SOC-GA 2332 Intro to Stats Lab 11

Risa Gelles-Watnick

11/14/2025

Part 0: Logistics

- **Problem Set #3** is due next Friday (11/21 @ 11:59pm)
- Agenda
  - Contingency tables
  - Chi-squared test
  - Ordered logistic regression
  - Multinomial logistic regression
  - Conditional logistic regression
- Next lab we will go over concepts from the longitudinal data analysis slides

Part 1: Bi-variate Associations (Contingency Tables)

For today, we will use a dataset about same-sex marriage support. This dataset has three support levels (1 = Oppose, 2 = Neutral, 3 = Support) instead of a binary outcome.

```
# explore the dataset
psych::describe(support_df)
```

```
##                vars    n  mean   sd median trimmed  mad min max range  skew
## support_level*    1 1000  2.26 0.84      3    2.33 0.00   1   3     2 -0.52
## eduy              2 1000 12.04 3.43     12   11.91 2.97   3  24    21  0.35
## age               3 1000 39.86 6.16     40   39.78 5.93  20  64    44  0.19
## female            4 1000  0.49 0.50      0    0.49 0.00   0   1     1  0.04
## black             5 1000  0.29 0.46      0    0.24 0.00   0   1     1  0.90
##                kurtosis   se
## support_level*    -1.37 0.03
## eduy               0.19 0.11
## age                0.23 0.19
## female            -2.00 0.02
## black             -1.19 0.01
```

Question: What is a contingency table? What is it used for?

A contingency table shows the frequency distribution between two categorical variables (i.e. how many values are in each cell in a table comparing the two variables). These tables are used for showing how two variables are related to each other and whether they're independent.

In `R`, you can create a contingency table by using the `table()` function and input the two categorical variables you are interested in. To conduct a chi-square test of independence, simply use the function `chisq.test(your_contingency_table)`.

```
## create variables for contingency tables
support_df <- support_df %>%
```

```
  mutate(## convert dummies to categorical variables
         gender = ifelse(female == 0, "male", "female"),
         race = ifelse(black == 1, "black", "white"))

## simple contingency table and chi-square test for support levels and race
t1 <- table(support_df$support_level, support_df$race)
t1
```

```
##
##     black white
##   1    72   180
##   2    74   161
##   3   148   365
```

```
chisq.test(t1)
```

```
##
##  Pearson's Chi-squared test
##
## data:  t1
## X-squared = 0.65239, df = 2, p-value = 0.7217
```

```
# can do this same test with the janitor package (works better with tidyr and provides more customizati
support_df %>%
  janitor::tabyl(support_level, race) %>%
  janitor::chisq.test()
```

```
##
##  Pearson's Chi-squared test
##
## data:  .
## X-squared = 0.65239, df = 2, p-value = 0.7217
```

Question: What is the Chi-squared test testing for? What is the null and alternative hypotheses?

The Chi-squared test is testing for whether two categorical variables are statistically independent. The null hypothesis is that the variables are independent.

Question: What conclusion do we draw from this Chi-squared test?

We can reject the null hypothesis that the variables are independent.

Question: If support for same-sex marriage is dependent on race, does this mean race is dependent on support for same-sex marriage?

Yes! We are using dependent/independent in a different way than when we are talking about regressions. In this context, variables being "dependent" just means that they have some association with each other. So if one variable is dependent on another, the opposite (by definition) is also true.

Quick R tangent

Notice that both the `janitor` package and base R (`stats` package) have a `chisq.test()` command. There are many packages that have commadns that share a name (that don't always do the same thing!). Make sure you know which package R is drawing commands from.

A simple way to specify the package you want to call the command from is with the double colon `::`. Another way to do this is by using the `conflict_prefer()` command from the `conflicted` package to globally set which package you want R to draw a certain command from.

For example:

```r
# specifying to use chisq.test() command from the janitor package
support_df %>%
  janitor::tabyl(support_level, race) %>%
  janitor::chisq.test()
```

```
##
##  Pearson's Chi-squared test
##
## data:  .
## X-squared = 0.65239, df = 2, p-value = 0.7217
```

```r
# telling R to always (in this script) use the chisq.test() command from the base R stats package
conflicted::conflict_prefer("chisq.test", "stats")
```

```
## [conflicted] Removing existing preference.
## [conflicted] Will prefer stats::chisq.test over any other package.
```

End R tangent

Question: What are the three types of distributions you can find in contingency tables? Joint, Marginal, and Conditional

We can make each of these types of contingency tables in R with the `janitor::tabyl()` command, modified with the `adorn` functions.

- **Joint distribution:** calculates probabilities of events happening together (i.e. what percent of the data is in each cell). For example, what percent of the data is Black people who support same-sex marriage? Remember, 3 is support.

```r
support_df %>%
  tabyl(support_level, race) %>%
  adorn_percentages("all") %>% # add row and column percentages (i.e. percentage of data in each cell)
  adorn_totals("row") %>% # add a total row
  adorn_ns() # add Ns to table
```

```
##  support_level       black       white
##              1 0.072  (72) 0.180 (180)
##              2 0.074  (74) 0.161 (161)
##              3 0.148 (148) 0.365 (365)
##          Total 0.294 (294) 0.706 (706)
```

14.8% of the data is Black people who support same-sex marraige.

- **Marginal distribution:** calculates probabilities of individual events (i.e. what percentage of the data is in each row or column). For example, how many people support same-sex marriage?

```r
support_df %>%
  tabyl(support_level)
```

```
##  support_level   n percent
##              1 252   0.252
##              2 235   0.235
##              3 513   0.513
```

51.3% of people support same-sex marriage.

