432 Class 19

https://thomaselove.github.io/432-2025

2025-03-25

## Today’s Topic

**Regression Models for Ordered Multi-Categorical Outcomes**

* Applying to Graduate School: A First Example
* Proportional Odds Logistic Regression Models
  + Using polr and then Using lrm
* Understanding and Interpreting the Model
* Testing the Proportional Odds Assumption
* Picturing the Model Fit

Chapter 27 of the Course Notes describes this material.

## Today’s R Setup

knitr::opts\_chunk$set(comment=NA)  
  
library(janitor)

Attaching package: 'janitor'

The following objects are masked from 'package:stats':  
  
 chisq.test, fisher.test

library(broom)  
library(gt)  
library(GGally) ## scatterplot matrix

Loading required package: ggplot2

Registered S3 method overwritten by 'GGally':  
 method from   
 +.gg ggplot2

library(scales) ## adjust label formatting within ggplot2  
library(MASS) ## fitting polr models  
library(nnet) ## fitting multinomial models  
library(conflicted)  
library(rms)

Loading required package: Hmisc

Attaching package: 'Hmisc'

The following object is masked from 'package:gt':  
  
 html

The following objects are masked from 'package:base':  
  
 format.pval, units

library(easystats)

# Attaching packages: easystats 0.7.4  
✔ bayestestR 0.15.2 ✔ correlation 0.8.6   
✔ datawizard 1.0.0 ✔ effectsize 1.0.0   
✔ insight 1.0.2 ✔ modelbased 0.9.0   
✔ performance 0.13.0 ✔ parameters 0.24.1  
✔ report 0.6.1 ✔ see 0.10.0

library(tidyverse)

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
✔ dplyr 1.1.4 ✔ readr 2.1.5  
✔ forcats 1.0.0 ✔ stringr 1.5.1  
✔ lubridate 1.9.4 ✔ tibble 3.2.1  
✔ purrr 1.0.4 ✔ tidyr 1.3.1

conflicts\_prefer(janitor::clean\_names, base::sum,  
 dplyr::filter, dplyr::select, dplyr::summarize)

[conflicted] Will prefer janitor::clean\_names over any other package.

[conflicted] Will prefer base::sum over any other package.  
[conflicted] Will prefer dplyr::filter over any other package.  
[conflicted] Will prefer dplyr::select over any other package.  
[conflicted] Will prefer dplyr::summarize over any other package.

theme\_set(theme\_bw())

# Applying to Graduate School

## The gradschool data and my **Source**

The **gradschool** example is adapted from a site at UCLA[[1]](#footnote-22).

* There, they look at 400 students.
* I simulated a new data set containing 530 college juniors.

Each subject is asked “Are you unlikely, somewhat likely, or very likely to apply to graduate school?” This is our outcome.

* No reason to think that the “distances” between these categories are equal.

## The gradschool variables

| Variable | Description |
| --- | --- |
| student | subject identifying code (A001 - A530) |
| apply | 3-level ordered outcome: “unlikely”, “somewhat likely” and “very likely” to apply |
| pared | 1 = at least one parent has a graduate degree, else 0 |
| public | 1 = undergraduate institution is public, else 0 |
| gpa | student’s undergraduate grade point average (max 4.00) |

## Ingesting the Data

gradschool <-   
 read\_csv("c19/data/gradschool.csv", show\_col\_types = FALSE) |>  
 clean\_names() |>  
 mutate(across(where(is\_character), as\_factor),  
 student = as.character(student))  
  
gradschool

# A tibble: 530 × 5  
 student apply pared public gpa  
 <chr> <fct> <dbl> <dbl> <dbl>  
 1 A001 very likely 0 0 3.41  
 2 A002 unlikely 0 0 2.38  
 3 A003 somewhat likely 0 0 3.35  
 4 A004 unlikely 0 1 3.45  
 5 A005 unlikely 1 1 3.27  
 6 A006 somewhat likely 1 0 3.41  
 7 A007 somewhat likely 0 0 2.83  
 8 A008 unlikely 0 0 3.64  
 9 A009 unlikely 0 0 2.52  
10 A010 unlikely 0 0 2.36  
# ℹ 520 more rows

