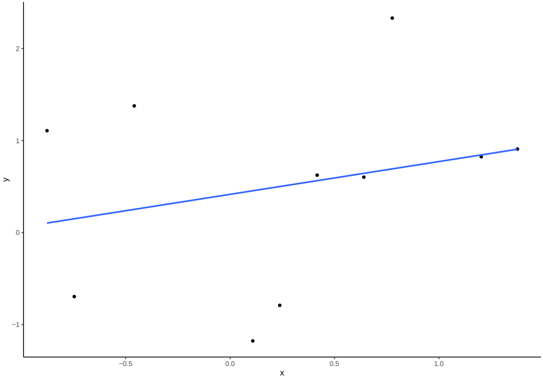
From Normal to General to Generalized Linear Models

Mark Andrews
Psychology Department, Nottingham Trent University

✉ mark.andrews@ntu.ac.uk

Regression models



- Regression models are often introduced as fitting lines to points.
- This is a limited perspective that makes understanding more complex regression models, like generalized linear models, harder to grasp.

Regression models

- Put simply and generally, a regression model is a model of how the probability distribution of one variable, known as the *outcome* variable and other names, varies as a function of other variables, known as the *explanatory* or *predictor* variables.
- The most common or basic type of regression models is the *normal linear* model.
- In normal linear models, we assume that the outcome variable is normally distributed and that its mean varies linearly with changes in a set of predictor variables.
- By understanding the normal linear model thoroughly, we can see how it can be extended to deal with data and problems beyond those that it is designed for.

Normal linear models

- In a normal linear model, we have n observations of an outcome variable:

  $$y_1, y_2 \ldots y_i \ldots y_n,$$

  and for each y_i, we have a set of $K \geqslant 0$ explantory variables:

  $$\vec{x}_1, \vec{x}_2 \ldots \vec{x}_i \ldots \vec{x}_n,$$

  where $\vec{x}_i = [x_{1i}, x_{2i} \ldots x_{ki} \ldots x_{Ki}]^\intercal$.
- We model $y_1, y_2 \ldots y_i \ldots y_n$ as observed values of the random variables $Y_1, Y_2 \ldots Y_i \ldots Y_n$.
- Each Y_i, being a random variable, is defined by a probability distribution, which we model as conditionally dependent on $\vec{x}_i$.
- In notation, for convenience, we often blur the distinction between an (ordinary) variable indicating an observed value and, e.g. y_i, and its corresponding random variable Y_i.

Normal linear models

- In normal linear models, we model $y_1, y_2 \ldots y_i \ldots y_n$ as follows:

$$y_i \sim N(\mu_i, \sigma^2), \quad \text{for } i \in 1 \ldots n,$$

$$\mu_i = \beta_0 + \sum_{k=1}^{K} \beta_k x_{ki}$$

- In words, each y_i is modelled a normal distribution, of equal variance σ^2, whose mean is a linear function of $\vec{x}_i$.
- From this model, for every hypothetically possible value of the K predictor variables, i.e. $\vec{x}_{i'}$, there is a corresponding mean $\mu_{i'}$, i.e. $\mu_{i'} = \beta_0 + \sum_{k=1}^{K} \beta_k x_{ki'}$.
- If we change $x_{ki'}$ by Δ_k, then $\mu_{i'}$ changes by $\beta_k \Delta_k$.

Examples

```r
weight_df <- read_csv(here('data/weight.csv'))
weight_male_df <- weight_df %>% filter(gender == 'male')

M_1 <- lm(weight ~ height, data = weight_male_df)

M_2 <- lm(weight ~ height + age, data = weight_male_df)
```

Categorical predictors

- To handle categorical predictors, we use binary re-coding of the values of the categorical predictor.
- In the simplest case of a dichomtomous categorical variables, e.g. gender $\in \{\text{female}, \text{male}\}$, we can recode gender as female $= 0$, male $= 1$.
- With $L > 2$ values, we can use a $L - 1$ *dummy* code, e.g.

race	x1	x2
black	0	0
white	0	1
hispanic	1	0

- We call linear normal models with categorical predictors *general linear models*[1]

[1] But this term is used in other, though related, ways too.

