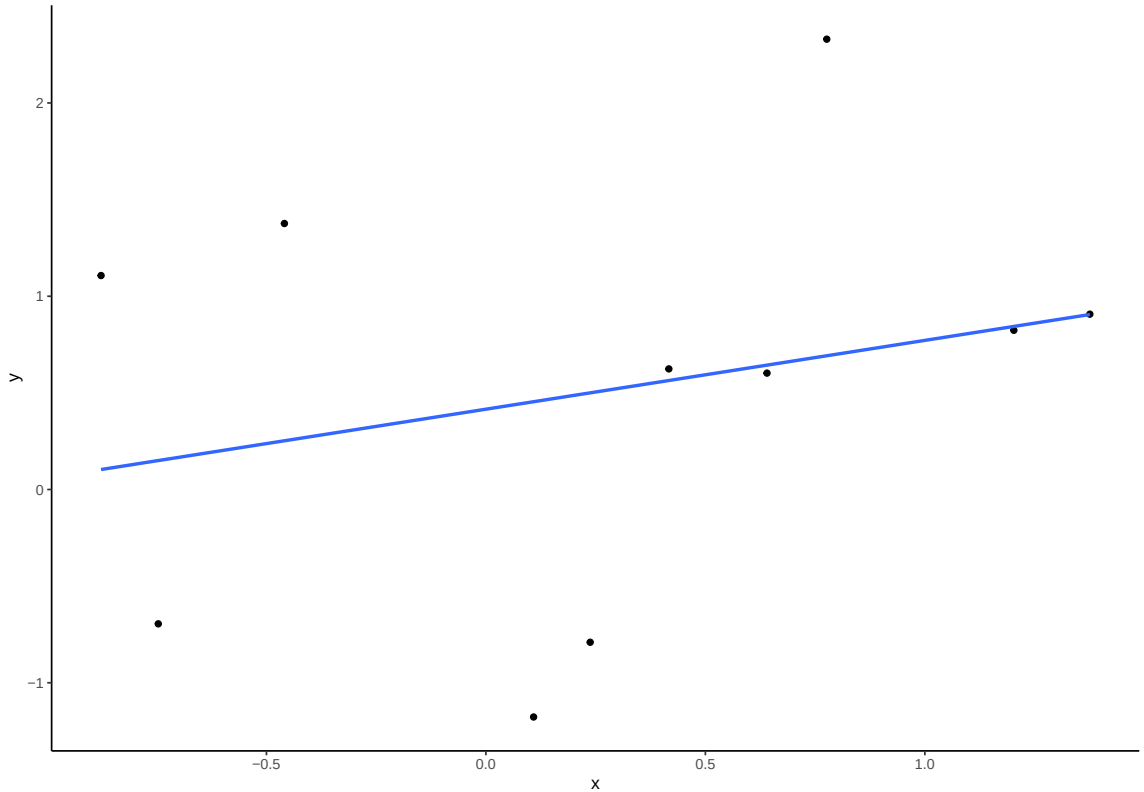
General & Generalized Linear Models

Mark Andrews

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:

  $$y_1, y_2 \dots y_i \dots y_n,$$

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

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

  where $\vec{x}_i = [x_{1i}, x_{2i} \dots x_{ki} \dots x_{Ki}]^\intercal$.
- We model $y_1, y_2 \dots y_i \dots y_n$ as observed values of the random variables $Y_1, Y_2 \dots Y_i \dots Y_n$.
- 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

```
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

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

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

Nested model comparison

- Model M_0 is nested in model M_1 if the parameter space of M_0 is a subset of the parameter space of M_1.
- For example, if M_0 is the following linear model:

$$\text{for } i \in 1 \dots n, \quad y_i = \beta_0 + \beta_1 x_{1i} + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2),$$

  its parameter space is β_0, β_1, σ^2.
- If M_1 is the following linear model:

$$\text{for } i \in 1 \dots n, \quad y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2),$$

  its parameter space is β_0, β_1, β_2, σ^2.
- Any set of values of β_0, β_1, σ^2 in M_0 is a point in the parameter space of β_0, β_1, β_2, σ^2 of M_1 if we simply set $\beta_2 = 0$.
- In other words, we can make M_0 with any given values of β_0, β_1, σ^2 from M_1 by setting β_0, β_1, σ^2 in M_1 to these same values and setting $\beta_2 = 0$.

Nested normal linear models

- We can compare nested normal linear models using F tests.
- Assume M_0 and M_1 are normal linear models, with M_0 nested in M_1.
- We calculate RSS_0 and RSS_1, the residual sums of squares of M_0 and M_1, respectively.
- RSS_0 will be greater than or equal to RSS_1.
- Then

$$\text{proportional increase in error} = \frac{\text{increase in error}}{\text{minimal error}},$$

$$= \frac{\text{RSS}_0 - \text{RSS}_1}{\text{RSS}_1},$$

Residual sum of squares

- The sum of squared residuals in a normal linear model is

$$\text{RSS} = \sum_{i=1}^{n} |y_i - (\beta_0 + \sum_{k=1}^{K} \beta_k x_{ki})|^2.$$

- The RSS when using the maximum likelihood estimators is

$$\begin{aligned}\text{RSS} &= \sum_{i=1}^{n} |y_i - (\beta_0 + \sum_{k=1}^{K} \beta_k x_{ki})|^2, \\ &= \sum_{i=1}^{n} |y_i - \hat{y}_i|^2\end{aligned}$$

Nested normal linear models

```
M1 <- lm(Fertility ~ Agriculture + Education + Catholic, data =
M0 <- lm(Fertility ~ Agriculture + Education, data = swiss)

RSS_0 <- sum(residuals(M0)^2)
RSS_1 <- sum(residuals(M1)^2)

c(RSS_0, RSS_1)
#> [1] 3953.270 2567.884
```

```
(RSS_0 - RSS_1)/RSS_1
#> [1] 0.5395049
```

In other words, RSS_0 is 1.54 greater than RSS_1.

Nested normal linear models

- The F ratio is

$$F = \underbrace{\frac{RSS_0 - RSS_1}{RSS_1}}_{\text{effect size}} \times \underbrace{\frac{df_1}{df_0 - df_1}}_{\text{sample size}} = \frac{(RSS_0 - RSS_1)/(df_0 - df_1)}{RSS_1/df_1}.$$

  where df_1 is $N - (K_1 + 1)$, where K_1 is number of (predictor; excluding intercept) coefficients in M_1.

```
df_0 <- M0$df.residual
df_1 <- M1$df.residual
c(df_0, df_1, df_0 - df_1, df_1/(df_0 - df_1))
#> [1] 44 43  1 43
```

Nested normal linear models

```
RSS_0
#> [1] 3953.27
RSS_1
#> [1] 2567.884
RSS_0 - RSS_1
#> [1] 1385.386
df_0 - df_1
#> [1] 1
df_1
#> [1] 43
(RSS_0 - RSS_1)/(df_0 - df_1)
#> [1] 1385.386
RSS_1/df_1
#> [1] 59.71823
((RSS_0 - RSS_1)/(df_0 - df_1))/(RSS_1/df_1)
#> [1] 23.19871
```

Nested normal linear models

```
anova(M0, M1)
#> Analysis of Variance Table
#>
#> Model 1: Fertility ~ Agriculture + Education
#> Model 2: Fertility ~ Agriculture + Education + Catholic
#>   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
#> 1     44 3953.3
#> 2     43 2567.9  1    1385.4 23.199 1.842e-05 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
```

Nested normal linear models

```
drop1(M1, scope = ~ Catholic, test = 'F')
#> Single term deletions
#>
#> Model:
#> Fertility ~ Agriculture + Education + Catholic
#>           Df Sum of Sq    RSS    AIC F value    Pr(>F)
#> <none>                 2567.9 196.03
#> Catholic  1    1385.4 3953.3 214.31  23.199 1.842e-05 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
```

Nested normal linear models

```
drop1(M0, scope = ~ Education, test = 'F')
#> Single term deletions
#>
#> Model:
#> Fertility ~ Agriculture + Education
#>             Df Sum of Sq    RSS    AIC F value    Pr(>F)
#> <none>                   3953.3 214.31
#> Education  1    2329.8 6283.1 234.09  25.931 7.105e-06 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
```

Nested normal linear models

```
drop1(M0, scope = ~ Education + Agriculture, test = 'F')
#> Single term deletions
#>
#> Model:
#> Fertility ~ Agriculture + Education
#>             Df Sum of Sq    RSS    AIC F value    Pr(>F)
#> <none>                   3953.3 214.31
#> Education    1   2329.85 6283.1 234.09 25.9312 7.105e-06 ***
#> Agriculture  1     61.97 4015.2 213.04  0.6897    0.4108
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
```

R^2

- If we have two models, M_0 and M_1, with M_0 nested in M_1, and with residual sums of squares RSS_0 and RSS_1, respectively, we can calculate:

$$\text{proportional decrease in error} = \frac{\text{decrease in error (from } M_0 \text{ to } M_1)}{\text{error in } M_0},$$

$$= \frac{RSS_0 - RSS_1}{RSS_0},$$

$$= R^2$$

```
(RSS_0 - RSS_1) / RSS_0
#> [1] 0.3504405
```

- In other words, the reduction in error from M_0 to M_1 is 0.35 of the error of M_0.

R^2: *The coefficient of determination*

- It can be shown that

$$\underbrace{\sum_{i=1}^{n}(y_i-\bar{y})^2}_{\text{TSS}}=\underbrace{\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2}_{\text{ESS}}+\underbrace{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}_{\text{RSS}},$$

  where TSS is *total* sum of squares, ESS is *explained* sum of squares, and RSS is *residual* sum of squares.

- The coefficient of determination R^2 is defined as

$$\begin{aligned}R^2 &= \frac{\text{ESS}}{\text{TSS}} = \text{Proportion of variation that is explained},\\ &= 1-\frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2}\end{aligned}$$

R^2

- If M_0 is a *null* model, i.e. no predictors, then TSS = RSS_0.
- It can be shown that

$$\underbrace{\sum_{i=1}^{n}(y_i - \bar{y})^2}_{RSS_0} = \underbrace{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}_{RSS_0 - RSS_1} + \underbrace{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}_{RSS_1}.$$

- As such, R^2 is defined as

$$R^2 = \frac{RSS_0 - RSS_1}{RSS_0} = 1 - \frac{RSS_1}{RSS_0},$$

or 1 minus the error of M_1 relative to M_0.

R^2

```
M_null <- lm(Fertility ~ 1, data = swiss)
RSS_null <- sum(residuals(M_null)^2)
RSS_0 / RSS_null
#> [1] 0.5507516
1 - RSS_0 / RSS_null
#> [1] 0.4492484
(RSS_null - RSS_0) / RSS_null
#> [1] 0.4492484
summary(M0)$r.squared
#> [1] 0.4492484
```

Adjusted R^2

- By explaining proportion of variance explained, R^2 is used a *goodness of fit* measure.
- However, R^2 will always grow with K, the number of predictors.
- R^2 can be *adjusted* to counteract the artificial effect of increasing numbers of predictors as follows:

$$R^2_{\text{Adj}} = \underbrace{1 - \frac{\text{RSS}}{\text{TSS}}}_{R^2} \underbrace{\frac{n-1}{n-K-1}}_{\text{penalty}},$$

  where n is sample size.
- R^2_{Adj} is not identical to the proportion of variance explained in the *sample*, but is an unbiased measured of the population R^2.

Adjusted R^2

