# More On Logistic Regression
# Week 9

**PH 700A, Spring 2025**

Rick Calvo

## Table of contents


**1 Other Logistic Regression Models** **2**
- 1.1 Session Overview . . . 2
- 1.2 Packages . . . 2
- 1.3 Logistic Regression for Other Outcome Types . . . 2

**2 Conditional Logistic Regression** **3**
- 2.1 Overview . . . 3
- 2.2 Regression Characteristics . . . 3
- 2.3 Data Requirements . . . 3
- 2.4 Commands . . . 4

**3 Multinomial Logistic Regression** **5**
- 3.1 Overview . . . 5
- 3.2 Basic Modeling Assumptions . . . 5
- 3.3 Example Model . . . 5
- 3.4 Analysis Details . . . 6
- 3.5 Regression Caveats . . . 6
- 3.6 Preparation . . . 7
- 3.7 Commands . . . 7
- 3.8 Model Development . . . 7
- 3.9 Using gtsummary for Display . . . 8

**4 Ordinal Logistic Regression** **10**
- 4.1 Overview . . . 10
- 4.2 Modeling Structure . . . 10
- 4.3 Proportional Odds Assumption . . . 10
- 4.4 Commands . . . 11
- 4.5 Modeling Demonstration . . . 11
- 4.6 Testing the Proportional Odds Assumption . . . 12
- 4.7 Testing Multicollinearity . . . 13

# 1 Other Logistic Regression Models

## 1.1 Session Overview

- New Packages!
- Other Logistic Regression Models
  - Types
  - Rationale
  - Requirements
  - Commands
- Examples
  - Model Development
  - Diagnostics
  - Interpretations
- Visualization

## 1.2 Packages

```
library(survival)
```

- For conditional logistic regression

```
library(nnet)
```

- For multinomial logistic regression

```
library(MASS)
```

- For ordinal logistic regression

```
library(gtsummary)
```

```
library(car)
```

## 1.3 Logistic Regression for Other Outcome Types

- Conditional Logistic
- For situations with non-independence (i.e. matched studies)
- Multinomial/Polytomous Logistic
- For outcomes with more than 2 categories with no assumed order
- Ordinal Logistic
- For outcomes with more than 2 ordered categories

## 2 Conditional Logistic Regression

### 2.1 Overview

- An extension of binary logistic regression
- For matched data with a binary outcome
- The unit of analysis is the *group/match*, not the individual
- Permits adjustment for variables *not involved* in direct matching

**Unmatched Sample**

| | Event | Non-event |
|---|---|---|
| Exposed | a | b |
| Unexposed | c | d |

$$OR = \frac{(a)(d)}{(b)(c)}$$

**1:1 Matched Sample**

| | Non-event Exposed | Non-event Unexposed |
|---|---|---|
| Event Exposed | W | X |
| Event Unexposed | Y | Z |

$$OR = \frac{(X)}{(Y)}$$

### 2.2 Regression Characteristics

- Conditional logit utilizes the *exact conditional likelihood*
- There is no valid baseline term ($b0$) in conditional logistic regression
- You cannot *adjust* for variables used in the matching process
- Conditional logit results are inherently *biased toward the null*

### 2.3 Data Requirements

- A binary `outcome` coded 0 vs. 1
- Covariates of interest
- Basic assumptions of normality apply for continuous variables
- Frequencies of categories in factor variables must be sufficient (i.e. $> 10$)
- A matching/grouping identifier (`group`)

| patientid | age | sex | x2 | x3 | group | outcome | ... |
|---|---|---|---|---|---|---|---|

| patientid | age | sex | x2 | x3 | group | outcome | ... |
|---|---|---|---|---|---|---|---|
| A_123456 | 25 | M | 243 | A | 1 | 0 | ... |
| B_298298 | 25 | M | 125 | G | 1 | 1 | ... |
| C_222444 | 49 | F | 284 | B | 2 | 0 | ... |
| D_554457 | 49 | F | 96 | B | 2 | 1 | ... |
| E_000456 | 18 | F | 101 | C | 3 | 0 | ... |
| F_234292 | 18 | F | 192 | C | 3 | 1 | ... |
| G_245221 | 62 | M | 204 | Y | 4 | 0 | ... |
| H_501100 | 62 | M | 222 | Z | 4 | 1 | ... |