- **Conditional distribution:** calculates probabilities of events given specific conditions (i.e. what percentage of a given row is in a column, or vice versa). For example, what percentage of Black people support same-sex marriage?

```r
support_df %>%
  tabyl(support_level, race) %>%
  adorn_percentages("col") %>% # add column percentages (i.e. percentage of column in each row)
  adorn_totals("row") %>% # add a total row
  adorn_ns() # add Ns to table
```

```
##  support_level           black           white
##              1 0.2448980  (72) 0.2549575 (180)
##              2 0.2517007  (74) 0.2280453 (161)
##              3 0.5034014 (148) 0.5169972 (365)
##          Total 1.0000000 (294) 1.0000000 (706)
```

50.3% of Black people support same-sex marriage.

Another way to think about independence is that if the product of the marginal distributions of each variable equal their joint distribution, then the two variables are independent (with matrix multiplication). This is essentially what the chi-squared test is testing.

Part 1 Exercise (10 minutes)

Recall that the χ^2 statistic is defined as:

$$\chi^2 = \sum \frac{(f^o - f^e)^2}{f^e},$$

where f^o is the observed frequency and f^e is the expected frequency.

You are given the following contingency table of support levels and gender:

```
   Cell Contents
|-------------------------|
|                       N |
|              Expected N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  1000

                         | support_df$gender
support_df$support_level |    female |      male | Row Total |
-------------------------|-----------|-----------|-----------|
                       1 |       105 |       147 |       252 |
                         |   123.228 |   128.772 |           |
                         |     0.417 |     0.583 |     0.252 |
                         |     0.215 |     0.288 |           |
                         |     0.105 |     0.147 |           |
-------------------------|-----------|-----------|-----------|
                       2 |       109 |       126 |       235 |
                         |   114.915 |   120.085 |           |
                         |     0.464 |     0.536 |     0.235 |
                         |     0.223 |     0.247 |           |
                         |     0.109 |     0.126 |           |
-------------------------|-----------|-----------|-----------|
                       3 |       275 |       238 |       513 |
```

```
                         |   250.857 |   262.143 |           |
                         |     0.536 |     0.464 |     0.513 |
                         |     0.562 |     0.466 |           |
                         |     0.275 |     0.238 |           |
-------------------------|-----------|-----------|-----------|
            Column Total |       489 |       511 |      1000 |
                         |     0.489 |     0.511 |           |
-------------------------|-----------|-----------|-----------|
```

1. How do you calculate the expected frequency for each cell? Verify your answer with the expected frequencies in the table.

Expected frequency is the N you would expect in each cell if there was truly no relationship between the two variables. This is calculated by calculating (row total * column total)/total N for each cell.

2. State your null and alternative hypotheses of the χ^2 test;

Null: there is no relationship between gender and level of support for same-sex marriage (they are independent)

Alternative: gender and level of support for same-sex marriage are dependent on each other.

3. Calculate the χ^2 statistic using the formula above;

$$\chi^2 = \sum \frac{(f^o - f^e)^2}{f^e}$$

$$\chi^2 = \frac{(105 - 123.228)^2}{123.228} + \frac{(147 - 128.772)^2}{128.772} + \frac{(109 - 114.915)^2}{114.915} + \frac{(126 - 120.085)^2}{120.085} + \frac{(275 - 250.857)^2}{250.857} + \frac{(238 - 262.143)^2}{262.143}$$

```
(((105 - 123.228)^2)/(123.228)) + (((147 - 128.772)^2)/(128.772)) + (((109 - 114.915)^2)/(114.915)) + (
```

```
## [1] 10.41945
```

$$\chi^2 = 10.41945$$

4. Calculate the p-value of your test statistic. *Hint*: (a) recall that the degree of freedom is calculated by $\text{df} = (\text{nrow} - 1) \cdot (\text{ncol} - 1)$, (b) search `pchisq`

$$\text{df} = (3 - 1) \cdot (2 - 1) = 2$$

```
pchisq(10.41945, 2, lower.tail = FALSE)
```

```
## [1] 0.005463176
```

We can check our answer with the R command:

```
support_df %>%
  janitor::tabyl(support_level, gender) %>%
  janitor::chisq.test()
```

```
##
##  Pearson's Chi-squared test
##
## data:  .
## X-squared = 10.419, df = 2, p-value = 0.005463
```

We get the same answer, woohoo!

Part 2: Ordered Logit Regression Model

2.1 Model Setup

- *Question:* When do we use an ordered logistic regression? When we have a categorical outcome variable that has an inherent order to it (ex. education level)
- Think back to when we were talking about p-values. We said one way of defining a p-value is the value of the cumulative probability function of the null t-distribution from your observed t-statistic to infinity. In other words, the total probability under the null t-distribution from your t-statistic to infinity (times 2, for a two-tailed p-value).
- For the ordered logit regression model, we're still using a cumulative probability function, but this time on a logistic distribution rather than a t-distribution.
- The cumulative probability for individual i's choice up to response level j is given by:

$$C_{i,j} = Pr(y_i \leq j) = \sum_{k=1}^{j} Pr(y_i = k) = \frac{1}{1 + exp(-(\phi_j - x_i\beta))} j = 1, 2, ..., J$$