```
n <- nrow(M0$model)
K <- length(coef(M0)) - 1 # no. of predictor coefs
penalty <- (n - 1)/(n - K - 1)
1 - (RSS_0 / RSS_null) * penalty
#> [1] 0.4242143
summary(M0)$adj.r.squared
#> [1] 0.4242143
```

The problem of binary outcome data

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

$$y_1, y_2 \dots y_i \dots y_n,$$

  with $y_i \in \{0, 1\}$?
- Modelling $y_1, y_2 \dots y_n$ 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 $i \in 1 \ldots n$,

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

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

  where

$$\text{logit}(\theta_i) \doteq \log\left(\frac{\theta_i}{1-\theta_i}\right).$$

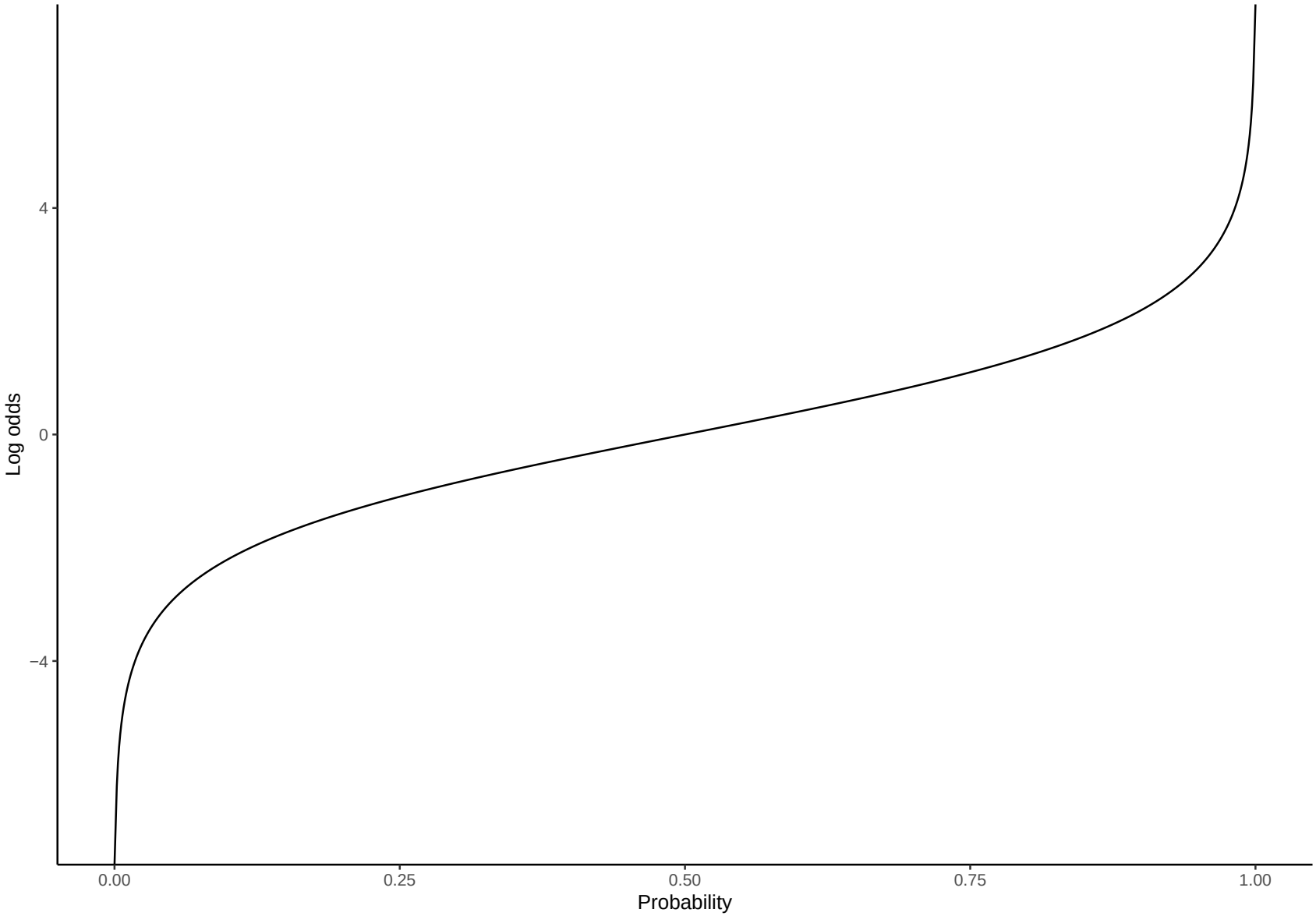
- In other word, we are saying that each observed outcome variable value $y_1, y_2 \ldots y_n$ is a sample from a *Bernoulli* distribution with parameter θ_i, and the log odds of θ_i is a *linear* function of the $\vec{x}_i$.

Odds

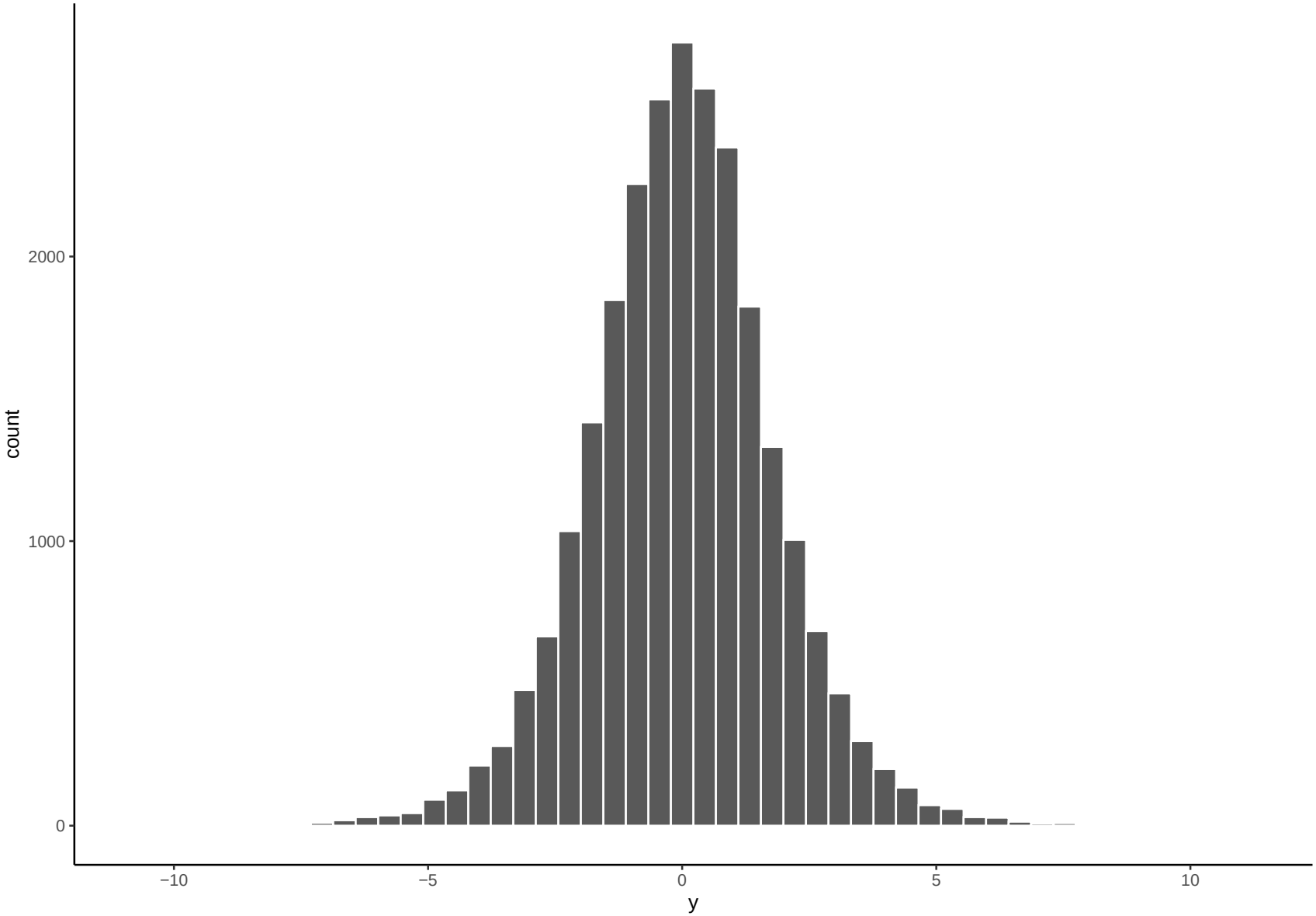
- Consider a coin toss. If the probability of Heads is p, then the probability of Tails is $1 - p$.
- The odds of the coin coming up Heads on a single toss are given by
$$\frac{p}{1-p}.$$
- The odds simply gives the ratio of the probability of Heads to the probability of Tails.

Log odds (or logit)

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

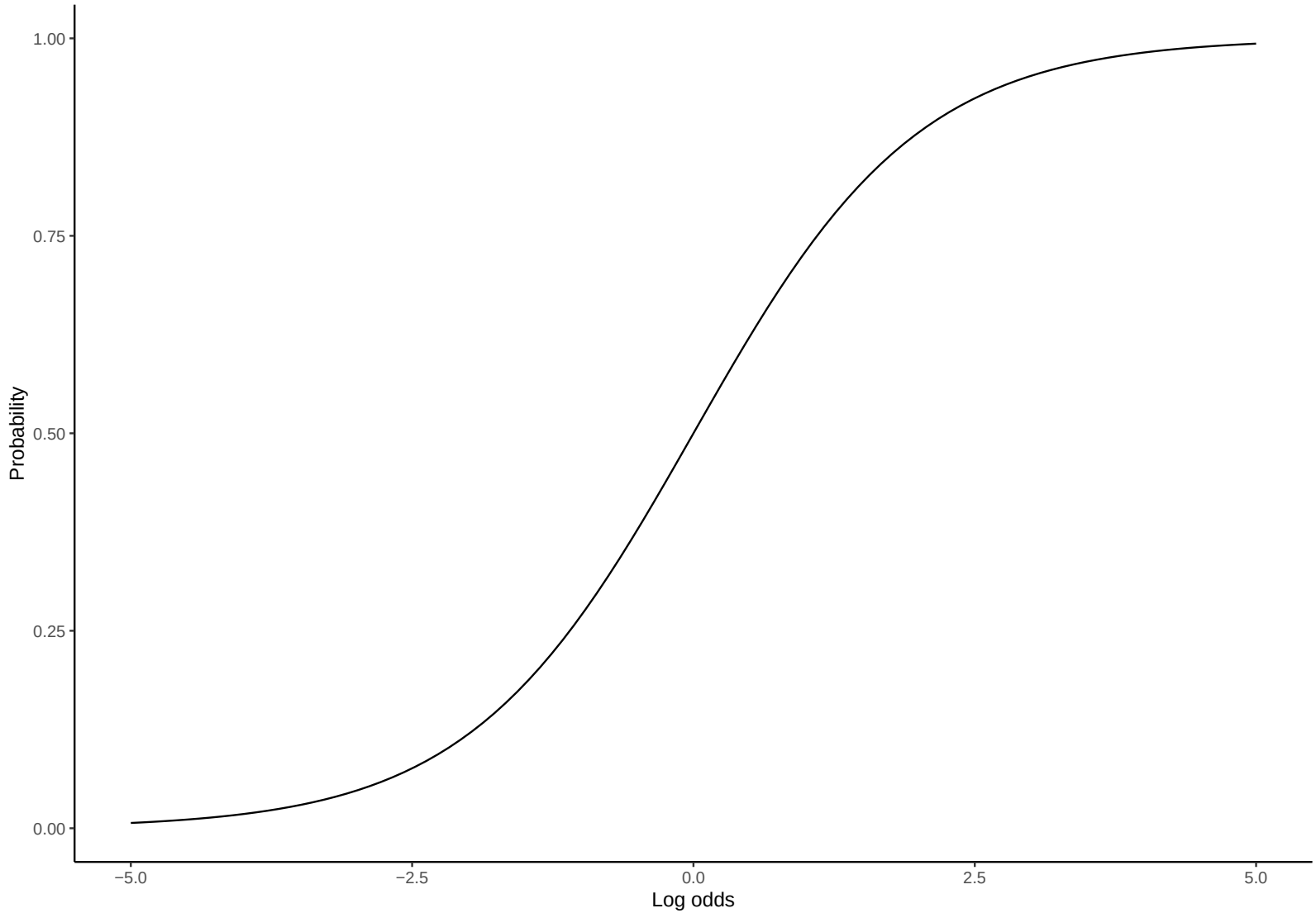


Logit transform of a uniform distribution over $(0, 1)$



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 $i \in 1 \dots n$,

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

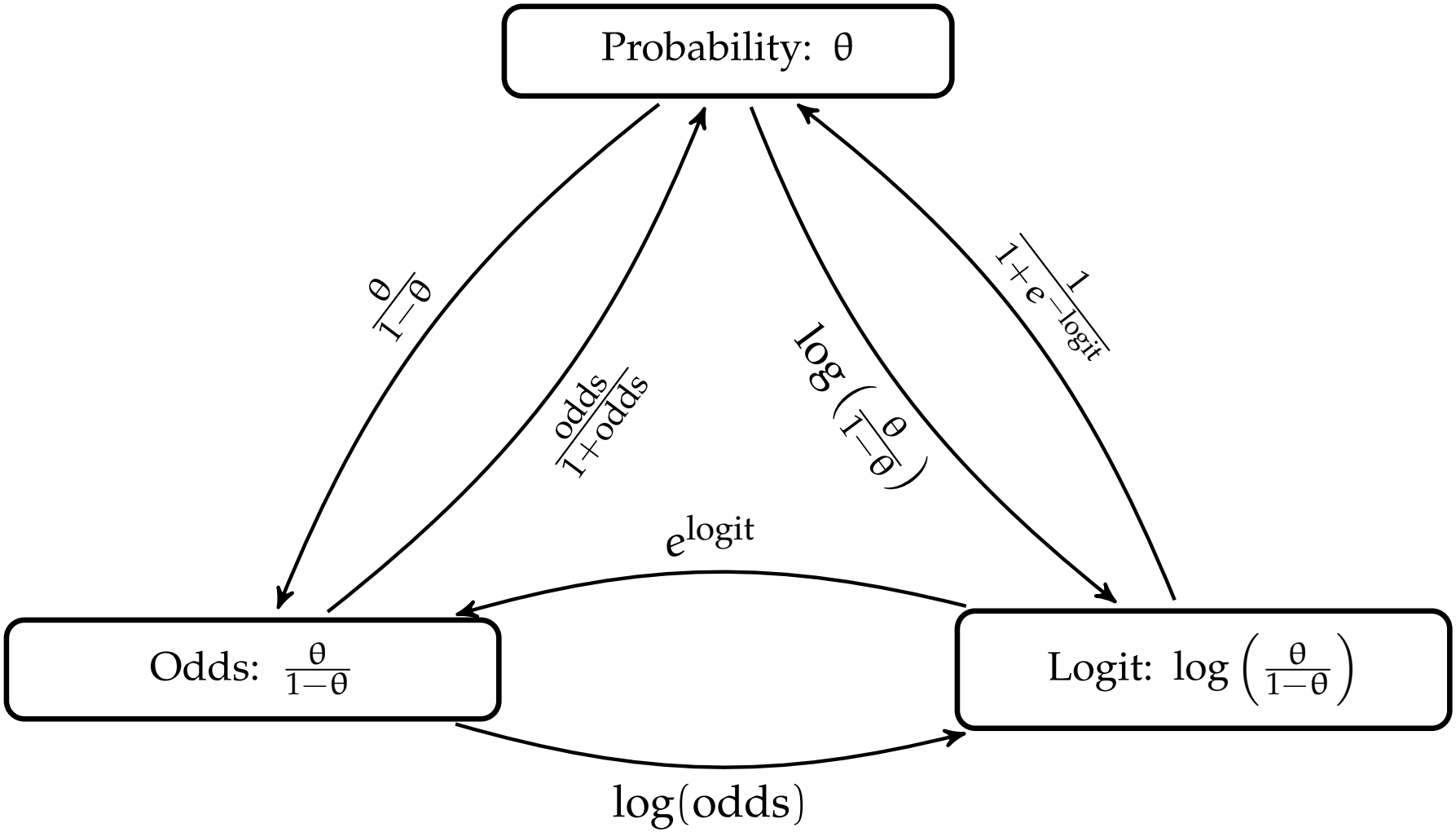
- This is equivalent to

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

where

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

From probabilities to odds to logits, and back



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

