B) 33.46% of sampled emails are spam
C) 33.46% of blank sampled emails are spam
D) 33.46% of sampled emails without "$", "credit", "money", or "re" are spam

How do you interpret the char_dollar coefficient?

A) 58.55% of sampled emails containing "$" are spam
B) A 1% increase in the proportion of characters in an email that are "$" is associated with a 58.55pp increase in the probability that the email is spam
C) One additional "$" character in a sampled email is associated with a 58.55% increase in the probability that the email is spam
D) A 100pp increase in the proportion of characters in an email that are "$" is associated with a 58.55pp increase in the probability that the email is spam

```python
# 1. char_dollar: % of characters in the email that are '$'
# 2. credit: % of words in the email that are 'credit'
# 3. money: % of words in the email that are 'money'
# 4. re: % of words in the email that are 're' (as in the subject line 're: hello')
formula = 'is_spam ~ 1 + char_dollar + credit + money + re'
model = smf.ols(formula=formula, data=spam).fit()
model.summary()
```

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.3346	0.007	45.696	0.000	0.320	0.349
char_dollar	0.5855	0.027	21.889	0.000	0.533	0.638
credit	0.1575	0.013	12.255	0.000	0.132	0.183
money	0.1879	0.015	12.635	0.000	0.159	0.217
re	-0.0536	0.006	-8.271	0.000	-0.066	-0.041

If all the covariates are zero, we estimate a 33% probability that the email is spam.

```python
# 1. char_dollar: % of characters in the email that are '$'
# 2. credit: % of words in the email that are 'credit'
# 3. money: % of words in the email that are 'money'
# 4. re: % of words in the email that are 're' (as in the subject line 're: hello')
formula = 'is_spam ~ 1 + char_dollar + credit + money + re'
model = smf.ols(formula=formula, data=spam).fit()
model.summary()
```

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.3346	0.007	45.696	0.000	0.320	0.349
char_dollar	0.5855	0.027	21.889	0.000	0.533	0.638
credit	0.1575	0.013	12.255	0.000	0.132	0.183
money	0.1879	0.015	12.635	0.000	0.159	0.217
re	-0.0536	0.006	-8.271	0.000	-0.066	-0.041

For every one unit increase in `char_dollar`, we estimate a 0.59 increase in `is_spam` (i.e., a 59 percentage point increase in the probability that the email is spam)

Note: a 59pp increase is different than a 59% increase!

```
pred = model.predict(spam)
pred.head()
```

```
0    0.334591
1    0.520799
2    0.500795
3    0.334591
4    0.334591
dtype: float64
```

```
pred.describe()
```

```
count    4601.000000
mean        0.394045
std         0.206687
min        -0.812540
25%         0.334591
50%         0.334591
75%         0.395442
max         3.849409
dtype: float64
```

Impossible predictions!

Logistic regression

Model the *probability* of occurrence

We seek to estimate the probability that

- it will rain on a particular day
- voter will vote for a Democrat
- message is spam

A fundamental problem

Probabilities fall in the range [0,1].

Linear combinations can take on any value in $(-\infty, +\infty)$.
[For example, the range of $3x_1+5x_2$ is $(-\infty, +\infty)$]

How can we transform $[0,1] \rightarrow (-\infty, +\infty)$?

Candidate functions

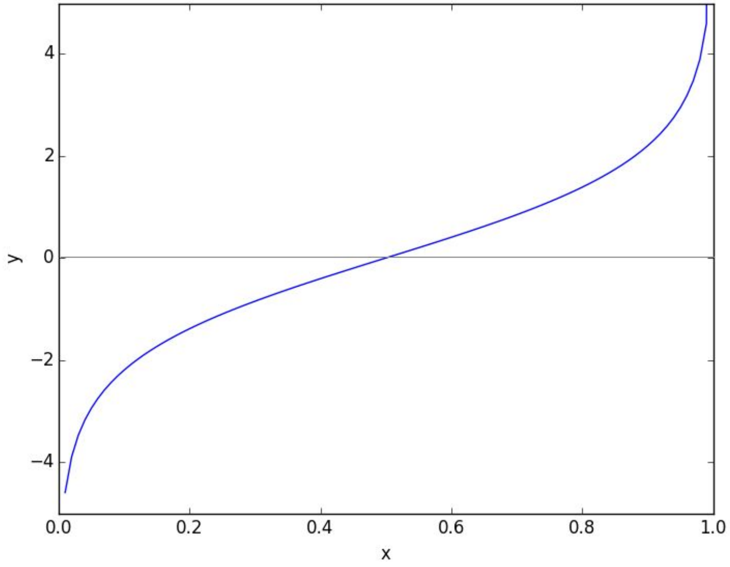
exp(x) maps [0, 1] → [1, e]

log(x) maps [0, 1] → (-∞, 0]

$$\operatorname{logit}(x) = \log \frac{x}{1-x}$$ maps [0,1] → (-∞, +∞)