- y_i is the outcome for individual i
- j is the category you're interested in. So $Pr(y_i \leq j)$ is the probability that individual i is in a category lower than or equal to j
- J is the "top" category in your data
- k is an index that assigns a number to each category. So summing from k = 1 to j runs through every category "below" and including j
- x_i is a vector of variables that affect y
- β is a vector of coefficients that go with those variables
- ϕ_j is the cutpoint or threshold between cumulative probabilities, i.e. the point in the cumulative probability distribution that separates category j from category j + 1
- So in sum, this equation is looking at how much probability there is that an individual i falls into a category j or lower
- This specific form of cumulative probability stems from the Sigmoid function:

$$f(x) = \frac{1}{1 + exp(-x)}$$

which is monotonically increasing with reference to x

- Here we replace x by the linear combination of category-specific cutpoints ϕ_j and individual-specific characteristics and their coefficients
  - Intuitively, we would use $\frac{1}{1+exp(-(\phi_j+x_i\beta))}$ as in binary logistic model
  - We use $\frac{1}{1+exp(-(\phi_j-x_i\beta))}$ because the model was specified in this way at the time of invention – path dependence
  - It only changes the sign of β, but not its magnitude
- As the Sigmoid function is monotonically increasing, we will have:
  - $\phi_0 < \phi_1 < ... < \phi_J$
  - ϕ_0 to be $-\infty$ and ϕ_J to be ∞.
- The probability of being in response category j for the same individual i is:

$$
\begin{aligned}
Pr(y_i = j) &= C_{i,j} - C_{i,j-1} \\
&= Pr(y_i \leq j) - Pr(y_i \leq j-1) \\
&= \frac{1}{1 + exp(-\phi_j + x_i\beta)} - \frac{1}{1 + exp(-\phi_{j-1} + x_i\beta)}
\end{aligned}
$$

- ϕ_j and β are estimated using Maximum Likelihood Estimation (MLE). MLE basically tries to find the parameters of a function that maximize the probability of seeing your observed data.
- In R, you can estimate a ordered logit model using the `polr()` function from the `MASS` package.

```
## estimate ordered logit model
ologit1 <- polr(support_level ~ eduy, data = support_df, method="logistic")
ologit2 <- polr(support_level ~ eduy + age, data = support_df, method="logistic")
ologit3 <- polr(support_level ~ eduy + age + female, data = support_df, method="logistic")
ologit4 <- polr(support_level ~ eduy + age + female + black, data = support_df, method="logistic")

## stargazer
stargazer(ologit1, ologit2, ologit3, ologit4, type="text")
```

```
##
## ================================================
##                         Dependent variable:
##              -------------------------------------
##                            support_level
##                 (1)        (2)        (3)        (4)
## ------------------------------------------------
## eduy         0.639*** 0.817***  0.864***  0.864***
##              (0.034)   (0.043)   (0.045)   (0.045)
##
## age                    -0.221*** -0.227*** -0.227***
##                         (0.016)   (0.016)   (0.016)
##
## female                            1.041***  1.044***
##                                    (0.165)   (0.165)
##
## black                                       -0.048
##                                             (0.171)
##
## ------------------------------------------------
## Observations  1,000     1,000     1,000     1,000
## ================================================
## Note:                    *p<0.1; **p<0.05; ***p<0.01
```

2.2 Coefficients Interpretation

- In ordered logit models, the coefficients capture the effect on the log odds of moving to the "higher rank". The exponentiated coefficients indicate the **ratio between the odds** after and before the given predictor increased by one unit.
- The odds here is defined as the probability of being in a higher category divided by the probability of being in the current or lower category.

$$\frac{\frac{Pr(y_i>j|X_z+1)}{Pr(y_i\leq j|X_z+1)}}{\frac{Pr(y_i>j|X_z)}{Pr(y_i\leq j|X_z)}} = \frac{\frac{1-Pr(y_i\leq j|X_z+1)}{Pr(y_i\leq j|X_z+1)}}{\frac{1-Pr(y_i\leq j|X_z)}{Pr(y_i\leq j|X_z)}}$$
$$= \frac{exp(-\phi_j + (x_z+1)\beta_z)}{exp(-\phi_j + x_z\beta_z)}$$
$$= exp(\beta_z)$$

- To get these odds ratios in R, use `exp(coef(your_model_object))` (same as the code you use for getting odds ratio for logistic models).

```
## odds Ratio
exp(coef(ologit4))
```