## Our outcome as an *ordered* factor

gradschool <- gradschool |>  
 mutate(apply = fct\_relevel(apply, "unlikely",   
 "somewhat likely", "very likely"),  
 apply = factor(apply, ordered = TRUE))  
  
is.ordered(gradschool$apply)

[1] TRUE

glimpse(gradschool)

Rows: 530  
Columns: 5  
$ student <chr> "A001", "A002", "A003", "A004", "A005", "A006", "A007", "A008"…  
$ apply <ord> very likely, unlikely, somewhat likely, unlikely, unlikely, so…  
$ pared <dbl> 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0,…  
$ public <dbl> 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…  
$ gpa <dbl> 3.41, 2.38, 3.35, 3.45, 3.27, 3.41, 2.83, 3.64, 2.52, 2.36, 2.…

## Describing the gradschool data

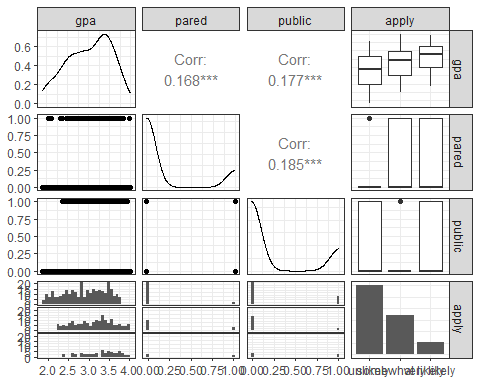
describe(gradschool) ## from Hmisc

gradschool   
  
 5 Variables 530 Observations  
--------------------------------------------------------------------------------  
student   
 n missing distinct   
 530 0 530   
  
lowest : A001 A002 A003 A004 A005, highest: A526 A527 A528 A529 A530  
--------------------------------------------------------------------------------  
apply   
 n missing distinct   
 530 0 3   
   
Value unlikely somewhat likely very likely  
Frequency 303 172 55  
Proportion 0.572 0.325 0.104  
--------------------------------------------------------------------------------  
pared   
 n missing distinct Info Sum Mean   
 530 0 2 0.47 103 0.1943   
  
--------------------------------------------------------------------------------  
public   
 n missing distinct Info Sum Mean   
 530 0 2 0.555 130 0.2453   
  
--------------------------------------------------------------------------------  
gpa   
 n missing distinct Info Mean pMedian Gmd .05   
 530 0 186 1 3.015 3.025 0.5919 2.104   
 .10 .25 .50 .75 .90 .95   
 2.279 2.610 3.080 3.440 3.660 3.760   
  
lowest : 1.9 1.91 1.92 1.93 1.94, highest: 3.95 3.97 3.98 3.99 4   
--------------------------------------------------------------------------------

## Scatterplot Matrix for gradschool

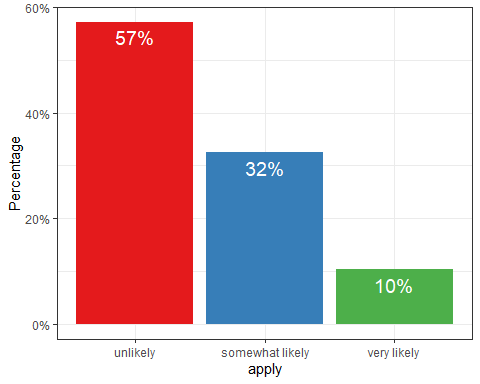
ggpairs(gradschool |> select(gpa, pared, public, apply)) ## outcome last

`stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
`stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
`stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



## Bar Chart of apply classifications with %s

ggplot(gradschool, aes(x = apply, fill = apply)) +   
 geom\_bar(aes(y =   
 (after\_stat(count)/sum(after\_stat(count))))) +  
 geom\_text(aes(y =   
 (after\_stat(count))/sum(after\_stat(count)),   
 label = scales::percent((after\_stat(count)) /   
 sum(after\_stat(count)))),  
 stat = "count", vjust = 1.5,   
 color = "white", size = 5) +  
 scale\_y\_continuous(labels = scales::percent) +  
 scale\_fill\_brewer(palette = "Set1") +  
 guides(fill = "none") +   
 labs(y = "Percentage")