Examples

```r
M_3 <- lm(weight ~ height + gender, data = weight_df)

M_3 <- lm(weight ~ height + race, data = weight_male_df)
```

The problem of binary outcome data

- What if our outcome variable is binary, e.g.,

$$y_1, y_2 \cdots y_i \cdots y_n,$$

  with $y_i \in \{0, 1\}$?

- Modelling $y_1, y_2 \ldots y_n$ as samples from a normal distribution is an extreme example of *model misspecifcation*.
- Instead, we should use a more appropriate model.
- The easiest way to do this is to use an extension of the normal linear model.

Logistic regression's assumed model

- For all $i \in 1 \ldots n$,

  $$y_i \sim \mathrm{Bernoulli}(\theta_i),$$
  $$\mathrm{logit}(\theta_i) = \beta_0 + \sum_{k=1}^{K} \beta_k x_{ki},$$

  where

  $$\mathrm{logit}(\theta_i) \doteq \log\left(\frac{\theta_i}{1 = \theta_i}\right).$$

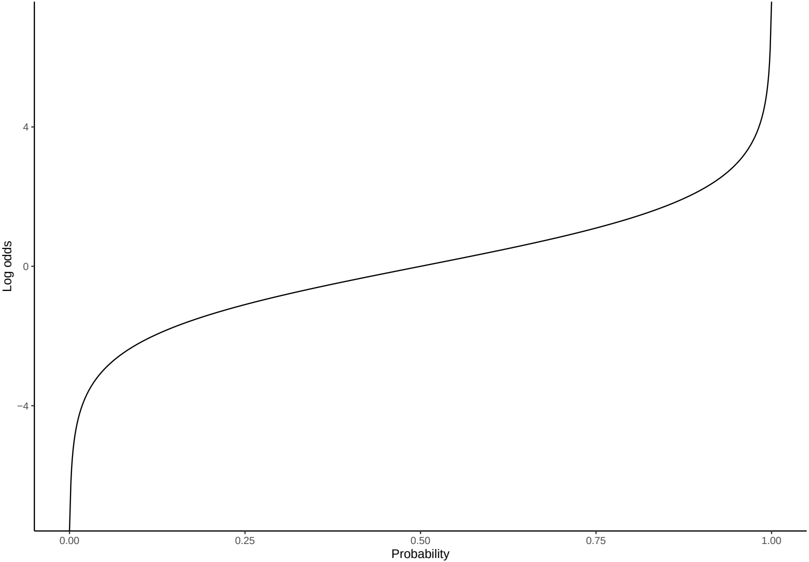
- In other word, we are saying that each observed outcome variable value $y_1, y_2 \ldots y_n$ is a sample from a *Bernoulli* distribution with parameter θ_i, and the log odds of θ_i is a *linear* function of the $\vec{x}_i$.

Odds

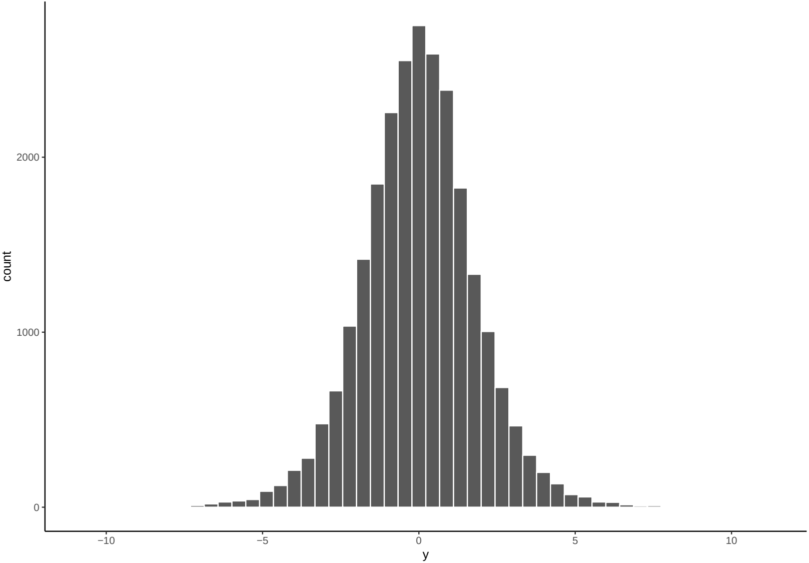
- Consider a coin toss. If the probability of Heads is p, then the probability of Tails is $1-p$.
- The odds of the coin coming up Heads on a single toss are given by
$$\frac{p}{1-p}.$$
- The odds simply gives the ratio of the probability of Heads to the probability of Tails.

