Week Six

Last Week

- Contingency Tables
- Simpson's Paradox
- Fisher's Exact Test

This Week: Generalized Linear Models

Today:

- Activity:
  - Generative models for binary data
  - MLE for logistic regression
  - Bayesian estimation for logistic regression
- Thursday: Lab

Next Week: Generalized Linear Models: Binary Data

Logistic Regression

Recall the logistic regression framework, which satisfies the three elements of a GLM (random component, systematic component, link function)

$$
\begin{aligned}
y &\sim Bernoulli(\pi) \\
\pi &= \frac{\exp(\beta_0 + \beta_1 x)}{1 + \exp(\beta_0 + \beta_1 x)} \\
\pi &= logit^{-1}(\beta_0 + \beta_1 x)
\end{aligned}
$$

Logistic Regression Activity: Continuous Predictor

We are going to focus on the generative process we assume underlies logistic regression (with a single continuous covariate).

1. Simulate 100 covariate values. This isn't necessary, but assume they are equally spaced between -3 and 3.

```
n <- 100
x <- seq(-3, 3, length.out = n)
```

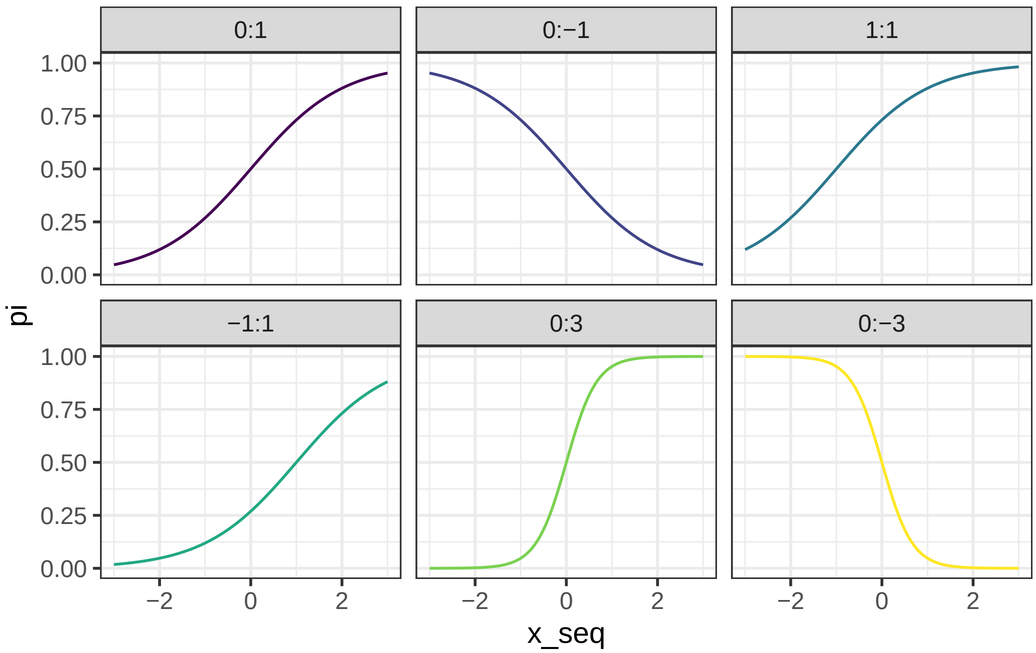
2. The β values will change the shape of our expected relationship. Using the following values below, create figures of π vs x.

   i. $\beta_0 = 0$, $\beta_1 = 1$
   ii. $\beta_0 = 0$, $\beta_1 = -1$
   iii. $\beta_0 = 1$, $\beta_1 = 1$
   iv. $\beta_0 =$ -1, $\beta_1 = 1$
   v. $\beta_0 = 0$, $\beta_1 = 3$
   vi. $\beta_0 = 0$, $\beta_1 = -3$

```
pi_table <- tibble(x_seq = rep(x, 6),
       pi = c(invlogit(0 + 1 * x),
              invlogit(0 - 1 * x),
              invlogit(1 + 1 * x),
              invlogit(-1 + 1 * x),
              invlogit(0 + 3 * x),
              invlogit(0 - 3 * x)),
       label = ordered(rep(c('0:1', '0:-1','1:1', '-1:1','0:3','0:-3'), each = n),
```

```
                             levels =c('0:1', '0:-1','1:1', '-1:1','0:3','0:-3') ))

pi_table |>
  ggplot(aes(y = pi, x = x_seq, color = label)) +
  geom_line() +
  facet_wrap(.~label) +
  theme_bw() +
  theme(legend.position = 'none')
```