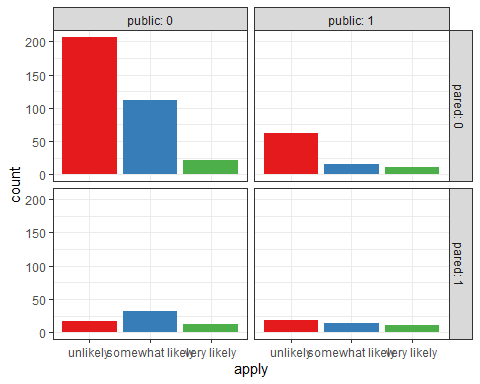
## Data (besides gpa) as Cross-Tabulation

ftable(xtabs(~ public + apply + pared, data = gradschool))

pared 0 1  
public apply   
0 unlikely 206 17  
 somewhat likely 111 32  
 very likely 22 12  
1 unlikely 62 18  
 somewhat likely 15 14  
 very likely 11 10

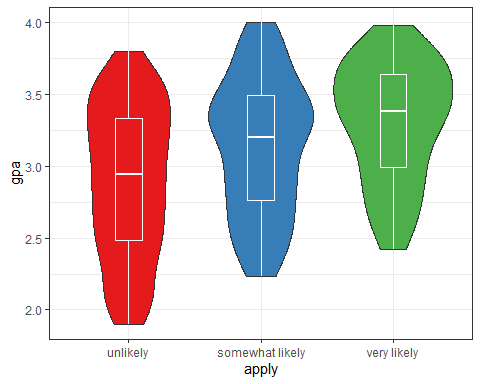
## apply percentages by public, pared

ggplot(gradschool, aes(x = apply, fill = apply)) +   
 geom\_bar() +  
 scale\_fill\_brewer(palette = "Set1") +  
 guides(fill = "none") +   
 facet\_grid(pared ~ public, labeller = "label\_both")



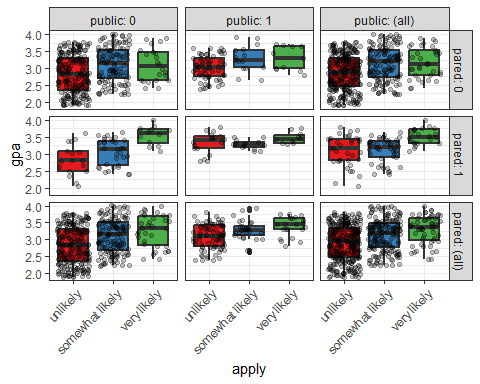
## Breakdown of gpa by apply

ggplot(gradschool, aes(x = apply, y = gpa, fill = apply)) +   
 geom\_violin(trim = TRUE) +  
 geom\_boxplot(col = "white", width = 0.2) +  
 scale\_fill\_brewer(palette = "Set1") +  
 guides(fill = "none")



## gpa by three other variables

ggplot(gradschool, aes(x = apply, y = gpa)) +  
 geom\_boxplot(aes(fill = apply), size = .75) +  
 geom\_jitter(alpha = .25) +  
 facet\_grid(pared ~ public, margins = TRUE,   
 labeller = "label\_both") +  
 scale\_fill\_brewer(palette = "Set1") +  
 guides(fill = "none") +  
 theme(axis.text.x =   
 element\_text(angle = 45, hjust = 1, vjust = 1))



# Proportional Odds Logit Model via polr

## Fitting the POLR model

We use the polr function from the MASS package:

mod\_p1 <- polr(apply ~ pared + public + gpa,   
 data = gradschool, Hess=TRUE)

The polr name comes from proportional odds logistic regression, highlighting a key assumption of this model.