## 2.4 Commands

from `library(survival)`

```r
model.clogit <- clogit(outcome ~ x1 + x2 + ... + xn + strata(group_variable), data = df)

summary(model.clogit)
```

Why is clogit in the survival package?

At its most basic level and with a binary outcome, a conditional logit has the same likelihood formula as a stratified Cox model where time is a constant. The `clogit` command is essentially a *wrapper* for the `coxph` command with pre-specified arguments.

By default, only the beta estimates are stored in `model.clogit`. The ORs will have to be exponentiated manually.

- Coefficients are stored in the object as: `clogit.res$coef`
- Confidence intervals are stored as: `clogit.res$confint`

```r
expConvert <- cbind("Odds Ratio" = exp(coef(model.clogit)), exp(confint(model.clogit)))
```

The `cbind()` command is from base R and will create a mini data frame to hold the exponentiated ORs and 95% CIs.

`cbind` stands for "Column Bind" and will append columns to data frames.

# 3 Multinomial Logistic Regression

## 3.1 Overview

- An extension of binary logistic regression
- Outcome variable is nominal with >2 categories
- Categories do not have a logical order to them
- Observations can only experience **one category** of the outcome
- Many names for this: Polytomous, Polychotomous, Multiclass
- Still utilizes *Maximum Likelihood Estimation*

## 3.2 Basic Modeling Assumptions

1. One *reference category* for the outcome ($event_0$)
2. An *outcome category* (where $R = 1$) to assess to the reference ($event_R$) and:

- At least *one other outcome category* (where $R > 1$) to assess to the reference ($event_R$)

3. A common list of independent variables ($x_1, x_2, ..., x_k$)
4. A unique baseline probability ($b_{0.R}$) specific to the outcome category that is being analyzed
5. Unique estimates of association ($b_{1.R}, b_{2.R}, ..., b_{k.R}$) specific to the outcome category and independent variables being analyzed

$$ln(\frac{p_{event_R}}{p_{event_0}}) = b_{0.R} + b_{1.R}(x_1) + b_{2.R}(x_2) + ... + b_{k.R}(x_k)$$

## 3.3 Example Model

Outcome: Recategorized `df.ed$disposition`

- Reference category: *HOME* ($event_0$)
- Outcome category 1: *ADMITTED* ($event_1$)
- Outcome category 2: *OTHER* ($event_2$) - basically, everything else (ELOPED, LEFT AGAINST MEDICAL ADVICE, LEFT WITHOUT BEING SEEN, OTHER, TRANSFER)
- Independent variables: *racewhite*, *highpain*, *los*, *gender*

Comparison 1: Discharge to `HOME` vs. `ADMITTED`

$$ln(\frac{p_{event_1}}{p_{event_0}}) = b_{0.1} + b_{1.1}(racewhite) + b_{2.1}(highpain) + b_{3.1}(los) + b_{4.1}(gender)$$

Comparison 2: Discharge to Care Facility vs. Routine Home Discharge

$$ln(\frac{p_{event_2}}{p_{event_0}}) = b_{0.2} + b_{1.2}(racewhite) + b_{2.2}(highpain) + b_{3.2}(los) + b_{4.2}(gender)$$

### 3.4 Analysis Details

Resembles two independent binary logistic regression models with the same reference category. HOWEVER:

**In binary logistic regression:**

$$1.00 = p + q$$

Where $p$ = probably of the event and $q$ = probability of not having the event and both must equal to 100%.