The logit function

$$\operatorname{logit}(x) = \log \frac{x}{1-x}$$



More realistic outcomes

Instead of getting impossible outcomes with this model:

$$\Pr(Y_i = 1) = \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

We could more accurately model outcomes like this:

$$\log \frac{\Pr(Y_i=1)}{1-\Pr(Y_i=1)} = \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

Odds

$$\log \frac{\Pr(Y_i = 1)}{1 - \Pr(Y_i = 1)}$$

"log odds" of Yi=1 occurring

$$\frac{\Pr(Y_i = 1)}{1 - \Pr(Y_i = 1)}$$

"odds" of Yi=1 occurring

Odds

$$\frac{\Pr(Y_i = 1)}{1 - \Pr(Y_i = 1)}$$

"odds" of Yi=1 occurring

Suppose the probability you will win a race is 60%.

Your odds of winning are 3 to 2 (i.e., 1.5).

Odds

Suppose I told you that I could double your odds of winning.

Your current Pr(Win) is 0.6.

What is your new Pr(Win)?
[Discuss with neighbors]

Suppose I told you that I could double your odds of winning.
Your current Pr(Win) is 0.6.
What is your new Pr(Win)?

A) 1.2
B) 0.8
C) 0.75
D) Impossible to double odds

Odds

Suppose I told you that I could double your odds of winning.

Your current Pr(Win) is 0.6.

What is your new Pr(Win)?

Old odds are 1.5 $\longrightarrow$ New odds are 3 $\longrightarrow$ New Pr(Win) is .75

$$3 = \frac{\Pr(\text{Win})}{1 - \Pr(\text{Win})} \qquad \Pr(\text{Win}) = 0.75$$

Odds

What if we keep doubling our odds?

$$\text{odds} = \frac{p}{1-p}$$

$$p = \frac{\text{odds}}{1+\text{odds}}$$

Odds	p
1	0.5
2	0.67
4	0.8
8	0.88
16	0.94
32	0.97
64	0.98

Log odds

Probabilities must be between 0 and 1

Odds can be any number from 0 to infinity

Log odds can be any number from -infty to infty

Log odds

Instead of getting impossible outcomes with this model:

$$\Pr(Y_i = 1) = \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

We could more accurately model outcomes like this:

$$\log \frac{\Pr(Y_i=1)}{1-\Pr(Y_i=1)} = \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

Inverse functions

$$\text{logit}(x) = \log \frac{x}{1-x}$$

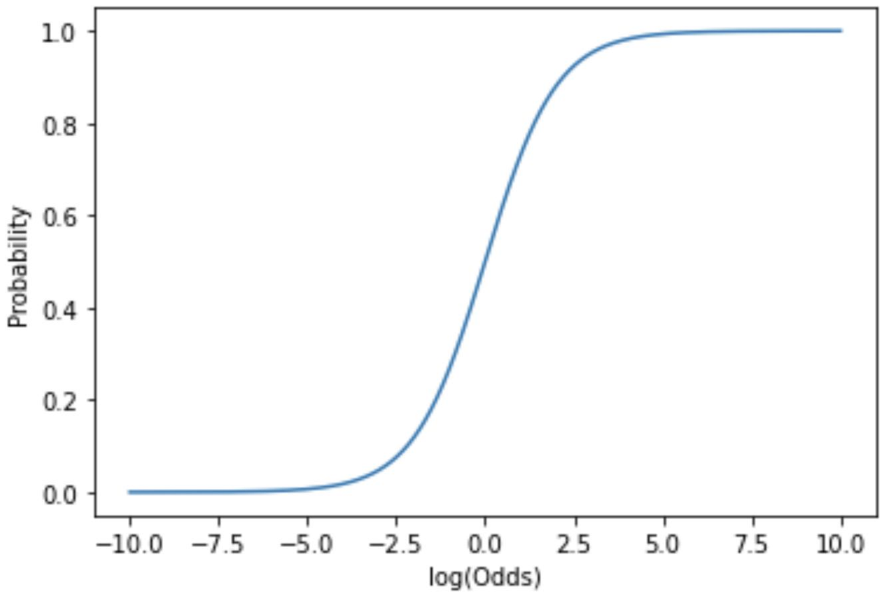
maps [0,1] → (-∞, +∞)

$$\text{logit}^{-1}(x) = \frac{e^x}{1+e^x}$$

maps (-∞, +∞) → [0,1]

```python
# inverse log-odds <--> inverse logit <--> expit
def inv_logit(log_odds):
  return(np.exp(log_odds)/(1+np.exp(log_odds)))

log_odds_range = np.linspace(-10, 10, 1000)
p_range = inv_logit(log_odds_range)
plt.plot(log_odds_range, p_range)
plt.xlabel("log(Odds)")
plt.ylabel("Probability")
plt.show()
```