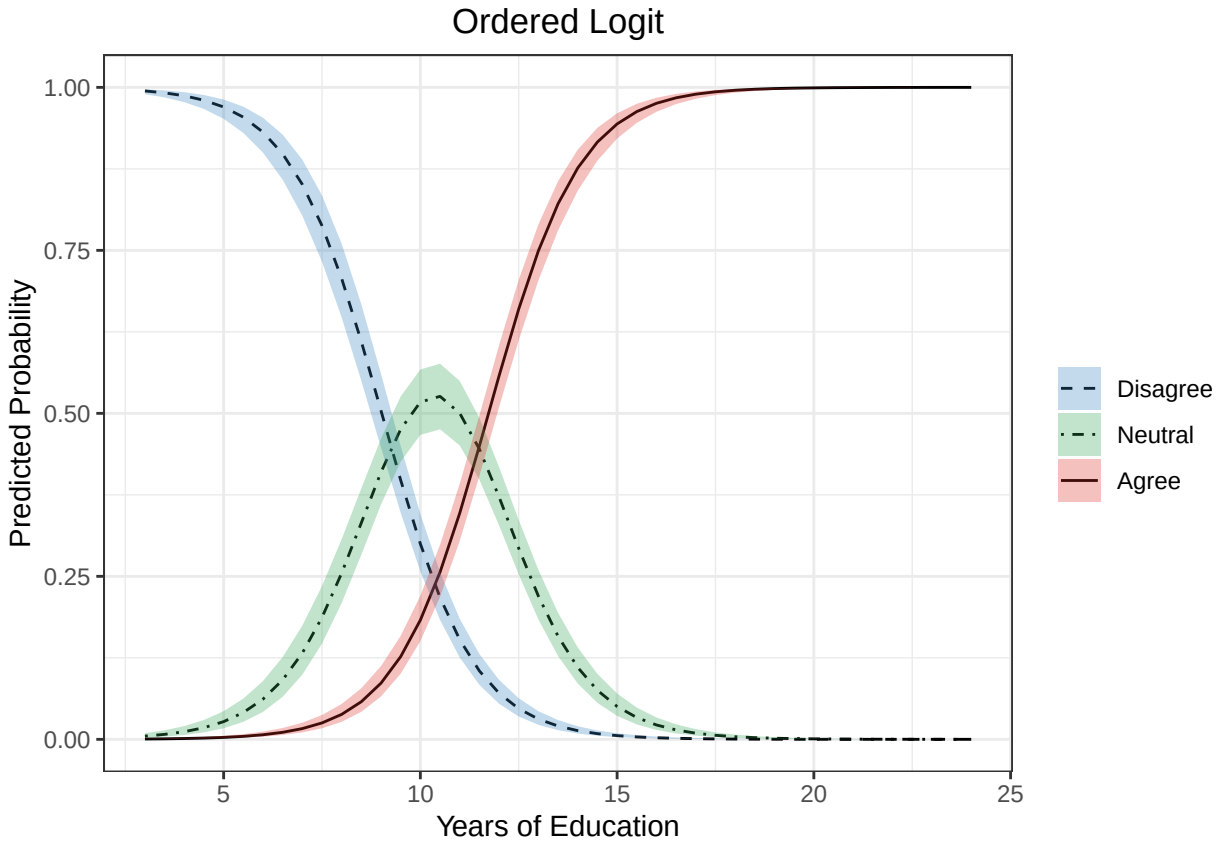
```
##      eduy       age    female     black
## 2.3726365 0.7965623 2.8403564 0.9533769
```

- Note that the term $exp(\beta_z)$ does not depend on j. In other words, the effect of X_z on y_i moving to a higher category is the same across j. The effect of moving from the lowest category to one higher is the same as from the second highest category to the highest category. This is **the proportional odds assumption/parallel regression assumption**
- *Question:* How would we interpret the coefficient on education? For each year of education a person has, their odds of moving into the next highest category of support increase by a factor of 2.37.

2.3 Plot Predicted Probability

```
## dataframe for prediction
predicted_ord <- as.data.frame(Effect(c("eduy"),
                                      ologit4,
                                      xlevels = list(
                                        eduy = seq(3, 24, by = 0.5),
                                        age = mean(support_df$age),
                                        black = mean(support_df$black),
                                        female = mean(support_df$female))
                                      ),
                               level=95)


## get predicted yhat, pivot to long form
predicted_y_ord <- predicted_ord %>%
  dplyr::select(eduy, prob.X1, prob.X2, prob.X3) %>%
  pivot_longer(!eduy, names_to = "level_y", values_to = "yhat")

## get predicted upper CI of yhat, pivot to long form
predicted_upr_ord <- predicted_ord %>%
  dplyr::select(eduy, U.prob.X1, U.prob.X2, U.prob.X3) %>%
  pivot_longer(!eduy, names_to = "level_upr", values_to = "upr") %>%
  dplyr::select(-eduy, -level_upr)

## get predicted lower CI of yhat, pivot to long form
predicted_lwr_ord <- predicted_ord %>%
  dplyr::select(eduy, L.prob.X1, L.prob.X2, L.prob.X3) %>%
  pivot_longer(!eduy, names_to = "level_lwr", values_to = "lwr") %>%
  dplyr::select(-eduy, -level_lwr)
```

```r
## combine to one df for plotting
predicted_plot_ord <- cbind(predicted_y_ord, predicted_upr_ord, predicted_lwr_ord)

## plot
figure1 <- predicted_plot_ord %>%
  ggplot(aes(x = eduy, y = yhat,
             ymax = upr, ymin = lwr,
             fill = as.factor(level_y),
             linetype = as.factor(level_y))) +
  geom_line() +
  geom_ribbon(alpha = 0.3) +
  labs(title = "Ordered Logit",
       x = "Years of Education",
       y = "Predicted Probability") +
  scale_fill_manual(name = "",
                    values = c("#3182bd", "#31a354", "#de2d26"),
                    label = c("Disagree", "Neutral", "Agree")) +
  scale_linetype_manual(name = "",
                        values = c("dashed", "dotdash", "solid"),
                        label = c("Disagree", "Neutral", "Agree")) +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5))
figure1
```



An Interlude: The Softmax Function

- So far, our logistic regression formulas (for binary logits and multinomial ordered logits) have been based on the sigmoid function:

$$\frac{1}{1+e^{-x}}$$

- Our next two types of logistic regression are going to be based on the softmax function. The general softmax form is:

$$\frac{e^{z_k}}{\sum_{j=1}^{k} e^{z_j}}$$

- Softmax vs. sigmoid
  - The softmax function has a similar but not identical shape to the sigmoid function.
  - The key difference is that the sigmoid function is for binary classification (ex. binary logistic regression, ordered multinomial regression), while the softmax function is for multiclass classification (ex. multinomial logistic regression, conditional logistic regression).
  - In the special case of multinomial logistic regression where the number of response options = 2, we can show algebraically that our multinomial softmax regression formula reduces down to the sigmoid binary regression formula
- The softmax function is often used in deep learning as the last layer of a neural network which transforms a vector into the probability that the input falls into each of the desired classification categories.