```
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

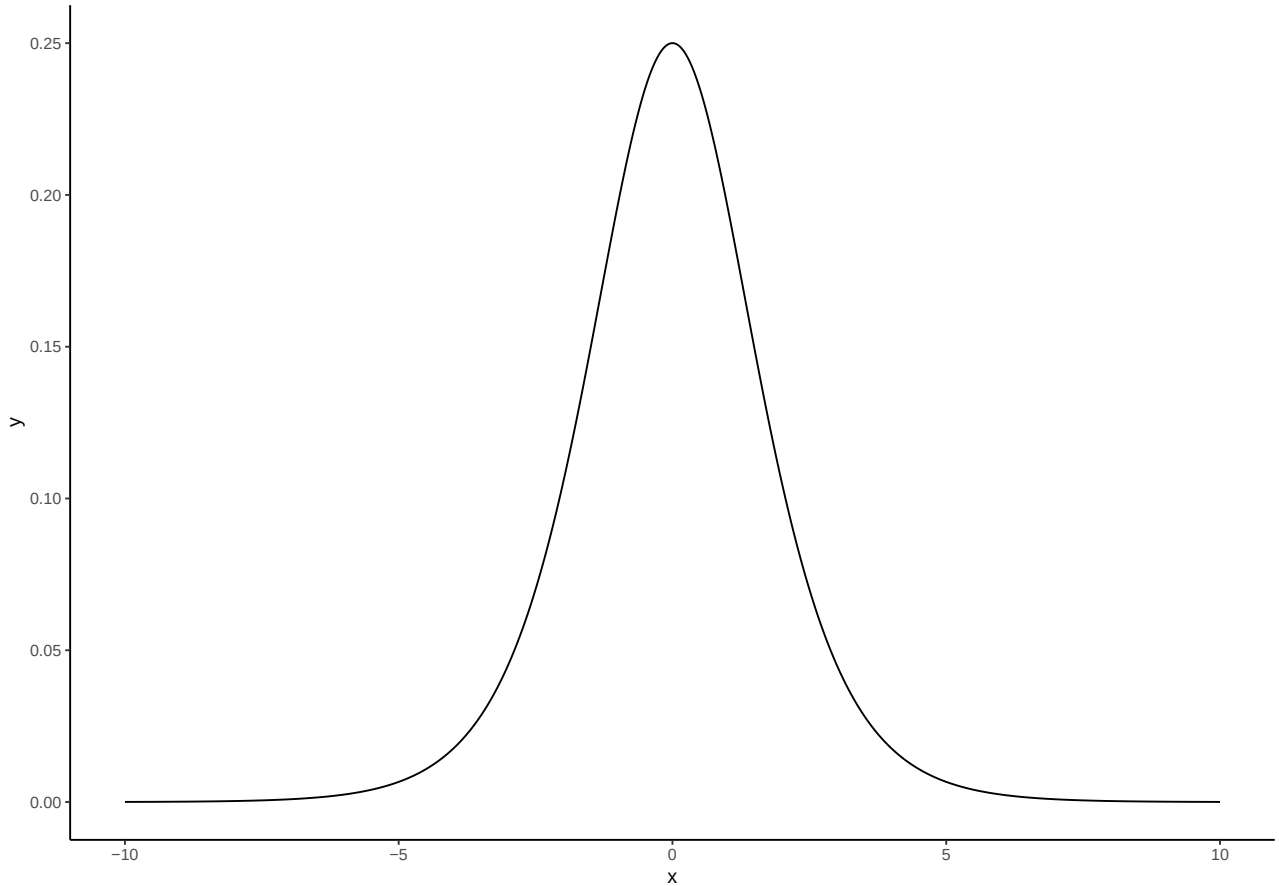
- We may describe a logistic regression exactly using the following latent variable formulation.
- For all $i \in 1 \dots n$,

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

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

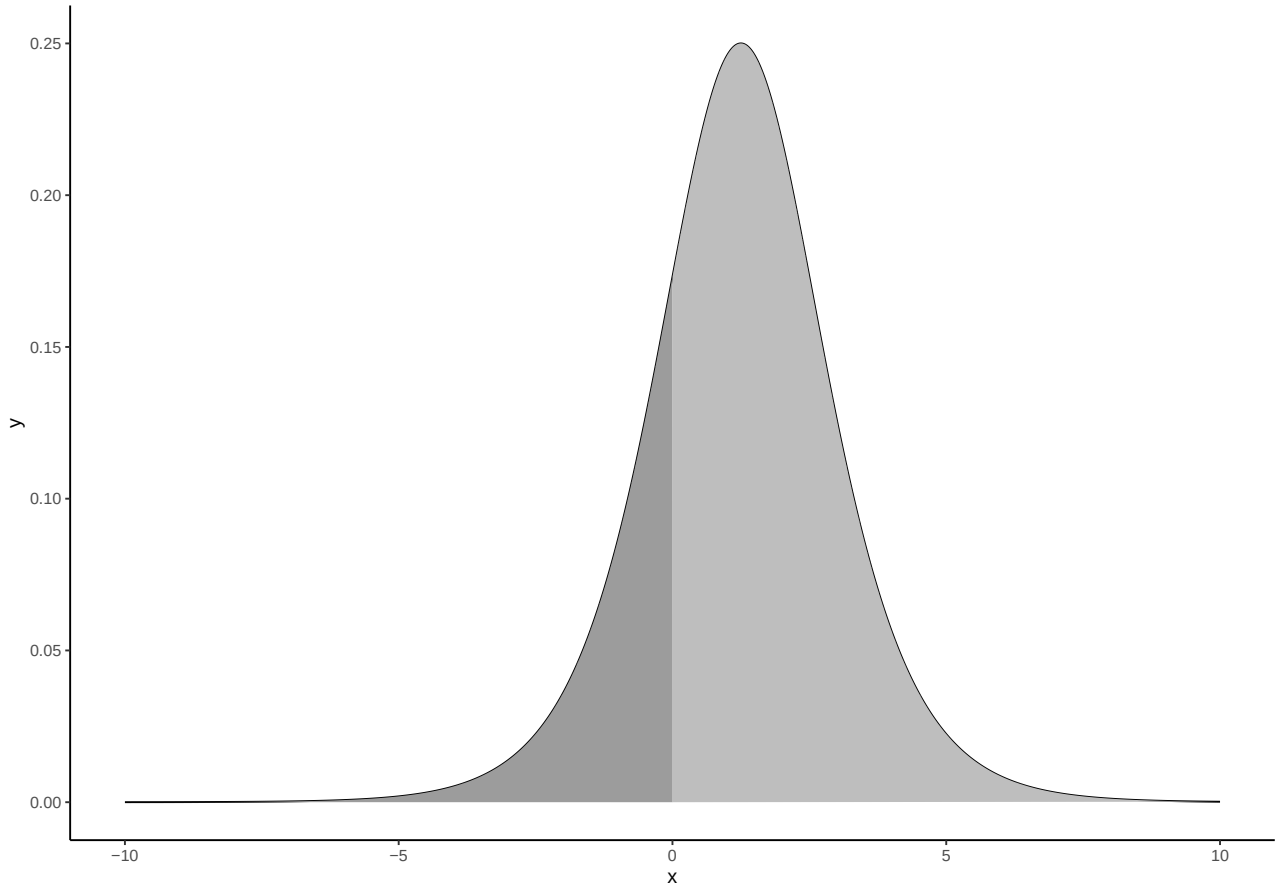
where dlogis is a *logistic distribution*.