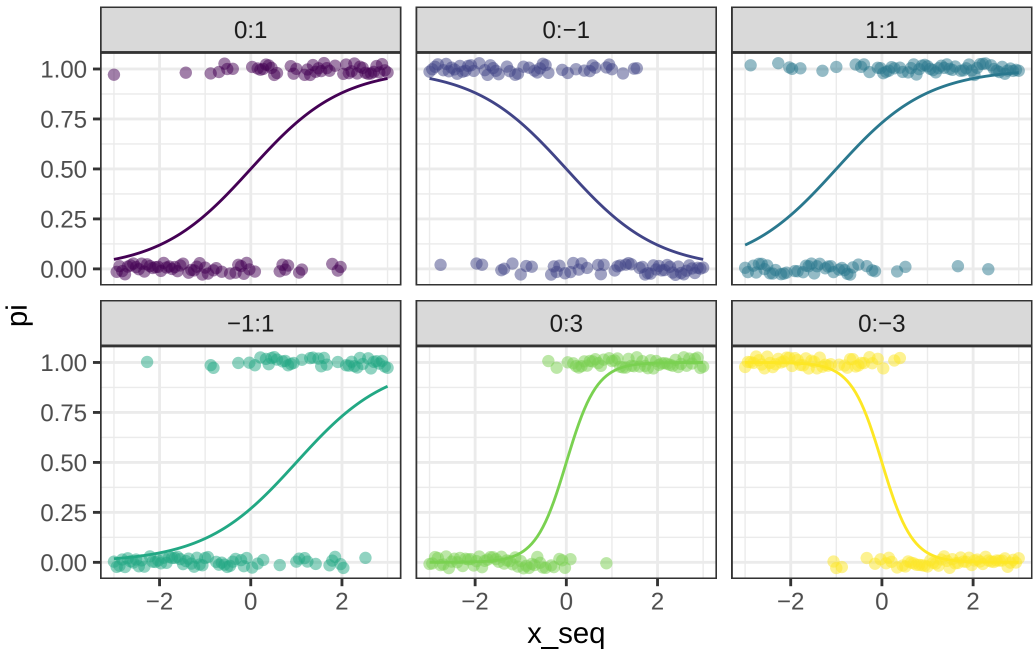
3. Based on the figure provide an intuitive summary of how β_0 and β_1 impact the curve.

- β_0: This controls the probability of success (when x = 0), larger values result in an increased probability of success and smaller value reduce the probability of success.
- β_1: This controls how quickly the probability of success changes (with respect to x). Positive values have an increasing probability of success as x increases. Larger values, in absolute terms, result in a steeper curve.
- $\beta_0 + \beta_1 x$: when the sum of these two values gets large (>2) or small (< -2), the probability of success approaches 1, or zero respectively.

4. Simulate a binary outcome at each x value. Update the figure from part to to include these data points.

```
y_table <- pi_table |> mutate(y = rbinom(n*6,1, pi))

y_table |>
  ggplot(aes(y = pi, x = x_seq, color = label)) +
  geom_line() +
  geom_jitter(aes(y = y, x = x_seq, color = label ),
              inherit.aes = F, alpha = .5, height = .03, width = 0) +
  facet_wrap(.~label) +
  theme_bw() +
  theme(legend.position = 'none')
```