* We specify Hess=TRUE to have the model return the observed information matrix from optimization (called the Hessian) which is used to get appropriate standard errors.

## mod\_p1 Predicted Probabilities

The model’s predicted probabilities are usually the best way to understand what it does.

For example, we vary gpa for each level of pared and public and calculate the model’s estimated probability of being in each category of apply.

First, create a new tibble of values to use for prediction.

newdat <- tibble(  
 pared = rep(0:1, 200),  
 public = rep(0:1, each = 200),  
 gpa = rep(seq(from = 1.9, to = 4, length.out = 100), 4))

## mod\_p1 Predicted Probabilities

Now, make predictions using model mod\_p1:

newdat\_p1 <- cbind(newdat,   
 predict(mod\_p1, newdat, type = "probs"))  
head(newdat\_p1) |> gt() |> fmt\_number(decimals = 3) |>  
 tab\_options(table.font.size = 20)

| pared | public | gpa | unlikely | somewhat likely | very likely |
| --- | --- | --- | --- | --- | --- |
| 0.000 | 0.000 | 1.900 | 0.846 | 0.132 | 0.022 |
| 1.000 | 0.000 | 1.921 | 0.629 | 0.302 | 0.069 |
| 0.000 | 0.000 | 1.942 | 0.840 | 0.137 | 0.024 |
| 1.000 | 0.000 | 1.964 | 0.617 | 0.310 | 0.073 |
| 0.000 | 0.000 | 1.985 | 0.833 | 0.142 | 0.025 |
| 1.000 | 0.000 | 2.006 | 0.606 | 0.318 | 0.076 |

## Reshape data

Now, we reshape the data with pivot\_longer:

newdat\_long <-   
 pivot\_longer(newdat\_p1,   
 cols = c("unlikely":"very likely"),  
 names\_to = "level",  
 values\_to = "probability") |>  
 mutate(level = fct\_relevel(level, "unlikely",  
 "somewhat likely"))

Result on next slide…

## The newdat\_long data

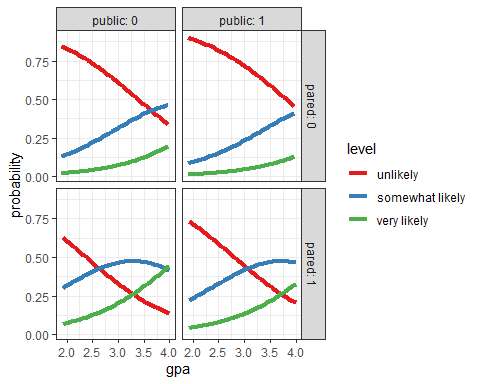
newdat\_long

# A tibble: 1,200 × 5  
 pared public gpa level probability  
 <int> <int> <dbl> <fct> <dbl>  
 1 0 0 1.9 unlikely 0.846   
 2 0 0 1.9 somewhat likely 0.132   
 3 0 0 1.9 very likely 0.0225  
 4 1 0 1.92 unlikely 0.629   
 5 1 0 1.92 somewhat likely 0.302   
 6 1 0 1.92 very likely 0.0694  
 7 0 0 1.94 unlikely 0.840   
 8 0 0 1.94 somewhat likely 0.137   
 9 0 0 1.94 very likely 0.0236  
10 1 0 1.96 unlikely 0.617   
# ℹ 1,190 more rows

## mod\_p1 Predictions

ggplot(newdat\_long, aes(x = gpa, y = probability,   
 color = level)) +  
 geom\_line(size = 1.5) +   
 scale\_color\_brewer(palette = "Set1") +  
 facet\_grid(pared ~ public, labeller="label\_both")

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
ℹ Please use `linewidth` instead.



## Predicted vs. Observed Classifications

Predictions in the rows, Observed in the columns

addmargins(table(predict(mod\_p1), gradschool$apply,   
 dnn = c("predicted", "actual")))

actual  
predicted unlikely somewhat likely very likely Sum  
 unlikely 264 112 29 405  
 somewhat likely 39 60 25 124  
 very likely 0 0 1 1  
 Sum 303 172 55 530

We only predict one subject to be in the “very likely” group by modal prediction.