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 $i \in 1 \dots n$,

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

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

- 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

```
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)
```

Categorical logistic regression

- If each $y_i \in \{1, 2 \dots L\}$, where these L values are treated as categorically distinct, then we model y_i as follows.

$$\log\left(\frac{P(y_i = l)}{P(y_i = 1)}\right) = \beta_{l0} + \sum_{k=1}^{K} \beta_{lk} x_{ki}, \quad \text{for } l \in \{2 \dots L\},$$

and

$$\log\left(\frac{P(y_i = 1)}{P(y_i = 1)}\right) = 0.$$

- This is equivalent to

$$P(y_i = l) = \frac{e^{z_{li}}}{1 + \sum_{l=2}^{L} e^{z_{li}}},$$

where

$$z_{li} = \beta_{l0} + \sum_{k=1}^{K} \beta_{lk} x_{ki}.$$

Examples

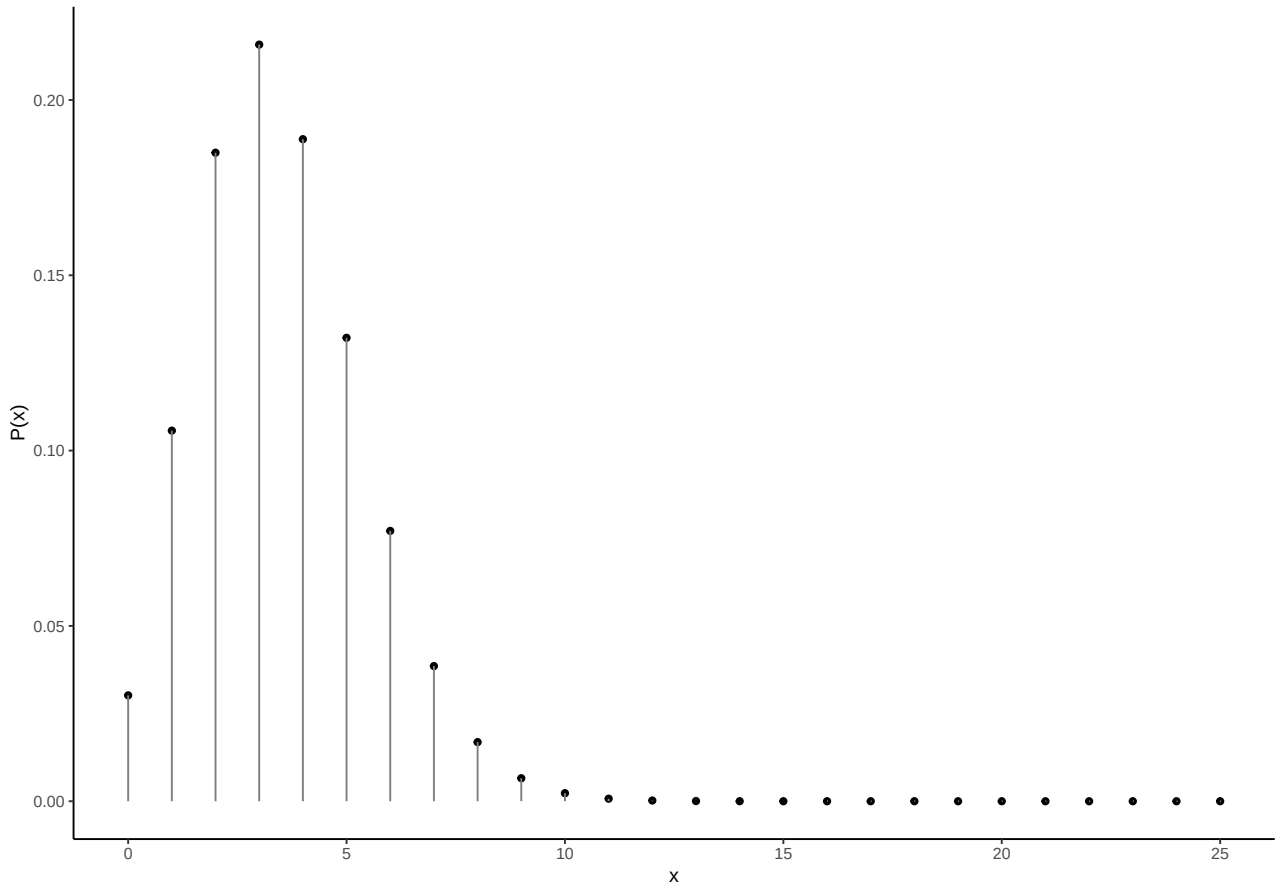
```
library(nnet)
M <- multinom(score ~ gre.quant, data=admit)


add_predictions(admit_df_new, M, type = 'probs')

# Compare to
z <- coef(M) %*% c(1, 600)
c(1, exp(z)) / sum(c(1, exp(z)))
```

The Poisson Distribution

- The Poisson distribution is a discrete probability distribution over the non-negative integers $0, 1, 2 \ldots$.



Shown here is a Poisson distribution with $\lambda = 3.5$.

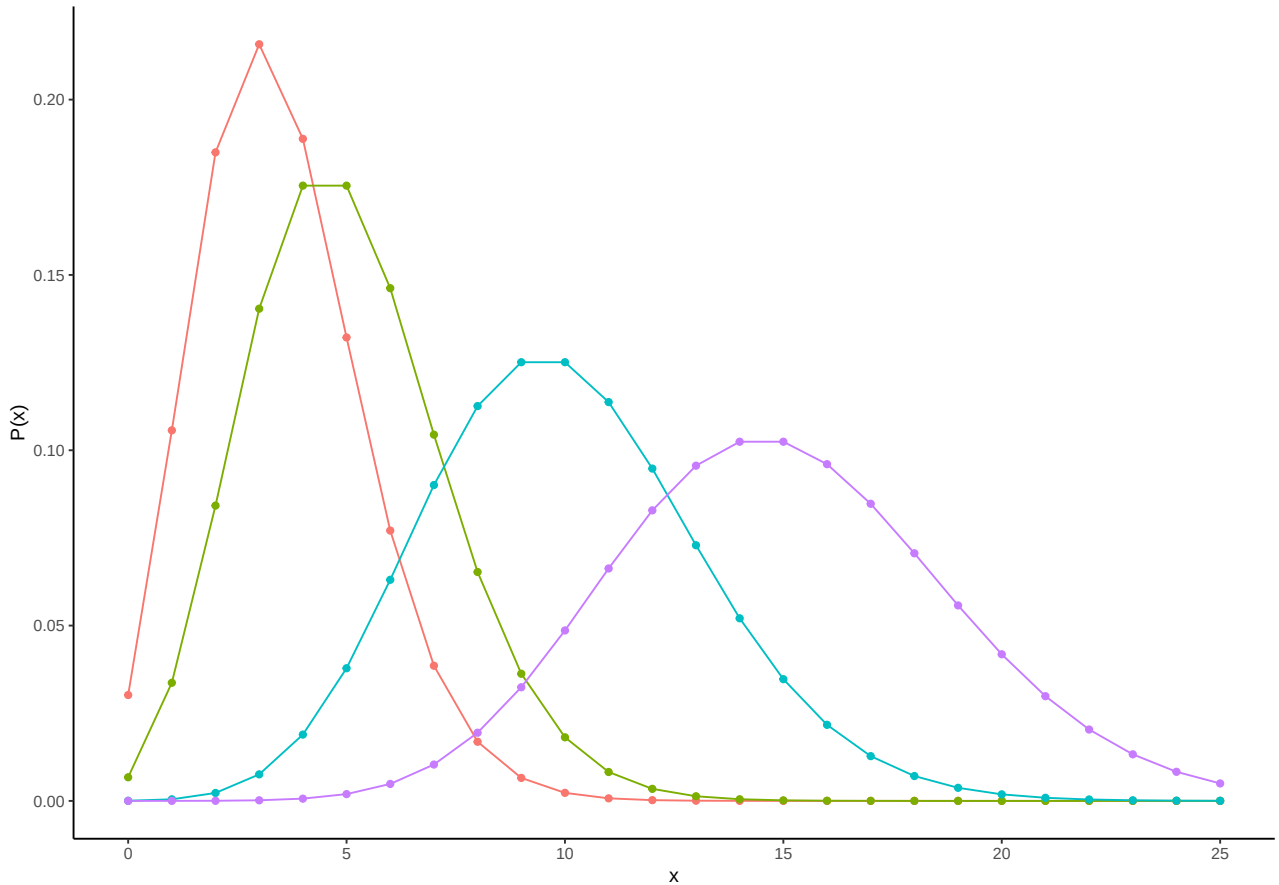
The Poisson Distribution

- The Poisson distribution is used to model the probability of a given number of events occurring in a fixed interval of time, e.g. the number of emails you get per hour, the number of shark attacks on Bondi beach every summer, etc.
- It has a single parameter λ, known as the *rate*.
- If x is a Poisson random variable whose, its probability mass function is