**For polychotomous logistic regression:**

$$1.00 = p_0 + p_1 + p_2 + ... + p_R$$

Therefore, the likelihood function (and log-likelihood) is based on the probability of *all outcome categories* and their inputs and respective coefficients.

### 3.5 Regression Caveats

- There are no specific diagnostic tools for multicollinearity or outlier assessment for multinomial logits
- Can use the `vif()` commands out of `library(car)` for each pairwise outcome set (manually)
- Typically requires a very large sample due to multiple equations
- Can take a long time to compute
- Convergence toward a solution is adversely affected by low cell counts or poorly-powered samples
- The **Independence of Irrelevant Alternatives Assumption**
- Assumes that removal of outcome categories does not affect the odds of retained outcome categories

### 3.6 Preparation

First, check the distribution of your outcome to confirm its structure:

```
table(df$outcome)
```

By default, R will use the first value it shows as the reference category. Revise the reference group as needed:

```
df$outcome <- relevel(df$outcome, ref = "REFERENCE")
```

Covariates should be appropriately treated before proceeding with the regression

### 3.7 Commands

Commands for the primary mlogit analysis come from `nnet`.

```
library(nnet)

model <- multinom(outcome ~ var1 + var2 + ... + varN, data=df)

summary(model)
```

The `outcome` variable should be already set as a factor. The lowest category is the default reference unless it has been `relevel`ed

`summary(model)` displays the *coefficients* and *standard errors* for each variable for each outcome category (vs. the reference)

### 3.8 Model Development

```
library(nnet)

df <- df %>%
  mutate(disp3cat = case_when(
    disposition == "HOME" ~ "HOME",
    disposition == "ADMITTED" ~ "ADMITTED",
    .default = "OTHER"
  ))

df <- df %>%
  mutate(across(c("disp3cat", "racewhite", "arrival_transport", "gender"), as.factor))

table(df$disp3cat)
```

```
ADMITTED     HOME    OTHER
     150       60       12
```

```r
df$disp3cat <- relevel(df$disp3cat, ref = "HOME")

mlogit.m1 <- multinom(disp3cat ~ racewhite + lnlos + highpain + gender, data = df)
```

```
# weights:  18 (10 variable)
initial  value 243.891928
iter  10 value 166.399395
final  value 165.942351
converged
```

```r
summary(mlogit.m1)
```

```
Call:
multinom(formula = disp3cat ~ racewhite + lnlos + highpain +
    gender, data = df)

Coefficients:
         (Intercept) racewhite1      lnlos    highpain   genderM
ADMITTED    4.726817 -0.2381286 -0.6649349  0.08767236 0.6667535
OTHER       1.611070 -1.0402787 -0.5116694 -0.01617277 1.0502466

Std. Errors:
         (Intercept) racewhite1     lnlos  highpain   genderM
ADMITTED    1.456993  0.3663295 0.2368832 0.3517708 0.3579354
OTHER       2.667780  0.7269330 0.4441803 0.7047875 0.7229142

Residual Deviance: 331.8847
AIC: 351.8847
```

The `multinom()` function doesn't automatically show p-values, and the output only contains raw coefficients and standard errors.

## 3.9 Using gtsummary for Display

```r
library(gtsummary)

mlogit.m1 %>% tbl_regression(exponentiate = TRUE)
```

```
i Multinomial models have a different underlying structure than the models
  gtsummary was designed for.
* Functions designed to work with `tbl_regression()` objects may yield
  unexpected results.
```

| **Characteristic** | **OR** | **95% CI** | **p-value** |
|---|---|---|---|
| ADMITTED | | | |
| racewhite | | | |
| 0 | — | — | |
| 1 | 0.79 | 0.38, 1.62 | 0.5 |
| lnlos | 0.51 | 0.32, 0.82 | 0.005 |
| highpain | 1.09 | 0.55, 2.18 | 0.8 |
| gender | | | |
| F | — | — | |
| M | 1.95 | 0.97, 3.93 | 0.062 |
| OTHER | | | |
| racewhite | | | |
| 0 | — | — | |
| 1 | 0.35 | 0.09, 1.47 | 0.2 |
| lnlos | 0.60 | 0.25, 1.43 | 0.2 |
| highpain | 0.98 | 0.25, 3.92 | >0.9 |
| gender | | | |
| F | — | — | |
| M | 2.86 | 0.69, 11.8 | 0.15 |