## Proportional Odds Logistic Model

Our outcome apply has three levels, so mod\_p1 includes two equations:

* one estimating the log odds that apply will be less than or equal to 1 (apply = “unlikely”)
* one estimating the log odds that apply 2 (apply = “unlikely” or “somewhat likely”)

That’s all we need, since Pr(apply 3) = 1, because “very likely” is the highest apply category.

## Parameters of the POLR Model

* The parameters to be fit include two intercepts:
  + will be the unlikely|somewhat likely parameter
  + will be the somewhat likely|very likely parameter (*read these as zeta-one, and zeta-two*)

We’ll have a total of five free parameters when we add in the slopes () for pared, public and gpa.

* The two logistic equations that will be fit differ only in their intercepts.

## summary(mod\_p1)

summary(mod\_p1)

Call:  
polr(formula = apply ~ pared + public + gpa, data = gradschool,   
 Hess = TRUE)  
  
Coefficients:  
 Value Std. Error t value  
pared 1.1525 0.2184 5.276  
public -0.4949 0.2195 -2.254  
gpa 1.1416 0.1850 6.171  
  
Intercepts:  
 Value Std. Error t value  
unlikely|somewhat likely 3.8727 0.5721 6.7692  
somewhat likely|very likely 5.9413 0.6063 9.7993  
  
Residual Deviance: 900.9629   
AIC: 910.9629

## Model Parameters for mod\_p1

model\_parameters(mod\_p1, pretty\_names = FALSE, ci = 0.90)

# alpha  
  
Parameter | Log-Odds | SE | 90% CI | t(525) | p  
------------------------------------------------------------------------------  
unlikely|somewhat likely | 3.87 | 0.57 | [2.93, 4.82] | 6.77 | < .001  
somewhat likely|very likely | 5.94 | 0.61 | [4.94, 6.94] | 9.80 | < .001  
  
# beta  
  
Parameter | Log-Odds | SE | 90% CI | t(525) | p  
--------------------------------------------------------------  
pared | 1.15 | 0.22 | [ 0.79, 1.51] | 5.28 | < .001  
public | -0.49 | 0.22 | [-0.86, -0.14] | -2.25 | 0.025   
gpa | 1.14 | 0.18 | [ 0.84, 1.45] | 6.17 | < .001

Uncertainty intervals (profile-likelihood) and p-values (two-tailed)  
 computed using a Wald t-distribution approximation.

The model has a log- or logit-link. Consider using `exponentiate =  
 TRUE` to interpret coefficients as ratios.  
   
Some coefficients are very large, which may indicate issues with  
 complete separation.

## Understanding the Model

in general. In our setting, we have …

## The mod\_p1 equations…

and

## confint(mod\_p1)

Confidence intervals for the slope coefficients on the log odds scale can be estimated in the usual way.

confint(mod\_p1, conf.level = 0.95)

Waiting for profiling to be done...

2.5 % 97.5 %  
pared 0.7257019 1.58305735  
public -0.9320573 -0.07029727  
gpa 0.7837559 1.50974002

These CIs describe results in units of ordered log odds.

* For example, for a one unit increase in gpa, we expect a 1.14 increase in the expected value of apply (95% CI 0.78, 1.51) in the log odds scale, holding pared and public constant.
* This would be more straightforward if we exponentiated.

## Exponentiating the Coefficients

exp(coef(mod\_p1))

pared public gpa   
3.1660446 0.6096623 3.1318247

exp(confint(mod\_p1, conf.level = 0.95))

Waiting for profiling to be done...