5. Use MLE to estimate the coefficients in each of these six settings. Report point estimates and uncertainty. You'll want to use the following formulation `glm(y~x, family = binomial, data =)`.

```
#'0:1', '0:-1','1:1', '-1:1','0:3','0:-3'
y_table |>
  filter(label == '0:1') %>%
  glm(y~x, data = ., family = binomial) |>
  display()
```

```
glm(formula = y ~ x, family = binomial, data = .)
```

```
             coef.est coef.se
(Intercept) -0.22     0.27
x            1.11     0.21
---
  n = 100, k = 2
  residual deviance = 86.9, null deviance = 138.3 (difference = 51.4)
```

```r
y_table |>
  filter(label == '0:-1') %>%
  glm(y~x, data = ., family = binomial) |>
  display()
```

```
glm(formula = y ~ x, family = binomial, data = .)
            coef.est coef.se
(Intercept) -0.13     0.25
x           -0.91     0.18
---
  n = 100, k = 2
  residual deviance = 97.9, null deviance = 138.5 (difference = 40.6)
```

```r
y_table |>
  filter(label == '1:1') %>%
  glm(y~x, data = ., family = binomial) |>
  display()
```

```
glm(formula = y ~ x, family = binomial, data = .)
            coef.est coef.se
(Intercept) 0.65     0.29
x           1.12     0.21
---
  n = 100, k = 2
  residual deviance = 85.2, null deviance = 135.4 (difference = 50.2)
```

```r
y_table |>
  filter(label == '-1:1') %>%
  glm(y~x, data = ., family = binomial) |>
  display()
```

```
glm(formula = y ~ x, family = binomial, data = .)
            coef.est coef.se
```

```
(Intercept) -0.67     0.28
x            1.00     0.19
---
  n = 100, k = 2
  residual deviance = 90.7, null deviance = 134.6 (difference = 43.9)
```

```r
y_table |>
  filter(label == '0:3') %>%
  glm(y~x, data = ., family = binomial) |>
  display()
```

```
glm(formula = y ~ x, family = binomial, data = .)
            coef.est coef.se
(Intercept) 0.00     0.50
x           4.15     1.15
---
  n = 100, k = 2
  residual deviance = 26.1, null deviance = 138.6 (difference = 112.5)
```

```r
y_table |>
  filter(label == '0:-3') %>%
  glm(y~x, data = ., family = binomial) |>
  display()
```

```
glm(formula = y ~ x, family = binomial, data = .)
            coef.est coef.se
(Intercept) -0.48     0.41
x           -2.64     0.59
---
  n = 100, k = 2
  residual deviance = 41.0, null deviance = 138.3 (difference = 97.3)
```