$$P(x = k|\lambda) = \frac{e^{-\lambda}\lambda^k}{k!}.$$

The Poisson Distribution

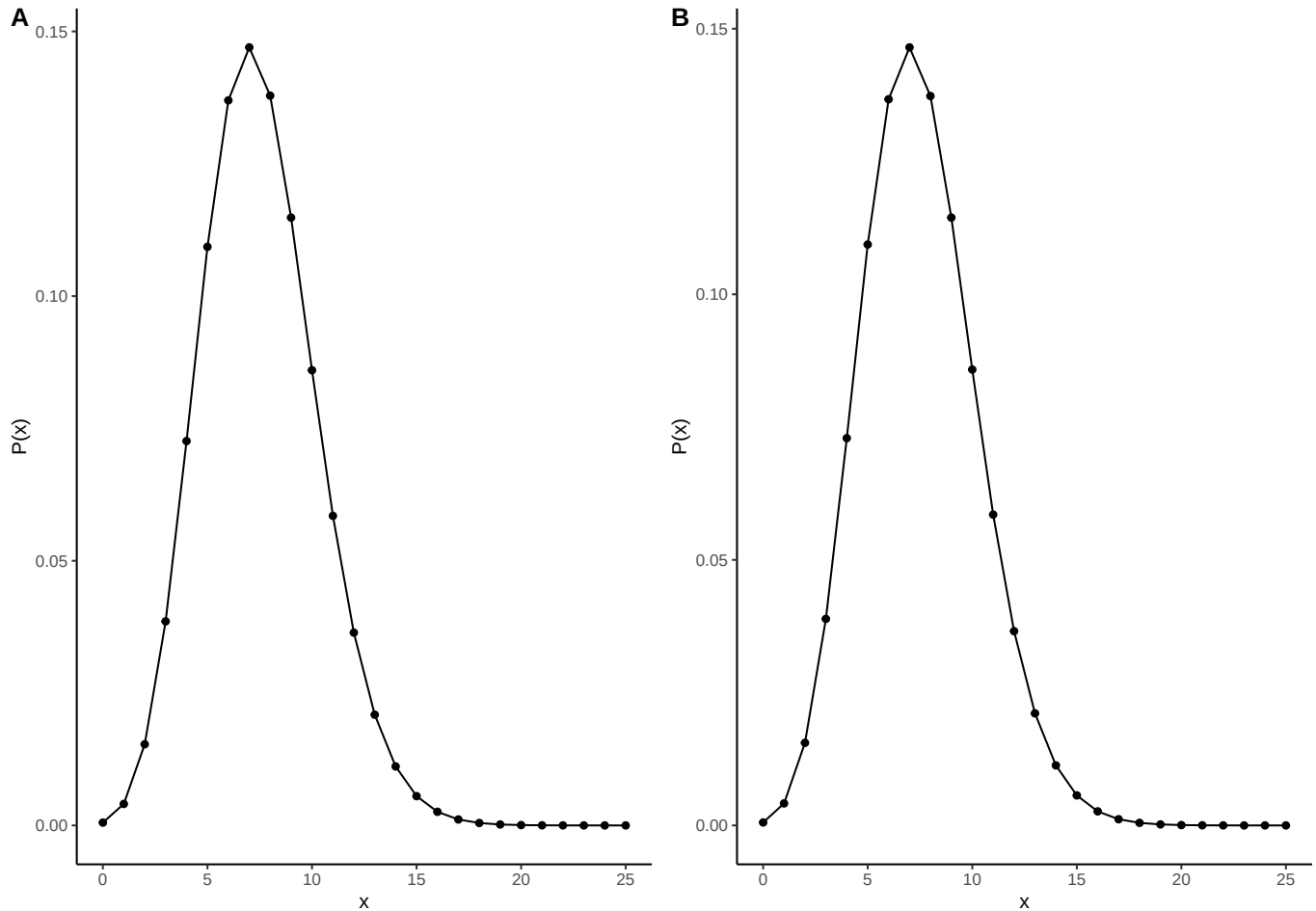
- The mean of a Poisson distribution is equal to its rate parameter λ.
- Its variance is also equal to λ.



As λ increases, so too does the variance.

The Poisson Distribution

- The Poisson distribution can be seen as the limit of a Binomial distribution as $N \to \infty$, and $\lambda = pN$.
- Shown are (left) Binomial$(N, p = \lambda/N)$ where $N \approx 10^3$ and $\lambda = 7.5$, and (right) Poisson(λ).



Poisson Regression

- In any regression problem, our data are $(y_1, x_1), (y_2, x_2) \dots (y_n, x_n)$, where each y_i is modelled as a stochastic function of x_i.
- In Poisson regression, we assume that each y_i is a Poisson random variable rate λ_i and

$$\log(\lambda_i) = \beta_0 + \sum_{k=1}^{K} \beta_k x_{ki},$$

or equivalently

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

Poisson Regression

- As an example of Poisson regression, we can look at the number visits to a doctor in a fixed period as a function of predictors such as gender.
- Using a data-set of over 5000 people, we estimate (using mle) that

$$\log(\lambda_i) = 1.65 + 0.43 \times x_i$$

where $x_i = 1$ for a female, and $x_i = 0$ for a male.

Poisson Regression

- Using this example, we see that for a female

$$\lambda_{\text{Female}} = e^{1.65+0.43} = 8.004$$

  and for males

$$\lambda_{\text{Male}} = e^{1.65} = 5.2$$

- In other words, the expected value for females is 8.2 and for males it is 5.2.

Coefficients

- In Poisson regression, coefficients can be understood as follows.
- In the previous example,

$$\begin{aligned} \lambda_{\text{Female}} &= e^{1.65+0.43}, \\ &= e^{1.65} e^{0.43}, \\ \lambda_{\text{Male}} &= e^{1.65}. \end{aligned}$$

- This means that the exponent of the gender coefficient, i.e. $e^{0.43}$, signifies the multiplicative increase to the average rate of doctor visits for women relative men.
- In other words, women visit doctors on average $e^{0.43} = 1.53$ times more than men.

Coefficients

- In an arbitrary example with a single continuous predictor variable,

$$\begin{aligned}\lambda &= e^{\alpha+\beta x_i},\\ &= e^{\alpha}e^{\beta x_i},\end{aligned}$$

  If we increase x_i by 1, we have

$$\begin{aligned}\lambda^{+} &= e^{\alpha+\beta(x_i+1)},\\ &= e^{\alpha+\beta x_i+\beta},\\ &= e^{\alpha}e^{\beta x_i}e^{\beta},\end{aligned}$$

- As $\lambda^{+} = \lambda e^{\beta}$, we see that e^{β} is the multiplicative effect of an increase in one unit to the predictor variable.

Example

```
doc_df <- read_csv(here('data/DoctorAUS.csv')) %>%
  mutate(gender = ifelse(sex == 1, 'female', 'male'))

M <- glm(doctorco ~ gender,
         data = doc_df,
         family = poisson)

doc_df_new <- tibble(gender = c('female', 'male'))

doc_df_new %>%
  add_predictions(M)

doc_df_new %>%
  add_predictions(M, type='response')
```

Model comparison

```
M_1 <- glm(doctorco ~ gender + insurance,
           data = doc_df,
           family = poisson)

anova(M, M_1, test='Chisq')
```

Exposure and offset

- In some problems, the length of time during which events are measured varies across individuals.
- In the doctor visits example, we might have recordings of number of visits per year for some people and number of visits per 9 months, etc, for others.
- These situations are dealt with using an *exposure* term for each individual.

Exposure and offset

- When using an exposure term, we use the original count data as before, but treat

$$y_i \sim \text{Poisson}(\lambda_i / u_i),$$

  where u_i is a term signifying the relative exposure time for person i.

- According to this,

$$\log(\lambda_i / u_i) = \alpha + \beta x_i,$$
$$\log(\lambda_i) = \alpha + \beta x_i + \log(u_i)$$

- In other words, $y_i \sim \text{Poisson}(\lambda_i / u_i)$ is equivalent to $y_i \sim \text{Poisson}(\lambda_i)$, where $\log(\lambda_i) = \alpha + \beta x_i + \log(u_i)$.

Exposure and offset

- For example, suppose we monitor people's drinking at social occasions. We find that three people drink 12, 7 and 3 drinks over the course of 7, 5 and 2 hours, respectively.
- If we are trying to predict drinking as a function of predictor variables, we ought to calibrate by the different time frames.
- Treating e.g. 12 as a draw from Poisson$(\lambda_i/7)$ where $\log(\lambda_i/7) = \alpha + \beta x_i$ is identical to treating 12 as a draw from Poisson(λ_i) where $\log(\lambda_i) = \alpha + \beta x_i + \log(7)$.

Exposure and offset

- In general, exposure terms are treated as fixed offsets.
- If our data is $(y_1, x_1), (y_2, x_2) \dots (y_n, x_n)$ with exposures $u_1, u_2 \dots u_n$, then we treat

$$y_i \sim \text{Poisson}(\lambda_i),$$

where

$$\log(\lambda_i) = \log(u_i) + \beta_0 + \sum_{k=1}^{K} \beta_k x_{ki}.$$

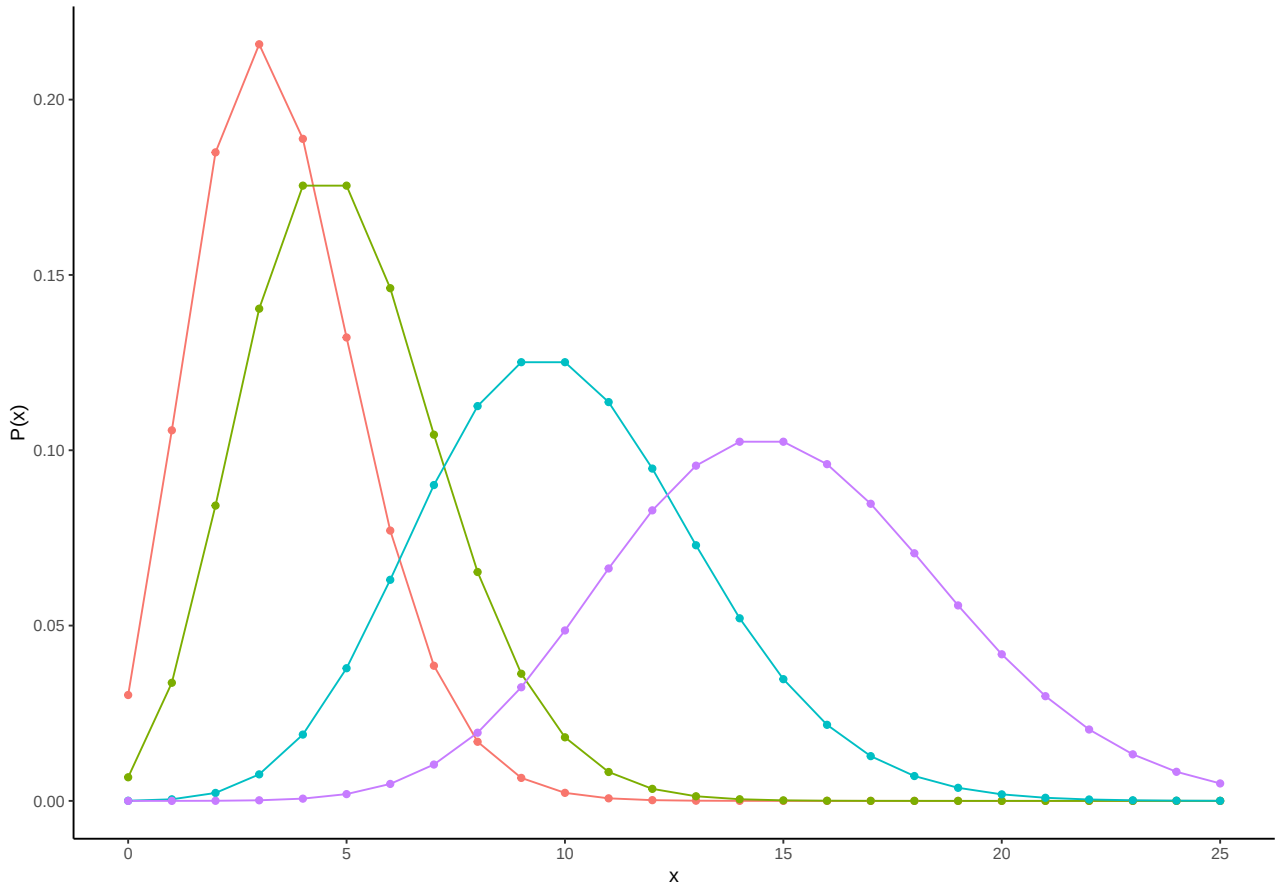
Example

```
insur_df <- read_csv(here('data/Insurance.csv')) %>%
  mutate(District = factor(District))

M <- glm(Claims ~ District + Group + Age + offset(log(Holders)),
         data = insur_df,
         family = poisson)
```

The Poisson Distribution

- The mean of a Poisson distribution is equal to its rate parameter λ.
- Its variance is also equal to λ.