2.5 % 97.5 %  
pared 2.0661808 4.8698218  
public 0.3937428 0.9321167  
gpa 2.1896811 4.5255541

## Parameters for mod\_p1 (exponentiating)

model\_parameters(mod\_p1, exponentiate = TRUE, pretty\_names = FALSE, ci = 0.95)

# alpha  
  
Parameter | Odds Ratio | SE | 95% CI | t(525) | p  
---------------------------------------------------------------------------------------  
unlikely|somewhat likely | 48.07 | 27.50 | [ 15.62, 147.92] | 6.77 | < .001  
somewhat likely|very likely | 380.42 | 230.65 | [115.61, 1251.79] | 9.80 | < .001  
  
# beta  
  
Parameter | Odds Ratio | SE | 95% CI | t(525) | p  
--------------------------------------------------------------  
pared | 3.17 | 0.69 | [2.07, 4.87] | 5.28 | < .001  
public | 0.61 | 0.13 | [0.39, 0.93] | -2.25 | 0.025   
gpa | 3.13 | 0.58 | [2.19, 4.53] | 6.17 | < .001

Uncertainty intervals (profile-likelihood) and p-values (two-tailed)  
 computed using a Wald t-distribution approximation.

## Interpreting the Coefficients

| Variable | Estimate | 95% CI |
| --- | --- | --- |
| gpa | 3.13 | (2.19, 4.53) |
| public | 0.61 | (0.39, 0.93) |
| pared | 3.17 | (2.07, 4.87) |

* When a student’s gpa increases by 1 unit, the odds of moving from “unlikely” applying to “somewhat likely” or “very likely” applying are multiplied by 3.13 (95% CI 2.19, 4.52), all else held constant.

## Interpreting the Coefficients

| Variable | Estimate | 95% CI |
| --- | --- | --- |
| gpa | 3.13 | (2.19, 4.53) |
| public | 0.61 | (0.39, 0.93) |
| pared | 3.17 | (2.07, 4.87) |

* For public, the odds of moving from a lower to higher apply status are multiplied by 0.61 (95% CI 0.39, 0.93) as we move from private to public, all else held constant.
* How about pared?

## Tidying mod\_p1

We’ll exponentiate here so that the estimates and confidence intervals describe the odds associated with changes in these coefficients.

tidy(mod\_p1, exponentiate = TRUE, conf.int = TRUE) |>   
 gt() |>   
 fmt\_number(columns = estimate:conf.high, decimals = 3) |>  
 tab\_options(table.font.size = 20)

| term | estimate | std.error | statistic | conf.low | conf.high | coef.type |
| --- | --- | --- | --- | --- | --- | --- |
| pared | 3.166 | 0.218 | 5.276 | 2.066 | 4.870 | coefficient |
| public | 0.610 | 0.220 | -2.254 | 0.394 | 0.932 | coefficient |
| gpa | 3.132 | 0.185 | 6.171 | 2.190 | 4.526 | coefficient |
| unlikely|somewhat likely | 48.074 | 0.572 | 6.769 | NA | NA | scale |
| somewhat likely|very likely | 380.416 | 0.606 | 9.799 | NA | NA | scale |

## Comparison to a Null Model

mod\_p0 <- polr(apply ~ 1, data = gradschool)  
  
anova(mod\_p1, mod\_p0)

Likelihood ratio tests of ordinal regression models  
  
Response: apply  
 Model Resid. df Resid. Dev Test Df LR stat. Pr(Chi)  
1 1 528 975.1828   
2 pared + public + gpa 525 900.9629 1 vs 2 3 74.21989 5.551115e-16

## AIC and BIC are available, too

# model including covariates  
glance(mod\_p1) |> gt() |>   
 fmt\_number(columns = logLik:deviance, decimals = 3) |>  
 tab\_options(table.font.size = 20)

| edf | logLik | AIC | BIC | deviance | df.residual | nobs |
| --- | --- | --- | --- | --- | --- | --- |
| 5 | -450.481 | 910.963 | 932.327 | 900.963 | 525 | 530 |

# null model; no covariates  
glance(mod\_p0) |> gt() |>   
 fmt\_number(columns = logLik:deviance, decimals = 3) |>  
 tab\_options(table.font.size = 20)

| edf | logLik | AIC | BIC | deviance | df.residual | nobs |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | -487.591 | 979.183 | 987.729 | 975.183 | 528 | 530 |

## Compare Parameters

compare\_parameters(mod\_p0, mod\_p1)