Logistic regression

Model the *probability* of occurrence

$$\Pr(Y_i = 1) = \operatorname{logit}^{-1}\left(\beta_1 x_{i1} + \cdots + \beta_p x_{ip}\right)$$

$$\operatorname{logit}^{-1}(x) = \frac{e^x}{1 + e^x}$$

$$\operatorname{logit}(x) = \log\left(\frac{x}{1 - x}\right)$$

Maximum likelihood estimation

Logistic regression

$$\mathcal{L}(\beta) = \prod_{i=1}^{n} p_i(\beta)^{Y_i}(1 - p_i(\beta))^{1-Y_i}$$

$$p_i(\beta) = \text{logit}^{-1}(X_i\beta)$$

Very similar to Bernoulli MLE!

```python
formula = 'is_spam ~ 1 + char_dollar + credit + money + re'
model = smf.logit(formula=formula, data=spam).fit()
model.summary()
```

```
is_spam ~ 1 + char_dollar + credit + money + re
Optimization terminated successfully.
         Current function value: 0.481178
         Iterations 8
```

Logit Regression Results

Dep. Variable:	is_spam	**No. Observations:**	4601
Model:	Logit	**Df Residuals:**	4596
Method:	MLE	**Df Model:**	4
Date:	Tue, 09 May 2023	**Pseudo R-squ.:**	0.2824
Time:	22:06:40	**Log-Likelihood:**	-2213.9
converged:	True	**LL-Null:**	-3085.1
Covariance Type:	nonrobust	**LLR p-value:**	0.000

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	-1.0666	0.043	-24.680	0.000	-1.151	-0.982
char_dollar	11.8176	0.605	19.549	0.000	10.633	13.002
credit	2.3119	0.343	6.741	0.000	1.640	2.984
money	1.9933	0.248	8.022	0.000	1.506	2.480
re	-0.7755	0.099	-7.805	0.000	-0.970	-0.581

Interpreting logistic regression coefficients

$$\Pr(Y_i = 1) = \operatorname{logit}^{-1}\left(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}\right)$$

Interpreting logistic regression coefficients

$$\Pr(Y_i = 1) = \operatorname{logit}^{-1}(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip})$$

$$\operatorname{logit}(\Pr(Y_i = 1)) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

Interpreting logistic regression coefficients

$$\Pr(Y_i = 1) = \operatorname{logit}^{-1}(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip})$$

$$\operatorname{logit}(\Pr(Y_i = 1)) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

$$\log\left(\frac{\Pr(Y_i = 1)}{1 - \Pr(Y_i = 1)}\right) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

Interpreting logistic regression coefficients

$$\log\left(\frac{\Pr(Y_i = 1)}{1 - \Pr(Y_i = 1)}\right) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

"A 1 unit increase in x is associated with a **β** increase in the log odds of Yi=1".

But, the average gambler doesn't usually think on the log odds scale!

Interpreting logistic regression coefficients

$$\frac{\Pr(Y_i = 1)}{1 - \Pr(Y_i = 1)} = e^{\beta_0} e^{\beta_1 x_{i1}} \ldots e^{\beta_p x_{ip}}$$

"A 1 unit increase in x is associated with an e^{β} *multiplicative* increase of the odds of Yi=1"

```python
# Coefficients are additive log odds ratios
# '1pp more "money" words in email associated with
# increase of +2 in log odds that email is spam'
model.params
```

```
Intercept      -1.066563
char_dollar    11.817567
credit          2.311898
money           1.993280
re             -0.775505
dtype: float64
```

```python
# Exponentiated coefs are multiplicative odds ratios
# Easier to interpret
# '1pp more "money" words in email associated with
# 7.4x increase in odds that email is spam'
np.exp(model.params)
```

```
Intercept           0.344190
char_dollar    135613.881439
credit             10.093568
money               7.339570
re                  0.460471
dtype: float64
```

The "divide by 4" trick

For a logistic regression model, log odds increase linearly as x increases, but probabilities do not.

But, one can show that for any unit increase in x, Pr(Yi=1) can change by at most $\beta/4$.

For example, if $\beta=0.4$ for a fitted logistic regression model, then the maximum possible change in Pr(Yi=1) for any unit increase in x is 0.1.