As λ increases, so too does the variance.

Means and variances in a Poisson distribution:

- In a Poisson distribution, the variance of a sample should be approximately the same as the mean of a sample.
- Example 1:

```
x <- rpois(25, lambda = 5)
c(mean(x), var(x), var(x)/mean(x))
#> [1] 5.4400000 5.4233333 0.9969363
```

- Example 2:

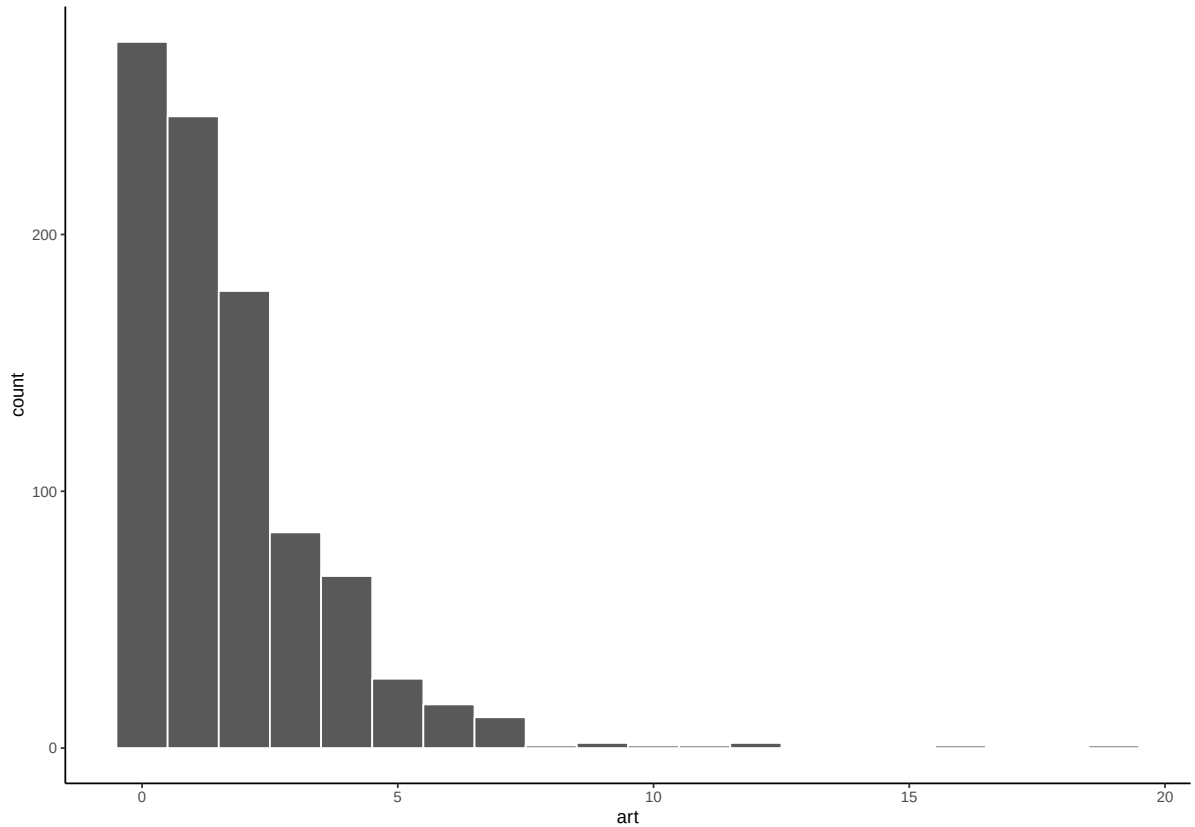
```
x <- rpois(25, lambda = 5)
c(mean(x), var(x), var(x)/mean(x))
#> [1] 5.400000 6.416667 1.188272
```

Overdispersion

- If the variance of a sample is greater than would be expected according to a given theoretical model, then we say the data is *overdispersed*.
- In count data, if the variance of a sample is much greater than its mean, we say it is overdispersed.
- Using a Poisson distribution in this situation, this is an example of model mis-specification.
- It will also usually underestimate the standard errors in the Poisson model.

Overdispersion

- In the `bioChemists` data set, we have counts of the number of articles published by PhD students in the last three years (`publications`):




```
var(publications)/mean(publications)
#> [1] 2.191358
```

Overdispersion

- This leads standard errors to be *under*estimated if we use a Poisson model:

```
M <- glm(publications ~ 1, family=poisson)
summary(M)$coefficients
#>                Estimate Std. Error  z value     Pr(>|z|)
#> (Intercept) 0.5264408 0.02540804 20.71945 2.312911e-95
```

Fixing overdispersion using a Quasi-poisson model

- A *quasi* Poisson model allows us to correct over-dispersion

```
M <- glm(publications ~ 1, family=quasipoisson)
summary(M)$coefficients
#>               Estimate Std. Error  t value     Pr(>|t|)
#> (Intercept) 0.5264408 0.03761239 13.99647 1.791686e-40
```

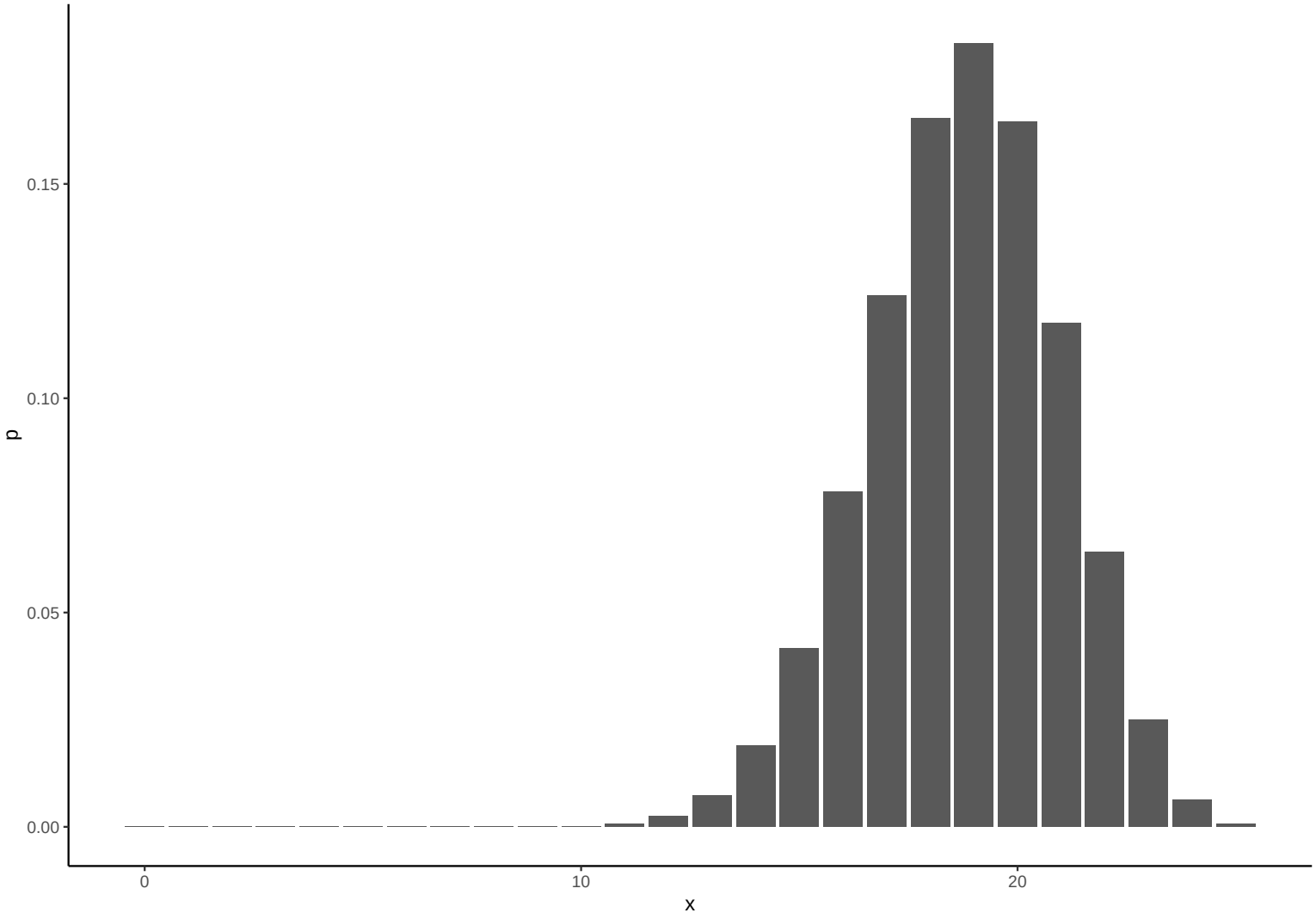
- It does so by calculating an overdispersion parameter (roughly, the ratio of the variance to the mean) and multiplying the standard error by its square root.
- In this example, the overdispersion parameter is 2.1913892 and so its square root is 1.4803341.
- Alternatively, a *negative binomial regression* is an alternative to Poisson regression that can be used with overdispersed count data.

Negative binomial distribution

- A negative binomial distribution is a distribution over non-negative integers.
- To understand the negative binomial distribution, we start with the binomial distribution:
- If, for example, we have a coin whose probability of coming up heads is θ, then the number of Heads in a sequence of n flips will follow a binomial distribution.
- In this example, an outcome of Heads can be termed a *success*.