Abbreviations: CI = Confidence Interval, OR = Odds Ratio

# 4 Ordinal Logistic Regression

## 4.1 Overview

- For categorical outcomes that had an order to to them
- Must have >2 categories
- Also known as *proportional odds* regression (not to be confused with *proportional hazards* regression)
- Only one model equation is generated and only one set of coefficients are estimated

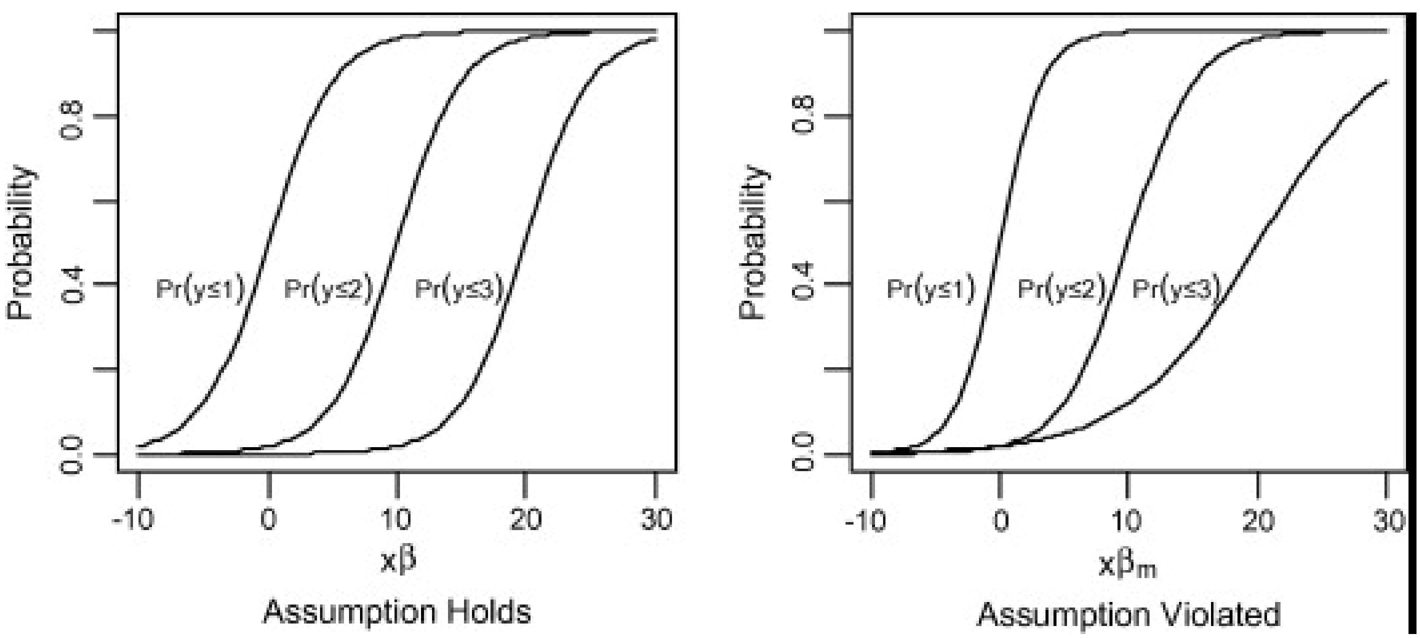
## 4.2 Modeling Structure

1. Any number of *ordinal outcome categories* ($event_R$), where $R > 1$
2. Reference outcome category is *all categories* that precede the one of interest ($event_{1-R}$) for each comparison
3. A common list of independent variables ($x_1, x_2, ..., x_n$)
4. Unique baseline probabilities ($b_{0.R}$) specific to the outcome categories being analyzed
5. One set of estimates ($b_1, b_2, ..., b_n$) for the entire model

$$ln(\frac{p_{event_R}}{p_{event_{1-R}}}) = b_{0.R} + b_1(x1) + b_2(x2) + ... + b_n(xn)$$