Part 3: Multinomial Logit Regression Model

3.1 Model Setup

- Multinomial logit model can be used to predict the probability of a response falling into a certain category among K categories that are *not ordered*.
- We think of the problem as fitting $K-1$ independent binary logit models, where one of the possible outcomes is defined as a pivot, and the $K-1$ outcomes are compared with the pivot outcome.
  - A binary logistic model is a special case of multinomial logit model
  - Recall a binary model has the form:

$$\begin{aligned}\text{logit}(Pr(Y_i=1)) &= \log\left(\frac{Pr(Y_i=1)}{1-Pr(Y_i=1)}\right)\\ &= \log\left(\frac{Pr(Y_i=1)}{Pr(Y_i=0)}\right)\\ &= \alpha_1+\beta_1 X_i\end{aligned}$$

- which essentially compares the outcome $(Y_i=1)$ with the pivot, reference outcome $(Y_i=0)$
- Now in parallel, with multiple K outcomes, the $K-1$ non-pivotal outcomes is assumed to be (assuming the first group is the pivot one):

$$\log\left(\frac{Pr(Y_i = 2)}{Pr(Y_i = 1)}\right) = \alpha_2 + \beta_2 X_i$$
$$\log\left(\frac{Pr(Y_i = 3)}{Pr(Y_i = 1)}\right) = \alpha_3 + \beta_3 X_i$$
$$\ldots$$
$$\log\left(\frac{Pr(Y_i = K)}{Pr(Y_i = 1)}\right) = \alpha_K + \beta_K X_i$$

- Rewriting each probability, we get:

$$Pr(Y_i = 2) = \exp(\alpha_2 + \beta_2 X_i) \times Pr(Y_i = 1)$$
$$Pr(Y_i = 3) = \exp(\alpha_3 + \beta_3 X_i) \times Pr(Y_i = 1)$$
$$\ldots$$
$$Pr(Y_i = K) = \exp(\alpha_K + \beta_K X_i) \times Pr(Y_i = 1)$$

- As $\sum_{k=1}^{K} Pr(Y_i = k) = 1$, we get:

$$Pr(Y_i = 1) = \frac{1}{1 + \sum_{k=2}^{K} \exp(\alpha_k + \beta_k X_i)}$$
$$Pr(Y_i = 2) = \frac{\exp(\alpha_k + \beta_2 X_i)}{1 + \sum_{k=2}^{K} \exp(\alpha_k + \beta_k X_i)}$$
$$\ldots$$
$$Pr(Y_i = K) = \frac{\exp(\alpha_k + \beta_2 X_i)}{1 + \sum_{k=2}^{K} \exp(\alpha_k + \beta_k X_i)}$$

- Multinomial logit model can be estimated using the `multinom()` function from the `nnet` package.

```
## estimate multinomial logit models
mlogit1 <- multinom(support_level ~ eduy, data = support_df)
```

```
## # weights:  9 (4 variable)
## initial  value 1098.612289
## iter  10 value 725.903623
## final  value 725.896424
## converged
```

```
mlogit2 <- multinom(support_level ~ eduy + age, data = support_df)
```

```
## # weights:  12 (6 variable)
## initial  value 1098.612289
## iter  10 value 600.038827
## iter  20 value 599.386590
## iter  20 value 599.386589
## iter  20 value 599.386589
## final  value 599.386589
## converged
```

```
mlogit3 <- multinom(support_level ~ eduy + age + female, data = support_df)
```

```
## # weights:  15 (8 variable)
## initial  value 1098.612289
## iter  10 value 587.970856
## final  value 578.365184
## converged
```

```
mlogit4 <- multinom(support_level ~ eduy + age + female + black, data = support_df)
```

```
## # weights:  18 (10 variable)
## initial  value 1098.612289
## iter  10 value 604.789528
## iter  20 value 578.283131
## iter  20 value 578.283129
## iter  20 value 578.283129
## final  value 578.283129
## converged
```

```
stargazer(mlogit1, mlogit2, mlogit3, mlogit4,
          type="text")
```

```
##
## ================================================================================================
##                                          Dependent variable:
##                     ---------------------------------------------------------------------------
##                        2         3         2         3         2         3         2         3
##                       (1)       (2)       (3)       (4)       (5)       (6)       (7)       (8)
## ------------------------------------------------------------------------------------------------
## eduy                0.384***  0.930***  0.556***  1.283***  0.599***  1.381***  0.599***  1.382***
##                      (0.047)   (0.057)   (0.058)   (0.079)   (0.061)   (0.085)   (0.062)   (0.085)
##
## age                                    -0.160*** -0.346*** -0.166*** -0.360*** -0.167*** -0.360***
##                                         (0.021)   (0.027)   (0.022)   (0.028)   (0.022)   (0.028)
##
## female                                                     0.721***  1.681***  0.723***  1.687***
##                                                             (0.223)   (0.274)   (0.224)   (0.274)
##
## black                                                                          -0.019    -0.099
##                                                                                (0.233)   (0.280)
##
## Constant            -3.854*** -9.815***   1.140     0.314     0.658    -1.007     0.670    -0.979
##                      (0.470)   (0.637)   (0.788)   (0.950)   (0.810)   (1.000)   (0.817)   (1.006)
##
## ------------------------------------------------------------------------------------------------
## Akaike Inf. Crit. 1,459.793 1,459.793 1,210.773 1,210.773 1,172.730 1,172.730 1,176.566 1,176.566
## ================================================================================================
## Note:                                                                *p<0.1; **p<0.05; ***p<0.01
```