Re-fitting to get Hessian

# alpha  
  
Parameter | mod\_p0 | mod\_p1  
-----------------------------------------------------------  
unlikely|somewhat likely | | 3.87 ( 2.75, 5.00)  
somewhat likely|very likely | | 5.94 ( 4.75, 7.13)  
  
# beta  
  
Parameter | mod\_p0 | mod\_p1  
-----------------------------------------  
pared | | 1.15 ( 0.72, 1.58)  
public | | -0.49 (-0.93, -0.06)  
gpa | | 1.14 ( 0.78, 1.51)  
  
# Fixed Effects  
  
Parameter | mod\_p0 | mod\_p1  
--------------------------------------------------------  
unlikely|somewhat likely | 0.29 (0.12, 0.46) |   
somewhat likely|very likely | 2.16 (1.88, 2.44) |

## Tidying mod\_p0

tidy(mod\_p0, exponentiate = TRUE, conf.int = TRUE) |>   
 gt() |>   
 fmt\_number(columns = estimate:conf.high, decimals = 3) |>  
 tab\_options(table.font.size = 20)

Re-fitting to get Hessian

| term | estimate | std.error | statistic | conf.low | conf.high | coef.type |
| --- | --- | --- | --- | --- | --- | --- |
| unlikely|somewhat likely | 1.335 | 0.088 | 3.290 | NA | NA | scale |
| somewhat likely|very likely | 8.636 | 0.142 | 15.136 | NA | NA | scale |

## Proportional Odds Assumption (1/2)

One way to assess the proportional odds assumption is to compare the fit of the proportional odds logistic regression to a model that does not make that assumption.

* A natural candidate is a **multinomial logit** model, which is typically used to model unordered multi-categorical outcomes, and fits a slope to each level of the apply outcome in this case, as opposed to the proportional odds logit, which fits only one slope across all levels.

## Proportional Odds Assumption (2/2)

Since the proportional odds logistic regression model is nested in the multinomial logit, we can perform a likelihood ratio test.

* To do this, we first fit the multinomial logit model, with the multinom function from the nnet package.

### Fitting the multinomial model

m1\_multi <- multinom(apply ~ pared + public + gpa,   
 data = gradschool)

# weights: 15 (8 variable)  
initial value 582.264513   
iter 10 value 446.199617  
final value 445.443366   
converged

## The multinomial model

m1\_multi

Call:  
multinom(formula = apply ~ pared + public + gpa, data = gradschool)  
  
Coefficients:  
 (Intercept) pared public gpa  
somewhat likely -3.527249 1.072451 -0.97765580 0.9857488  
very likely -7.311227 1.400955 -0.02934361 1.6937996  
  
Residual Deviance: 890.8867   
AIC: 906.8867

## Parameters: m1\_multi

model\_parameters(m1\_multi, ci = 0.95, pretty\_names = FALSE)

# Response level: somewhat likely  
  
Parameter | Log-Odds | SE | 95% CI | z | p  
---------------------------------------------------------------  
(Intercept) | -3.53 | 0.62 | [-4.75, -2.31] | -5.66 | < .001  
pared | 1.07 | 0.26 | [ 0.56, 1.59] | 4.08 | < .001  
public | -0.98 | 0.26 | [-1.49, -0.46] | -3.73 | < .001  
gpa | 0.99 | 0.20 | [ 0.59, 1.39] | 4.83 | < .001  
  
# Response level: very likely  
  
Parameter | Log-Odds | SE | 95% CI | z | p  
---------------------------------------------------------------  
(Intercept) | -7.31 | 1.20 | [-9.66, -4.96] | -6.11 | < .001  
pared | 1.40 | 0.35 | [ 0.72, 2.08] | 4.02 | < .001  
public | -0.03 | 0.34 | [-0.69, 0.63] | -0.09 | 0.930   
gpa | 1.69 | 0.37 | [ 0.98, 2.41] | 4.62 | < .001

Uncertainty intervals (equal-tailed) and p-values (two-tailed) computed  
 using a Wald normal distribution approximation.