6. Use Bayesian estimation for the coefficients in each of these six settings. Report point estimates and uncertainty. You'll want to use the following formulation `stan_glm(y~x, family = binomial, refresh = 0, data =)` which is the `rstanarm` package. *Note this has a weakly informative prior distribution embedded in the function.*

```r
#'0:1', '0:-1','1:1', '-1:1','0:3','0:-3'
y_table |>
  filter(label == '0:1') %>%
  stan_glm(y~x, data = ., family = binomial, refresh = 0) |>
  print(digits = 2)
```

```
stan_glm
 family:       binomial [logit]
 formula:      y ~ x
 observations: 100
 predictors:   2
------
            Median MAD_SD
(Intercept) -0.21   0.28
x            1.12   0.21

------
* For help interpreting the printed output see ?print.stanreg
* For info on the priors used see ?prior_summary.stanreg
```

```r
y_table |>
  filter(label == '0:-1') %>%
  stan_glm(y~x, data = ., family = binomial, refresh = 0) |>
  print(digits = 2)
```

```
stan_glm
 family:       binomial [logit]
 formula:      y ~ x
 observations: 100
 predictors:   2
------
            Median MAD_SD
(Intercept) -0.13   0.25
x           -0.92   0.18

------
* For help interpreting the printed output see ?print.stanreg
* For info on the priors used see ?prior_summary.stanreg
```

```r
y_table |>
  filter(label == '1:1') %>%
  stan_glm(y~x, data = ., family = binomial, refresh = 0) |>
  print(digits = 2)
```

```
stan_glm
 family:       binomial [logit]
 formula:      y ~ x
```

```
 observations: 100
 predictors:   2
------
            Median MAD_SD
(Intercept) 0.65   0.28  
x           1.12   0.21  

------
* For help interpreting the printed output see ?print.stanreg
* For info on the priors used see ?prior_summary.stanreg
```

```r
y_table |>
  filter(label == '-1:1') %>%
  stan_glm(y~x, data = ., family = binomial, refresh = 0) |>
  print(digits = 2)
```

```
stan_glm
 family:       binomial [logit]
 formula:      y ~ x
 observations: 100
 predictors:   2
------
            Median MAD_SD
(Intercept) -0.67   0.27 
x            1.01   0.19 

------
* For help interpreting the printed output see ?print.stanreg
* For info on the priors used see ?prior_summary.stanreg
```

```r
y_table |>
  filter(label == '0:3') %>%
  stan_glm(y~x, data = ., family = binomial, refresh = 0) |>
  print(digits = 2)
```

```
stan_glm
 family:       binomial [logit]
 formula:      y ~ x
 observations: 100
 predictors:   2
------
```

```
            Median MAD_SD
(Intercept) 0.01   0.44  
x           3.16   0.65  

------
* For help interpreting the printed output see ?print.stanreg
* For info on the priors used see ?prior_summary.stanreg
```

```r
y_table |>
  filter(label == '0:-3') %>%
  stan_glm(y~x, data = ., family = binomial, refresh = 0) |>
  print(digits = 2)
```

```
stan_glm
 family:       binomial [logit]
 formula:      y ~ x
 observations: 100
 predictors:   2
------
            Median MAD_SD
(Intercept) -0.42   0.39 
x           -2.43   0.47 

------
* For help interpreting the printed output see ?print.stanreg
* For info on the priors used see ?prior_summary.stanreg
```

7. How do values from parts 6 and 7 compare with each other? Do the values match your expectation?

The values are quite similar across both estimates. They are largely what I'd expect with just a 100 samples, if n was much larger then they'd likely be closer to the true values (with more precise intervals).

Logistic Regression Activity: Binary Predictor

Now let's consider a data structure that we've already seen, one binary predictor and one binary covariate.

There are two formulations of this model, the first is known as the reference case model.

$$
\begin{aligned}
y &\sim Bernoulli(\pi) \\
\pi &= \frac{\exp(\beta_0 + \beta_1 I_{x=1})}{1 + \exp(\beta_0 + \beta_1 I_{x=1})} \\
\pi &= logit^{-1}(\beta_0 + \beta_1 I_{x=1})
\end{aligned}
$$

The second is the cell means model

$$
\begin{aligned}
y &\sim Bernoulli(\pi) \\
\pi &= \frac{\exp(\beta_0 I_{x=0} + \beta_1 I_{x=1})}{1 + \exp(\beta_0 I_{x=0} + \beta_1 I_{x=1})} \\
\pi &= logit^{-1}(\beta_0 I_{x=0} + \beta_1 I_{x=1})
\end{aligned}
$$

what is the difference?

Let's repeat similar steps to the continuous setting. Use the cell means formulation for this question.