3.2 Coefficients Interpretation

- The exponentiated regression coefficients from the multinomial logit model can be interpreted in terms of **relative risk ratios**. This makes the interpretation of the coefficients a bit more intuitive, compared to the coefficients from either binary or ordinal logistic regression.

$$
\begin{aligned}
\text{Relative Risk Ratio} &= \frac{\frac{Pr(Y_i=k|x_i+1)}{Pr(Y_i=1|x_i+1)}}{\frac{Pr(Y_i=k|x_i)}{Pr(Y_i=1|x_i)}} \\
&= \frac{\exp(\alpha_k)\exp\left[\beta_k(x_i+1)\right]}{\exp(\alpha_k)\exp(\beta_k x_i)} \\
&= \exp(\beta_k)
\end{aligned}
$$

- The interpretation for β_k is: holding all other factors constant, for one unit increase of the predictor, the relative risk of falling into the category k, compared with falling into the baseline category, increases by a factor of $exp(\beta_k)$ (note: multiplicative!).
- To get the relative risk ratios in R, use `exp(coef(your_model_object))` (same as the code you use for getting odds ratio for logistic models).

```
## get relative risk ratios for the 4th model
exp(coef(mlogit4))
```

```
##   (Intercept)     eduy       age   female     black
## 2   1.9535790 1.820853 0.8464970 2.060538 0.9811196
## 3   0.3756124 3.983080 0.6977276 5.400878 0.9056941
```

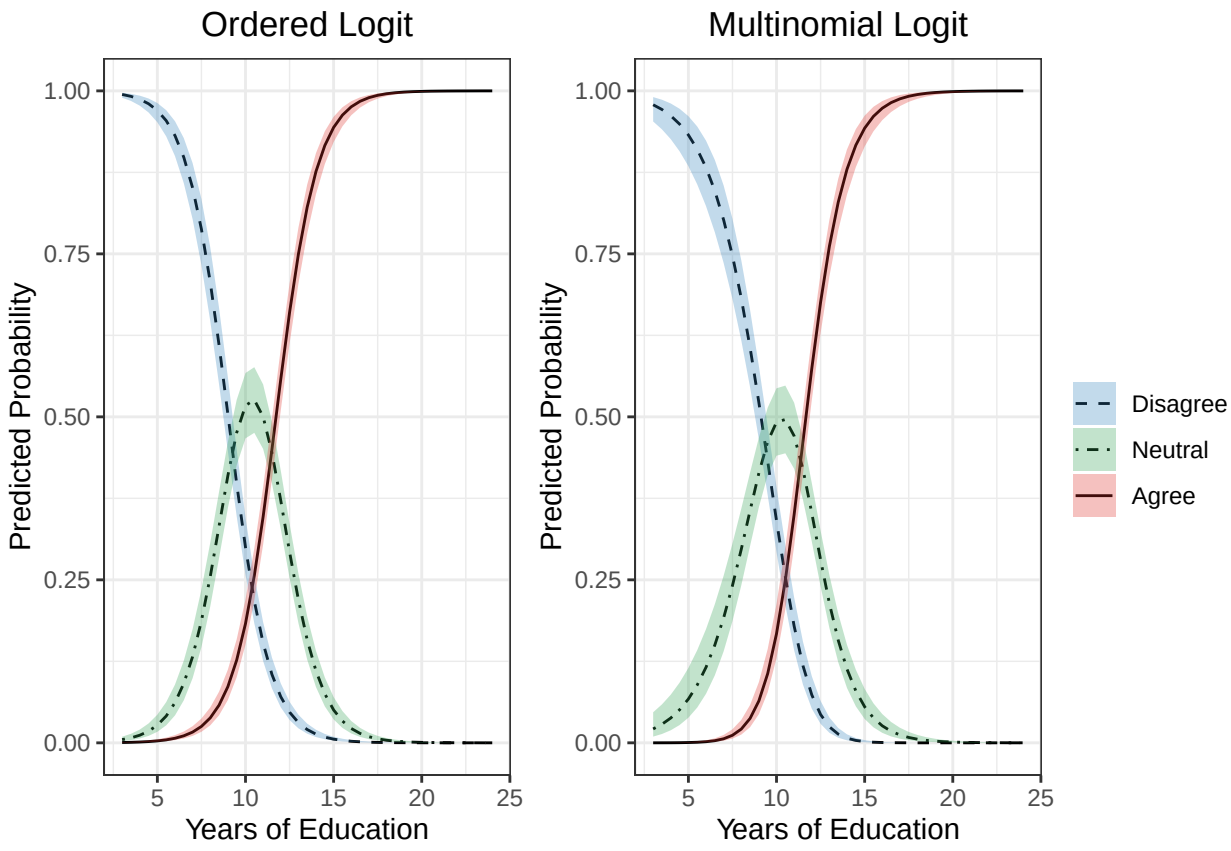
- *Question:* How would we interpret the coefficient on education in the first row? For each year of education a person has, their odds of being neutral on same-sex marriage versus opposing it increase by a factor of 1.82.