Log odds (or logit)

- The log odds, or logit, is simply the logarithm of the odds.

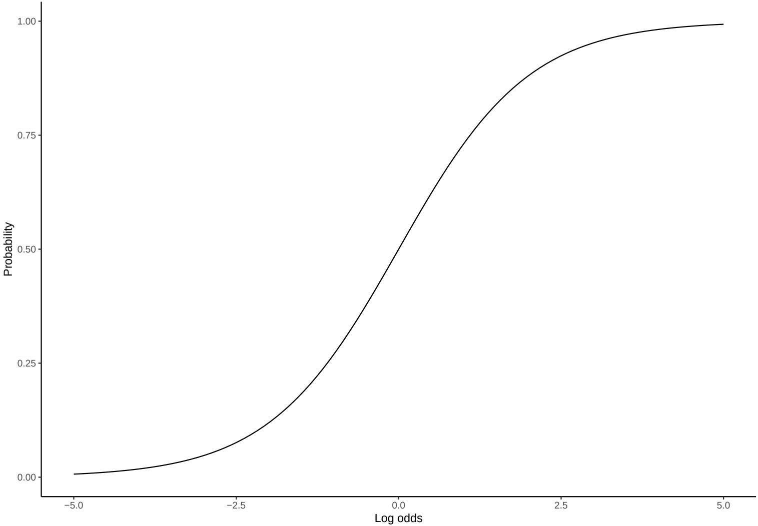


Logit transform of a uniform distribution over $(0, 1)$



The inverse logit transformation (ilogit)

- Just as we can map a probability to a log odds with the logit transformation, we can map a log odds back to a probability with the inverse logit, or ilogit, transformation:



Equivalent definitions of the binary logistic regression

- For all $i \in 1 \ldots n$,

$$y_i \sim \text{Bernoulli}(\theta_i),$$

$$\text{logit}(\theta_i) = \beta_0 + \sum_{k=1}^{K} \beta_k x_{ki}.$$

- This is equivalent to

$$y_i \sim \text{Bernoulli}(\theta_i),$$

$$\theta_i = \text{ilogit}\left(\beta_0 + \sum_{k=1}^{K} \beta_k x_{ki}\right),$$

where

$$\text{ilogit}(x) \triangleq \frac{1}{1 + e^{-x}}.$$

From probabilities to odds to logits, and back

$$\begin{tikzcd}
& \text{Probability: } \theta \arrow[dl, bend right, "\frac{\theta}{1-\theta}"'] \arrow[dr, bend left, "\log\left(\frac{\theta}{1-\theta}\right)"] & \\
\text{Odds: } \frac{\theta}{1-\theta} \arrow[ur, bend right, "\frac{\text{odds}}{1+\text{odds}}"'] \arrow[rr, bend right, "\log(\text{odds})"'] & & \text{Logit: } \log\left(\frac{\theta}{1-\theta}\right) \arrow[ul, bend right, "\frac{1}{1+e^{-\text{logit}}}"'] \arrow[ll, bend right, "e^{\text{logit}}"']
\end{tikzcd}$$

Examples

```
affairs_df <- read_csv(here('data/affairs.csv')) %>%
  mutate(cheater = affairs > 0)

M <- glm(cheater ~ yearsmarried,
         family = binomial(link = 'logit'),
         data = affairs_df)
```

Understanding β coefficients

- In linear models, a coefficient for a predictor variable has a straightforward interpretation: 1 unit change for a predictor variable corresponds to β change in the outcome variable.
- As logistic regression curves are nonlinear, the change in the outcome variable is not a constant function of change in the predictor.
- This makes interpretation more challenging.
- The most common means to interpret β coefficients is in terms of odds ratios.