## 4.3 Proportional Odds Assumption

The relationship between every *category of elevation* and *category of reference* is approximately equal.




We can test the assumption for each risk factor using the **Cochran-Mantel-Haenszel** test.

Involves calculation of the probability of the outcome when increasing from a *lower category* to the *next category* in the order based on the independent variables

### 4.4 Commands

From `library(MASS)`

```
library(MASS)

model <- polr(outcome ~ x1 + x2 + x3, data=df, Hess=TRUE)

summary(model)
```

- `polr()` is the command for ordinal logistic regression
- `outcome ~ x1 + x2 + x3` is the standard formula notation in R
- `Hess = TRUE` includes the Hessian matrix; required for generating standard errors

### 4.5 Modeling Demonstration

```
library(MASS)
```

```

Attaching package: 'MASS'

The following object is masked from 'package:gtsummary':

    select

The following object is masked from 'package:dplyr':

    select
```

```
table(df$opioidlvl)
```

```

 Extreme     Mild Moderate     None 
      20       23       23      156 
```

```
df$opioidlvl <- factor(df$opioidlvl, levels = c("None", "Mild", "Moderate", "Extreme"), ordered = TR

ologit.m1 <- polr(opioidlvl ~ lnlos + gender, data = df, Hess=TRUE)

summary(ologit.m1)
```

| Characteristic | OR | 95% CI |
|---|---|---|
| lnlos | 1.39 | 0.91, 2.16 |
| gender | | |
| F | — | — |
| M | 4.45 | 2.44, 8.35 |

Abbreviations: CI = Confidence Interval, OR = Odds Ratio

```
Call:
polr(formula = opioidlvl ~ lnlos + gender, data = df, Hess = TRUE)

Coefficients:
         Value Std. Error t value
lnlos   0.3292     0.2211   1.489
genderM 1.4920     0.3131   4.766

Intercepts:
                 Value  Std. Error t value
None|Mild        3.5860 1.3397     2.6767
Mild|Moderate    4.2099 1.3468     3.1259
Moderate|Extreme 5.1651 1.3665     3.7797

Residual Deviance: 387.0998
AIC: 397.0998
```

```
tbl_regression(ologit.m1, exponentiate = TRUE)
```

## 4.6 Testing the Proportional Odds Assumption

Used to be an involved process that required you to generate the dummy variables and evaluate each association in normal binary logits.

Now, you can use the `car` package!

```
library(car)
```

```
Loading required package: carData


Attaching package: 'car'


The following object is masked from 'package:dplyr':

    recode
```

```
The following object is masked from 'package:purrr':

    some
```

```r
poTest(ologit.m1)
```

```
Tests for Proportional Odds
polr(formula = opioidlvl ~ lnlos + gender, data = df, Hess = TRUE)

        b[polr] b[>None] b[>Mild] b[>Moderate] Chisquare df Pr(>Chisq)
Overall                                             6.70  4       0.15
lnlos      0.329    0.241    0.189        0.811     5.97  2       0.05 .
genderM    1.492    1.441    1.534        2.045     1.00  2       0.61
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

### 4.7 Testing Multicollinearity

Collinearity can be evaluated with the variance inflation factor `vif()` from the `car` package

```r
library(car)

vif(ologit.m1)
```

```
   lnlos   gender
1.000113 1.000113
```