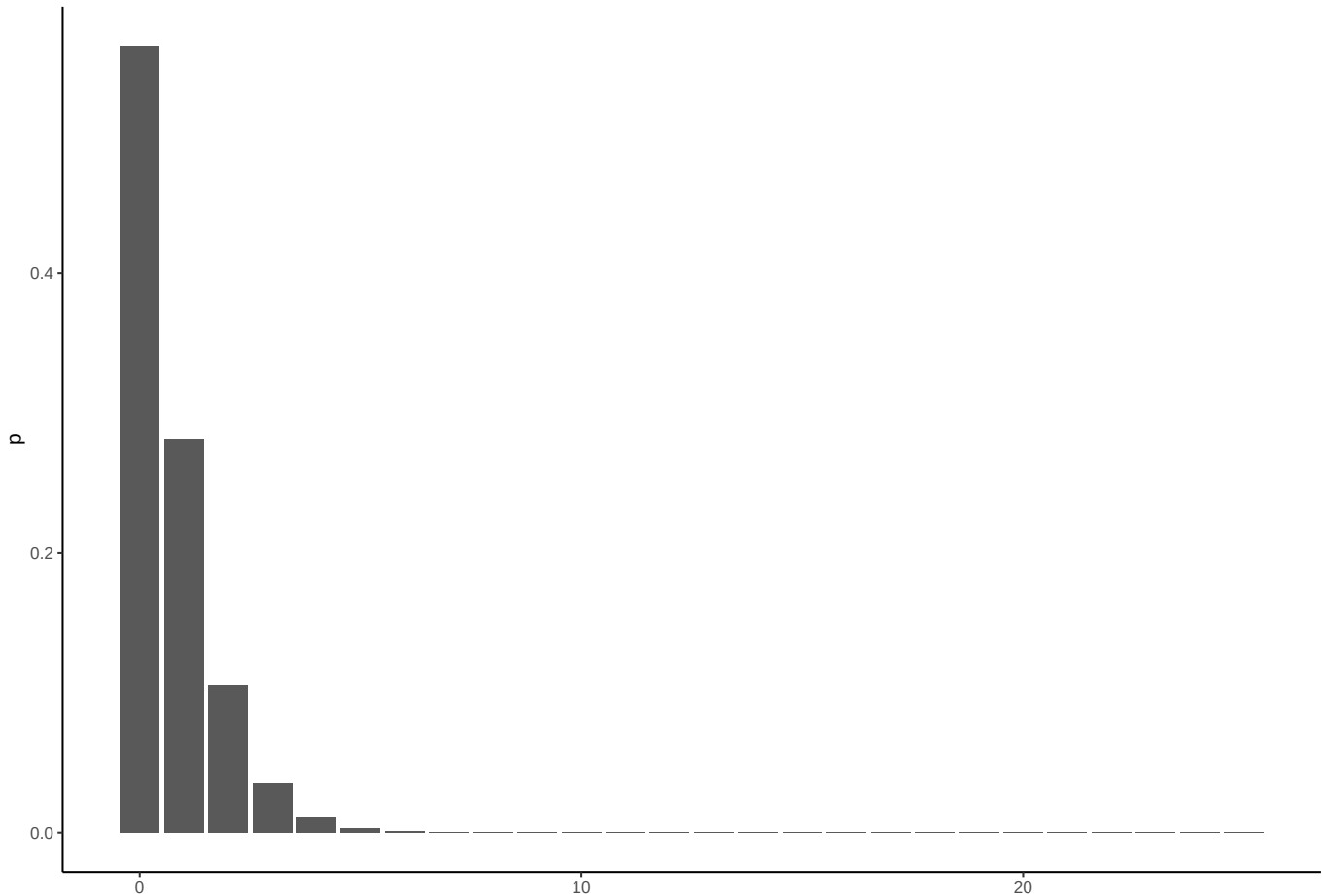
Negative binomial distribution

- Here is a binomial distribution where $n = 25$ and $\theta = 0.75$.



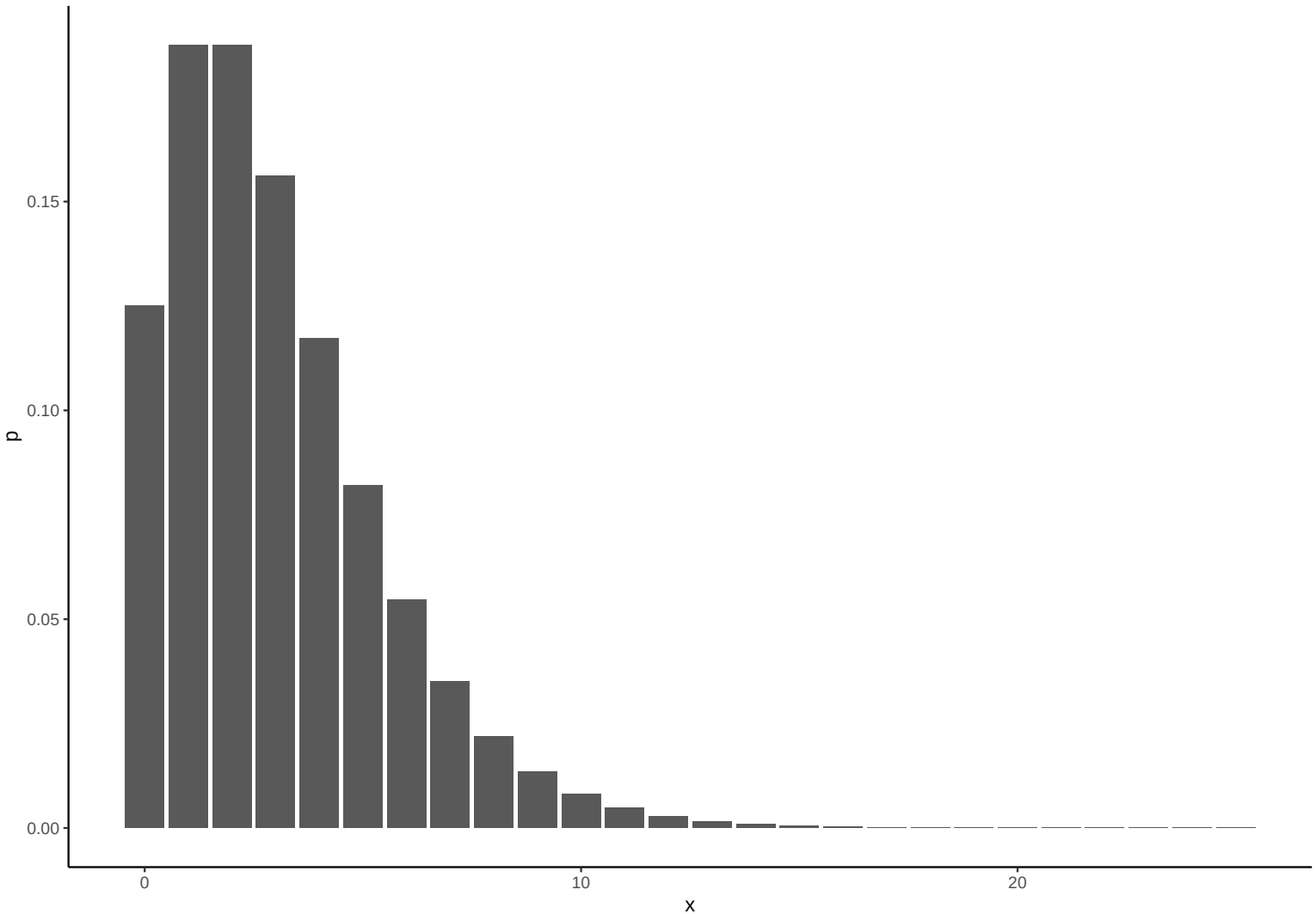
Negative binomial distribution

- A *negative* binomial distribution gives the probability distribution over the number of *failures* (e.g. Tails) before r *successes* (e.g. r Heads).
- For example, here we have the number of Tails (*failures*) that occur before we observe $r = 2$ Heads (*sucesses*), when the probability of Heads is $\theta = 0.75$:



Negative binomial distribution

- Here, we have the number of Tails (*failures*) that occur before we observe $r = 3$ Heads (*successes*), when the probability of Heads is $\theta = 0.5$:



Negative binomial distribution

- The probability mass function for the negative binomial distribution is:

$$P(x = k|r, \theta) = \binom{r + k - 1}{k} \theta^r (1 - \theta)^k$$

or more generally

$$P(x = k|r, \theta) = \frac{\Gamma(r + k)}{\Gamma(r)k!} \theta^r (1 - \theta)^k,$$

where $\Gamma()$ is a Gamma function ($\Gamma(n) = (n - 1)!$).

- In R, for any k, r, and θ, we can calculate $P(x = k|r, \theta)$ using `dnbinom`, e.g. $P(x = k = 2|r = 3, \theta = 0.75)$ is

```
dnbinom(2, 3, 0.75)
#> [1] 0.1582031
```

Negative binomial distribution

- In the negative binomial distribution, the mean is

$$\mu = \frac{\theta}{1-\theta} \times r,$$

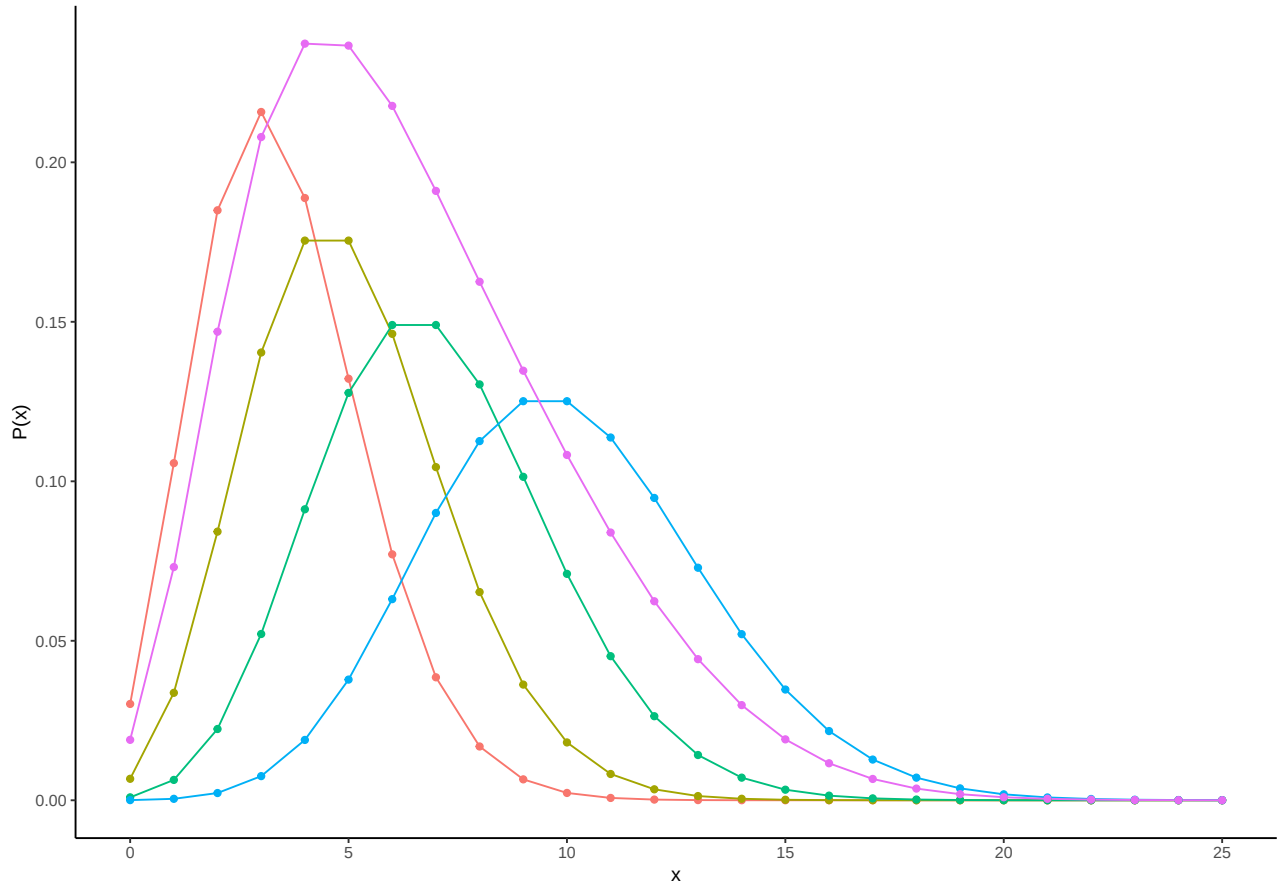
and so

$$\theta = \frac{r}{r+\mu},$$

and we can generally parameterize the distribution by μ and r.