Odds ratios

- We have seen that an odds in favour of an event are $\frac{p}{1-p}$.
- We can compare two odds with an odds ratio.
- For example, the odds of getting a certain job for someone with a MBA might be $\frac{p}{1-p}$, while the odds of getting the same job for someone without an MBA might be $\frac{q}{1-q}$.
- The ratio of the odds for the MBA to those of the non-MBA are

$$\frac{p}{1-p} \Big/ \frac{q}{1-q}$$

- This gives the factor by which odds for the job change for someone who gains an MBA.

β coefficients as (log) odds ratios

- Consider a logistic regression model with a single dichotomous predictor, i.e.

  $$\log\left(\frac{P(y_i = 1)}{1 - P(y_i = 1)}\right) = \alpha + \beta x_i,$$

  where $x_i \in \{0, 1\}$.
- The log odds that $y_i = 1$ when $x_i = 1$ is $\alpha + \beta$.
- The log odds that $y_i = 1$ when $x_i = 0$ is α.
- The log odds that $y_i = 1$ when $x_i = 1$ minus the log odds that $y_i = 1$ when $x_i = 0$ is

  $$(\alpha + \beta) - \alpha = \beta.$$

β *coefficients as (log) odds ratios*

- Let's denote the probability that $y_i = 1$ when $x_i = 1$ by p, and denote the probability that $y_i = 1$ when $x_i = 0$ by q.
- Subtracting the log odds is the log of the odds ratio, i.e.

$$\log\left(\frac{p}{1-p}\right) - \log\left(\frac{q}{1-q}\right) = \log\left(\frac{p}{1-p} \Big/ \frac{q}{1-q}\right) = \beta$$

- As such,

$$e^{\beta} = \frac{p}{1-p} \Big/ \frac{q}{1-q}.$$

- This provides a general interpretation for the β coefficients.

Prediction in logistic regression

- Given inferred values for $\beta_0, \beta_1 \ldots \beta_K$, the predicted log odds of the outcome variable given $\vec{x}_i$ is

$$\beta_0 + \sum_{k=1}^{K} \beta_k x_{ki}$$

- Knowing the predicted log odds, the predicted probability or predicted odds is easily calculated:

$$\text{ilogit}\left(\beta_0 + \sum_{k=1}^{K} \beta_k x_{ki}\right).$$

Examples

```r
affairs_df_new <- tibble(yearsmarried = c(1, 5, 10, 20, 25))

library(modelr)

# predicted log odds
affairs_df_new %>%
  add_predictions(M)

# predicted probabilities
affairs_df_new %>%
  add_predictions(M, type = 'response')
```

Model Fit with Deviance

- Once we have the estimates of the parameters, we can calculate *goodness of fit*.
- The *deviance* of a model is defined

  $$-2 \log L(\hat{\beta}|\mathcal{D}),$$

  where $\hat{\beta}$ are the maximum likelihood estimates.
- This is a counterpart to R^2 for generalized linear models.

Model Fit with Deviance: Model testing

- In a model with K predictors ($\mathcal{M}_1$), a comparison "null" model ($\mathcal{M}_0$) could be a model with a subset $K' < K$ of these predictors.
- The difference in the deviance of the null model minus the deviance of the full model is

$$\Delta_D = D_0 - D_1 = -2 \log \frac{L(\hat{\beta}_0|\mathcal{D})}{L(\hat{\beta}_1|\mathcal{D})},$$

  where $\hat{\beta}_1$ and $\hat{\beta}_0$ are the maximum likelihood estimators of the models $\mathcal{M}_1$ and $\mathcal{M}_0$, respectively.
- Under the null hypothesis, Δ_D is distributed as χ^2 with $K - K'$ degrees of freedom.
- In other words, under the null hypothesis that subset and full models are identical, the difference in the deviances will be distributed as a χ^2 with df equal to the difference in the number of parameters between the two models.