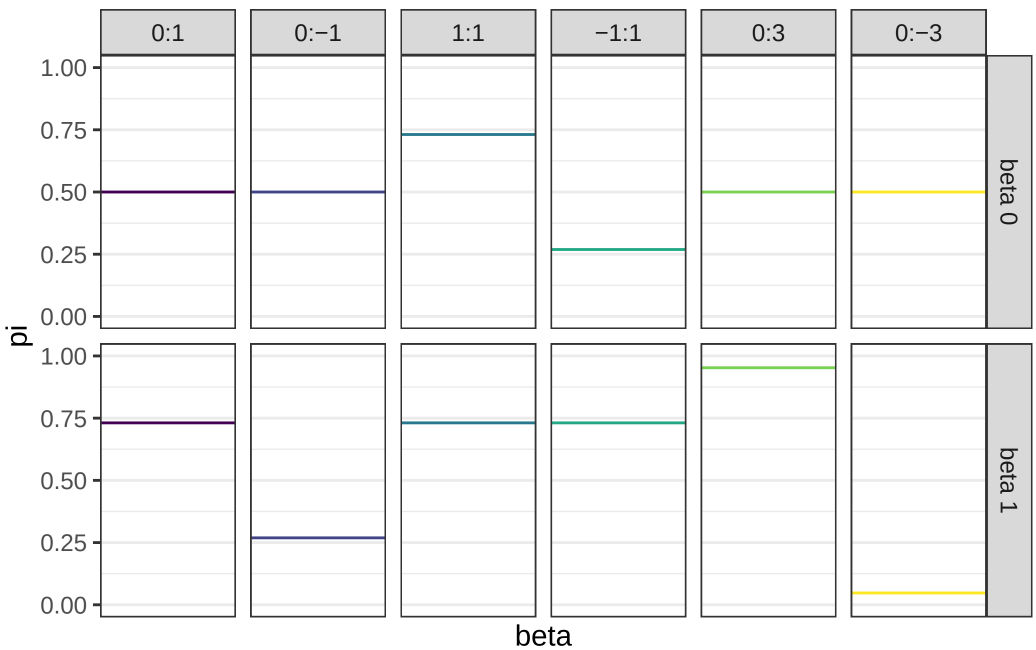
1. Let there be a total of 100 observations, 50 from $x = 1$ and 50 from $x = 2$
2. The β values will change our expected relationship. Using the following values below, create figures of π vs x.

   i. $\beta_0 = 0$, $\beta_1 = 1$
   ii. $\beta_0 = 0$, $\beta_1 = -1$
   iii. $\beta_0 = 1$, $\beta_1 = 1$
   iv. $\beta_0 =$ -1, $\beta_1 = 1$
   v. $\beta_0 = 0$, $\beta_1 = 3$
   vi. $\beta_0 = 0$, $\beta_1 = -3$

```
x_disc <- rep(c(0, 1), each = 50)
pi_table <- tibble(x_vals = rep(x_disc, 6),
       pi = c(invlogit(0 * as.numeric(x_disc==0) + 1 * as.numeric(x_disc==1)),
              invlogit(0 * as.numeric(x_disc==0)  - 1 * as.numeric(x_disc==1)),
              invlogit(1 * as.numeric(x_disc==0)  + 1 * as.numeric(x_disc==1)),
              invlogit(-1 * as.numeric(x_disc==0)  + 1 * as.numeric(x_disc==1)),
              invlogit(0 * as.numeric(x_disc==0)  + 3 * as.numeric(x_disc==1)),
              invlogit(0 * as.numeric(x_disc==0)  -3 * as.numeric(x_disc==1))),
```

```
       label = ordered(rep(c('0:1', '0:-1','1:1', '-1:1','0:3','0:-3'), each = n),
                       levels =c('0:1', '0:-1','1:1', '-1:1','0:3','0:-3') ))

pi_table |>
  mutate(beta = case_when(
           x_vals == 0 ~ 'beta 0',
           x_vals== 1 ~ 'beta 1'
         )) |>
  ggplot(aes(y = pi, x = beta, color = label)) +
  geom_hline(aes(yintercept = pi, color = label)) +
  facet_grid(beta~label) +
  theme_bw() +
  theme(legend.position = 'none') +
  ylim(0,1)
```



4. Simulate a binary outcome at each x value. Update the figure from part to to include these data points.

```
y_table <- pi_table |> mutate(y = rbinom(n*6,1, pi))

y_table |>
  mutate(beta = case_when(
```

```
          x_vals == 0 ~ 'beta 0',
          x_vals== 1 ~ 'beta 1'
        )) |>
  ggplot(aes(y = pi, x = beta, color = label)) +
  geom_hline(aes(yintercept = pi, color = label)) +
  facet_grid(beta~label) +
  theme_bw() +
  theme(legend.position = 'none',
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank()) +
  #ylim(0,1) +
  geom_jitter(aes(y = y, x = beta, color = label ),
              inherit.aes = F, alpha = .5, height = .03, width = 1.25)
```