Why use negative binomial distribution?

- A negative binomial distribution is equivalent as weighted sum of Poissons.



- So it is appropriate to use when the data can be seen as arising from a mixture of Poisson distributions, each with different means.

Negative binomial regression

- In negative binomial regression, we have observed counts $y_1, y_2 \dots y_n$, and some predictor variables $x_1, x_2 \dots x_n$, and we assume that

$$y_i \sim \text{NegBinomial}(\mu_i, r),$$

where $\text{NegBinomial}(\mu_i, r)$ is a negative binomial with mean μ_i and a dispersion parameter r, and then

$$\log(\mu_i) = \beta_0 + \beta x_i.$$

Example

```
M <- glm.nb(publications ~ gender, data = biochemists_Df)
M1 <- glm.nb(publications ~ gender + married + I(children > 0),
```

Binomial logistic regression

- If y_i is the number of "successes" in n_i "trials", we can model this as

$$y_i \sim \text{Binomial}(\theta_i, n_i),$$

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

- If $n_i = 1$ for all i, then this is exactly binary logistic regression.
- In general, it models the probability of something happening in a number of independent trials, and how the probability varies by the values of predictors.

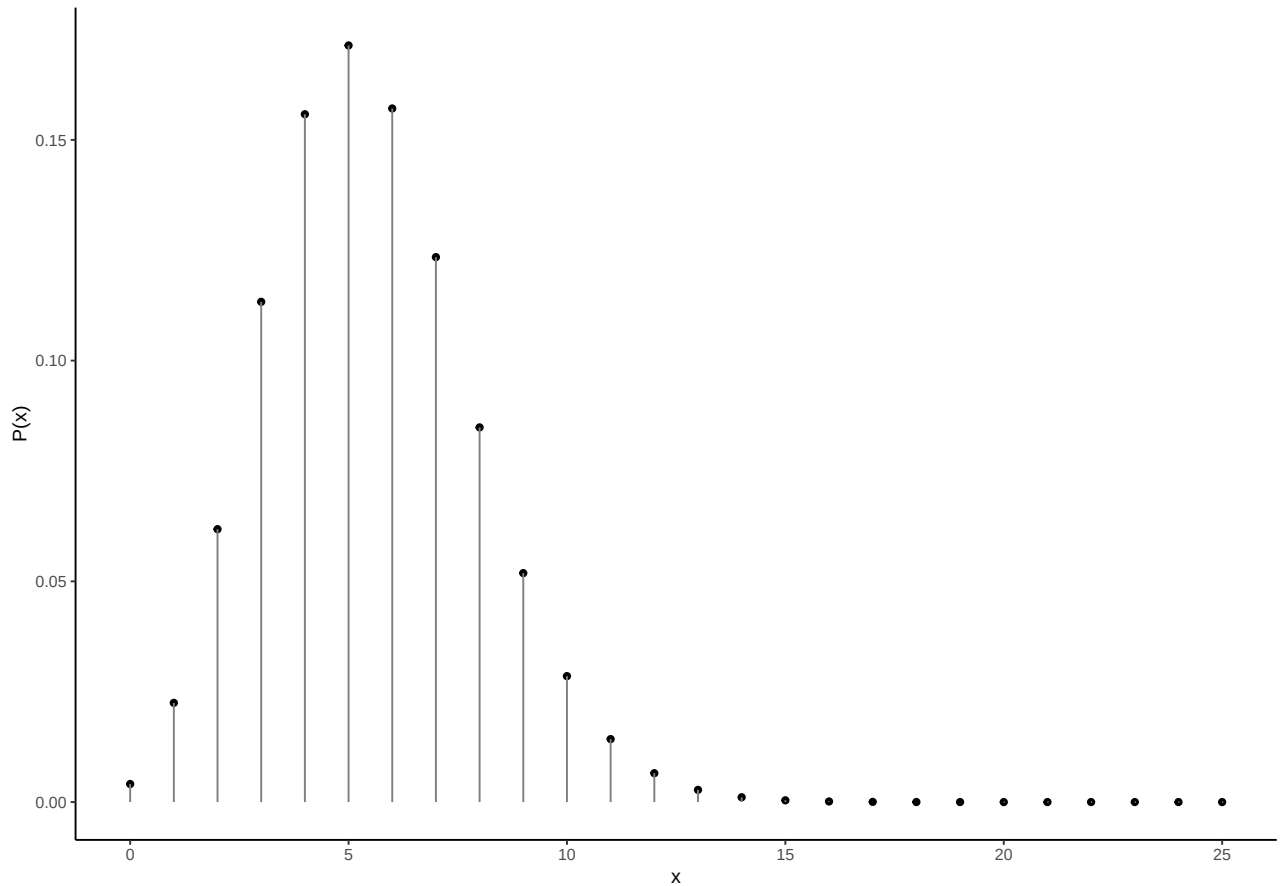
Example

```
golf_df <- read_csv(here('data/golf_putts.csv')) %>%
  mutate(failure = attempts - success,
         p = success/attempts)

M <- glm(cbind(success, failure) ~ distance,
         family = binomial(link = 'logit'),
         data = golf_df)
```

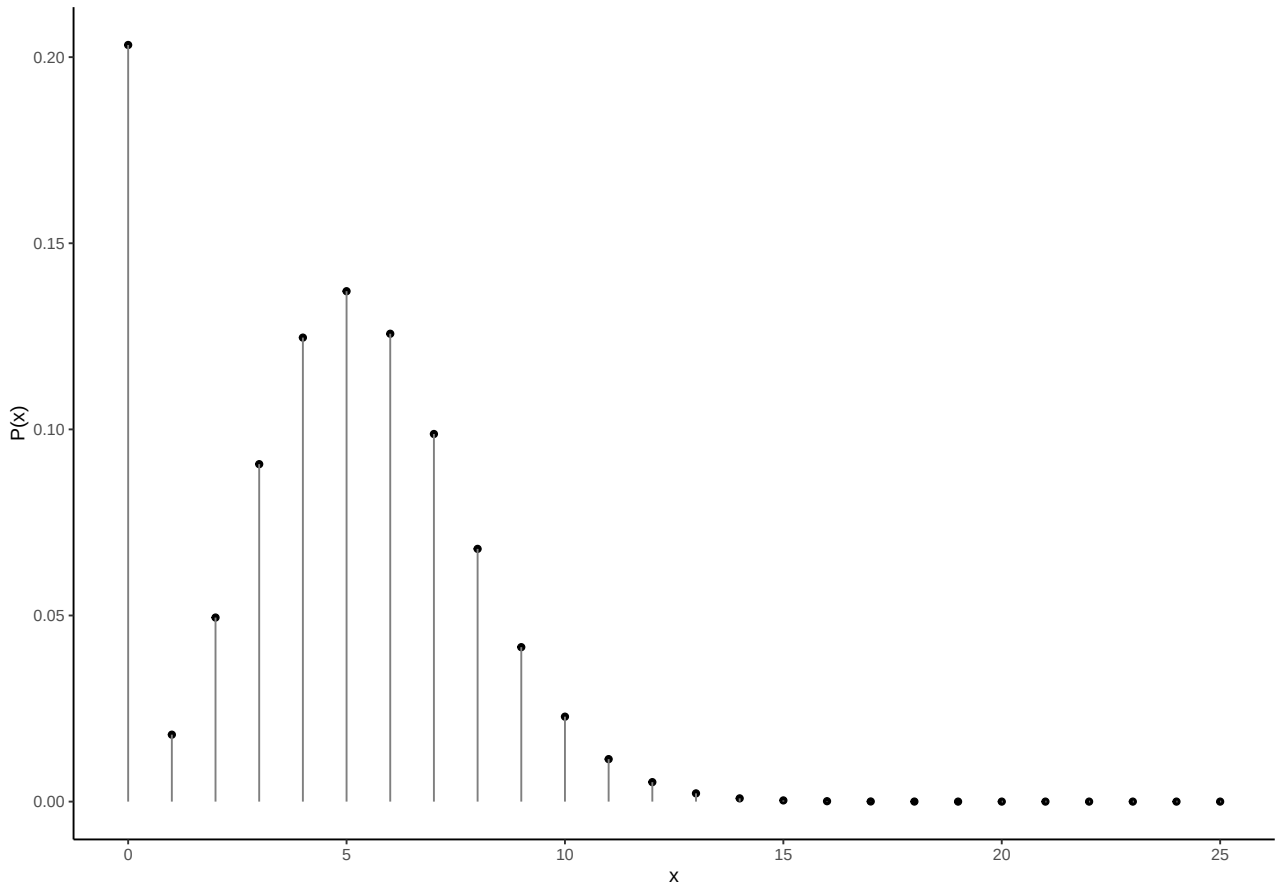
Poisson Distribution

A sample from a Poisson distribution with $\lambda = 5.5$.



Zero inflated Poisson Distribution

A sample from a zero inflated Poisson distribution with $\lambda = 5.5$, with probability of *zero-component* is 0.2.



Poisson regression to Zero-Inflated Poisson regression

- In Poisson regression (with a single predictor, for simplicity), we assume that each y_i is a Poisson random variable with rate λ_i that is a function of the predictor x_i.
- In Zero-Inflated Poisson regression, we assume that each y_i is distributed as a Zero-Inflated Poisson mixture model:

$$y_i \sim \begin{cases} \text{Poisson}(\lambda_i) & \text{if } z_i = 0, \\ 0, & \text{if } z_i = 1 \end{cases}$$

where rate λ_i and $P(z_i = 1)$ are functions of the predictor x_i.

Zero-Inflated Poisson regression

- Assuming data $\{(x_i, y_i), (x_2, y_2) \dots (x_n, y_n)\}$, Poisson regression models this data as:

$$y_i \sim \begin{cases} \text{Poisson}(\lambda_i) & \text{if } z_i = 0, \\ 0, & \text{if } z_i = 1 \end{cases},$$
$$z_i \sim \text{Bernoulli}(\theta_i),$$

  where θ_i and λ_i are functions of x_i.

Zero-Inflated Poisson regression

- The θ_i and λ_i variables are the usual suspects, i.e.

$$\log(\lambda_i) = \alpha + \beta x_i,$$

and

$$\log\left(\frac{\theta_i}{1-\theta_i}\right) = a + b x_i.$$

- In other words, λ_i is modelled just as in ordinary Poisson regression and θ_i is modelled in logistic regression.

Examples

```r
smoking_df <- read_csv(here('data/smoking.csv'))
M <- glm(cigs ~ educ, data = smoking_df)
M_zip <- zeroinfl(cigs ~ educ, data=smoking_df)

Df_new <- data.frame(educ = seq(20))
# Predited average smoking rate
Df_new %>%
  add_predictions(M_zip, type='response')

# Predicted average smoking rate of "smokers"
Df_new %>%
  add_predictions(M_zip, type='count')

# Predicted probability of being a "smoker"
Df_new %>%
  add_predictions(M_zip, type='zero') %>%
  mutate(pred = 1-pred)
```