Example

```
M_1 <- glm(cheater ~ yearsmarried + age + gender,
           family = binomial(link = 'logit'),
           data = affairs_df)

# The "null" model
M_0 <- glm(cheater ~ yearsmarried,
           family = binomial(link = 'logit'),
           data = affairs_df)

anova(M_0, M, test = 'Chisq')
```

Ordinal outcome variables

- Ordinal variables have values that can be ordered but these values are not in a metric space.
- For example, “very unsatisfied”, “unsatisfied”, “neutral”, “satisfied” and “very satisfied” are ordinal values. The “distance” between “very unsatisfied” and “unsatisfied” is not necessarily the same as the “distance” between “satisfied” and “very satisfied”.
- We can model ordinal data using an extension of the binary logistic regression model.
- To understand this, we must understand the latent variable formulation of binary logistic regression.

Latent variable formulation of binary logistic regression

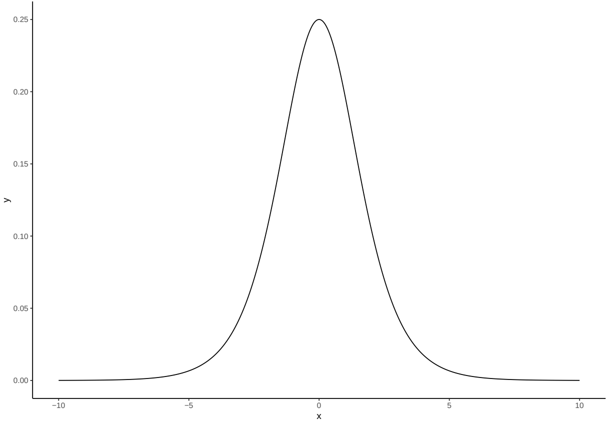
- We may describe a logistic regression exactly using the following latent variable formulation.
- For all $i \in 1 \ldots n$,

$$y_i = \begin{cases} 1, & \text{if } z_i \geqslant 0 \\ 0, & \text{if } z_i < 0 \end{cases},$$

$$z_i \sim \text{dlogis}(\beta_0 + \sum_{k=1}^{K} \beta_k x_{ki}),$$

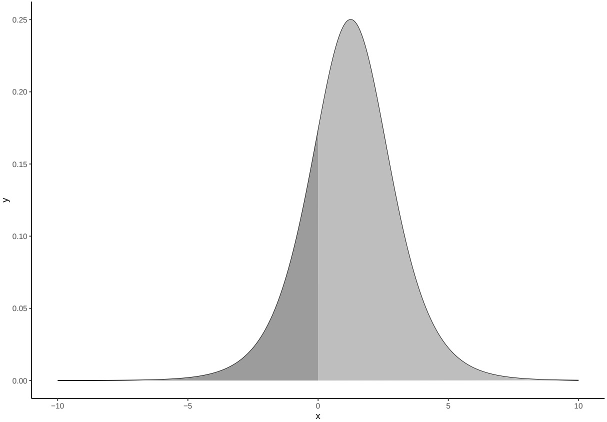
where dlogis is a *logistic distribution*.

Standard logistic distribution



- The standard logistic distribution has a mean of 0 and scale parameter of 1.

Logistic distribution with of 1.25



- Here, we shade the area above and below 0.

Latent variable formulation for ordinal logistic regression

- Let us assume that each $y_i \in \{0, 1, 2\}$ (or any other ordered values).
- We introduce two *cutpoints*: ζ_1 and ζ_2.
- Then our *cumulative logit* model is: for all $i \in 1 \ldots n$,

$$y_i = \begin{cases} 2, & \text{if } z_i \geqslant \zeta_2 \\ 1, & \text{if } \zeta_1 \leqslant z_i < \zeta_2 \\ 0, & \text{if } z_i < \zeta_1 \end{cases},$$

$$z_i \sim \text{dlogis}(\sum_{k=1}^{K} \beta_k x_{ki}).$$

- In general, for L ordered values, we have $L - 1$ cutpoints, which defines L regions under the logistic distribution.

$$-\infty < \zeta_1 < \zeta_2 < \ldots < \zeta_{L-1} < \infty$$

Example

```r
library(pscl)
library(MASS)
M <- polr(score ~ gre.quant, data=admit)

admit_df_new <- tibble(gre.quant = c(600, 700, 800))
add_predictions(admit_df_new, M, type = 'probs')

# compare to
plogis(q = M$zeta, location = M$coefficients * 600)
```