3.3 Plot Predicted Probabilities

- We can also plot the predicted effect for multinomial logistic models. For example, we can plot the predicted probabilities for the three possible outcomes (support, neutral, oppose) using the `Effect()` function.

```
# Get predicted y values
predicted_mul <- as.data.frame(Effect(c("eduy"),
                                      mlogit4,
                                      xlevels = list(
                                        eduy = seq(3, 24, by = 0.5),
                                        age = mean(support_df$age),
                                        black = mean(support_df$black),
                                        female = mean(support_df$female))
                                      ),
                               level=95)

# Get predicted yhat, pivot to long form
predicted_y_mul <- predicted_mul %>%
  dplyr::select(eduy, prob.X1, prob.X2, prob.X3) %>%
  pivot_longer(!eduy, names_to = "level_y", values_to = "yhat")

# Get predicted upper CI of yhat, pivot to long form
predicted_upr_mul <- predicted_mul %>%
  dplyr::select(eduy, U.prob.X1, U.prob.X2, U.prob.X3) %>%
  pivot_longer(!eduy, names_to = "level_upr", values_to = "upr") %>%
  dplyr::select(-eduy, -level_upr)

# Get predicted lower CI of yhat, pivot to long form
predicted_lwr_mul <- predicted_mul %>%
  dplyr::select(eduy, L.prob.X1, L.prob.X2, L.prob.X3) %>%
  pivot_longer(!eduy, names_to = "level_lwr", values_to = "lwr") %>%
  dplyr::select(-eduy, -level_lwr)

# Combine to one df for plotting
predicted_plot_mul <- cbind(predicted_y_mul, predicted_upr_mul, predicted_lwr_mul)

# Plot
```

```r
figure2 <- predicted_plot_mul %>%
  ggplot(aes(x = eduy, y = yhat,
             ymax = upr, ymin = lwr,
             fill = as.factor(level_y),
             linetype = as.factor(level_y))) +
  geom_line() +
  geom_ribbon(alpha = 0.3) +
  labs(title = "Multinomial Logit",
       x = "Years of Education",
       y = "Predicted Probability") +
  scale_fill_manual(name = "",
                    values = c("#3182bd", "#31a354", "#de2d26"),
                    label = c("Disagree", "Neutral", "Agree")) +
  scale_linetype_manual(name = "",
                        values = c("dashed", "dotdash", "solid"),
                        label = c("Disagree", "Neutral", "Agree")) +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5))

ggarrange(figure1, figure2, ncol=2, common.legend = TRUE,
          legend="right")
```



Part 4: Conditional Logit Regression Model

4.1 Model Setup

- Conceptually, conditional logistic regressions calculate how much features of a certain outcome influence the probability of person i choosing that outcome compared with the other outcome options. You can also use these models with strata rather than individuals.
  - Notice that this can be thought of as a type of fixed-effects model. We are calculating how much explanatory variables affect the probability of each outcome given a set of underlying individual traits that can also affect how much these explanatory variables matter (individual fixed-effects).
- For subject i and response choice j, suppose there are Q possible choices. Let X_{ij} denote the predictors that affect whether i will choose j (which may depend on characteristics of both i and j).
- The probability of person i selecting option j is:

$$\pi_{ij} = \frac{exp(\beta^T X_{ij})}{\sum_{q=1}^{Q} exp(\beta^T X_{iq})}$$

- This is in the form of a softmax function.
- Conceptually, the numerator is calculating how much evidence there is that individual i will choose a particular category and the denominator adds up the evidence for each category that individual i will choose each of them. This makes all the probabilities sum to 1, which is logical (individual i has an 100% chance of picking any category at all).
- Using the softmax function here has several advantages, including the linearity of the relative probability between π_{ij} and π_{iq}

$$log(\frac{\pi_{ij}}{\pi_{iq}}) = \beta^T X_{ij} - \beta^T X_{iq}$$

- Conditional logit models can be estimated using the `clogit()` function from the `survival` package.
- To demonstrate this we'll use a simulated dataset that mimics the neighborhood choice example we talked about in class. Notice this data is in long form (meaning that each person is represented through 3 rows, one for each neighborhood). This is necessary to use the `clogit()` command.

```
# load in data
neighborhoods <- read.csv("data/neighborhoods.csv")
```

- This dataset contains both individual-level traits and neighborhood traits.
  - Individual
    - **person_id**: identifies the person
    - **income**: individual income
    - **kids**: does the person have children
    - **choice**: observed choice (1 if they chose the neighborhood, 0 if they didn't)
  - Neighborhood
    - **neigh_id**: identifies the neighborhood
    - **trees**: neighborhood tree density
    - **schools**: neighborhood school rating
    - **price**: average neighborhood rent

```
## estimate conditional logit model
clogit_mod <-
  clogit(choice ~ trees + schools + price + income*price + kids*schools + income,
           strata(person_id),
  data = neighborhoods
```

```
)

stargazer(clogit_mod,
          type="text")
```

```
##
## ================================================
##                           Dependent variable:
##                       ---------------------------
##                                  choice
## ------------------------------------------------
## trees                           0.118***
##                                  (0.012)
##
## schools                         0.671***
##                                  (0.094)
##
## price                           -0.004**
##                                  (0.002)
##
## income                          -0.00003
##                                 (0.00005)
##
## kids                            -2.736**
##                                  (1.378)
##
## price:income                      0.000
##                                 (0.00000)
##
## schools:kids                     0.308*
##                                  (0.177)
##
## ------------------------------------------------
## Observations                       800
## R2                                0.293
## Max. Possible R2                  0.672
## Log Likelihood                  -307.919
## Wald Test                 147.040*** (df = 7)
## LR Test                   277.049*** (df = 7)
## Score (Logrank) Test      227.105*** (df = 7)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

Notice that individual-level traits do not enter the model except as interaction terms. This is because they are "conditioned out" because we're only looking at the probability of choosing each neighborhood within person. Therefore, individual-level traits won't vary unless they're interacted with a neighborhood-level characteristic.

You can also use this model for matched data or data in stratum.

4.2 Coefficients Interpretation

- Just like other logistic regressions, the coefficients in this model represent log odds. To get odds ratios, we exponentiate:

```r
## get odds for model
exp(coef(clogit_mod))
```

```
##        trees      schools        price       income         kids price:income
##   1.12510307   1.95599779   0.99592066   0.99997475   0.06480903   1.00000001
## schools:kids
##   1.36090416
```

- These odds represent how much higher odds a given outcome has of being selected based on a one-unit increase in a certain feature of that outcome.
- *Question:* How would we interpret the coefficient on trees? For a given person, a neighborhood with 1-unit higher tree density has 1.13 times higher odds of being selected.

Exercise (10 min)

How does age affect the likelihood that one will support same-sex marriage? Use `support_df` to investigate.

1. Pick one of the logistic regression models we have learned so far and use it to calculate the affect of age on probability of support of same-sex marriage.

```r
# first releveling so that support is the reference category
support_df <- support_df %>%
  mutate(
    support_level = support_level
  )

# running regression
support_logit <- multinom(support_level ~ age + eduy + female + black,
                          data = support_df)
```

```
## # weights:  18 (10 variable)
## initial  value 1098.612289
## iter  10 value 604.789528
## iter  20 value 578.283131
## iter  20 value 578.283129
## iter  20 value 578.283129
## final  value 578.283129
## converged
```

2. Create a clean table of your regression.

```r
stargazer(support_logit,
          type="text")
```

```
##
## ================================================
##                         Dependent variable:
##                     ---------------------------
##                           2             3
##                          (1)           (2)
## ------------------------------------------------
## age                    -0.167***     -0.360***
##                         (0.022)       (0.028)
##
## eduy                    0.599***      1.382***
##                         (0.062)       (0.085)
```

```
##
## female                0.723***      1.687***
##                       (0.224)        (0.274)
##
## black                  -0.019        -0.099
##                       (0.233)        (0.280)
##
## Constant               0.670         -0.979
##                       (0.817)        (1.006)
##
## ------------------------------------------------
## Akaike Inf. Crit.   1,176.566      1,176.566
## ================================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```

3. Interpret your regression in 1-2 sentences.

```
## get relative risk ratios
exp(coef(support_logit))
```

```
##   (Intercept)       age     eduy   female     black
## 2   1.9535790 0.8464970 1.820853 2.060538 0.9811196
## 3   0.3756124 0.6977276 3.983080 5.400878 0.9056941
```

Holding other factors constant, for every year a person ages, their odds of being neutral on same-sex marriage versus supporting it increase by a factor of 1.21. Their odds of opposing same-sex marriage as opposed to supporting it increase by a factor of 2.66.

4. Plot a graph of the predicted probability of support for each age.

```
predicted_mul <- as.data.frame(Effect(c("age"),
                                      support_logit,
                                      xlevels = list(
                                        eduy = seq(3, 24, by = 0.5),
                                        eduy = mean(support_df$eduy),
                                        black = mean(support_df$black),
                                        female = mean(support_df$female))
                                      ),
                               level=95)

# Get predicted yhat, pivot to long form
predicted_y_mul <- predicted_mul %>%
  dplyr::select(age, prob.X1, prob.X2, prob.X3) %>%
  pivot_longer(!age, names_to = "level_y", values_to = "yhat")

# Get predicted upper CI of yhat, pivot to long form
predicted_upr_mul <- predicted_mul %>%
  dplyr::select(age, U.prob.X1, U.prob.X2, U.prob.X3) %>%
  pivot_longer(!age, names_to = "level_upr", values_to = "upr") %>%
  dplyr::select(-age, -level_upr)

# Get predicted lower CI of yhat, pivot to long form
predicted_lwr_mul <- predicted_mul %>%
  dplyr::select(age, L.prob.X1, L.prob.X2, L.prob.X3) %>%
  pivot_longer(!age, names_to = "level_lwr", values_to = "lwr") %>%
  dplyr::select(-age, -level_lwr)
```

```r
# Combine to one df for plotting
predicted_plot_mul <- cbind(predicted_y_mul, predicted_upr_mul, predicted_lwr_mul)

# Plot
predicted_plot_mul %>%
  ggplot(aes(x = age, y = yhat,
             ymax = upr, ymin = lwr,
             fill = as.factor(level_y),
             linetype = as.factor(level_y))) +
  geom_line() +
  geom_ribbon(alpha = 0.3) +
  labs(title = "Multinomial Logit",
       x = "Age",
       y = "Predicted Probability") +
  scale_fill_manual(name = "",
                    values = c("red", "tan", "lightgreen"),
                    label = c("Oppose", "Neutral", "Support")) +
  scale_linetype_manual(name = "",
                        values = c("dashed", "dotdash", "solid"),
                        label = c("Oppose", "Neutral", "Support")) +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5))
```