## Tidying m1\_multi

tidy(m1\_multi, conf.int = TRUE) |>   
 gt() |>   
 fmt\_number(columns = estimate:conf.high, decimals = 3) |>  
 tab\_options(table.font.size = 20)

| y.level | term | estimate | std.error | statistic | p.value | conf.low | conf.high |
| --- | --- | --- | --- | --- | --- | --- | --- |
| somewhat likely | (Intercept) | -3.527 | 0.623 | -5.663 | 0.000 | -4.748 | -2.306 |
| somewhat likely | pared | 1.072 | 0.263 | 4.081 | 0.000 | 0.557 | 1.587 |
| somewhat likely | public | -0.978 | 0.262 | -3.731 | 0.000 | -1.491 | -0.464 |
| somewhat likely | gpa | 0.986 | 0.204 | 4.826 | 0.000 | 0.585 | 1.386 |
| very likely | (Intercept) | -7.311 | 1.197 | -6.106 | 0.000 | -9.658 | -4.965 |
| very likely | pared | 1.401 | 0.348 | 4.021 | 0.000 | 0.718 | 2.084 |
| very likely | public | -0.029 | 0.336 | -0.087 | 0.930 | -0.689 | 0.630 |
| very likely | gpa | 1.694 | 0.366 | 4.622 | 0.000 | 0.975 | 2.412 |

## Comparing the Models

The multinomial logit fits two intercepts and six slopes, for a total of 8 estimated parameters.

The proportional odds logit, as we’ve seen, fits two intercepts and three slopes, for a total of 5. The difference is 3, and we use that number in the sequence below to build our test of the proportional odds assumption.

## Testing the Proportional Odds Assumption

LL\_1 <- logLik(mod\_p1)  
LL\_1m <- logLik(m1\_multi)  
(G <- -2 \* (LL\_1[1] - LL\_1m[1]))

[1] 10.07618

pchisq(G, 3, lower.tail = FALSE)

[1] 0.01792959

The *p* value is 0.018, so it indicates that the proportional odds model fits less well than the more complex multinomial logit.

## Comparing mod\_p1 and m1\_multi

glance(mod\_p1) |>   
 gt() |> tab\_options(table.font.size = 20)

| edf | logLik | AIC | BIC | deviance | df.residual | nobs |
| --- | --- | --- | --- | --- | --- | --- |
| 5 | -450.4815 | 910.9629 | 932.3273 | 900.9629 | 525 | 530 |

glance(m1\_multi) |>   
 gt() |> tab\_options(table.font.size = 20)

| edf | deviance | AIC | nobs |
| --- | --- | --- | --- |
| 8 | 890.8867 | 906.8867 | 530 |

BIC(mod\_p1); BIC(m1\_multi)

[1] 932.3273

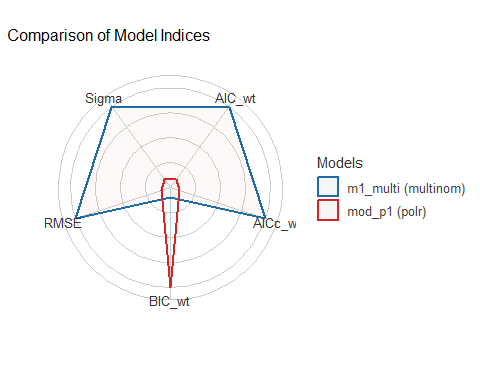
[1] 941.0697

## What to do in light of these results…

* A *p* value isn’t usually the best way to assess the proportional odds assumption, but it does provide some evidence of model adequacy.
* The stronger BIC for our POLR model relative to the multinomial gives conflicting advice.
  + One alternative: fit the multinomial model instead.
  + Another: fit a check of residuals (see Harrell’s RMS text.)
  + Another: fit a different model for ordinal regression. For example, orm in the rms package. (Coming soon.)

## Comparing mod\_p1 and m1\_multi

plot(compare\_performance(mod\_p1, m1\_multi))



## Classification Tables

addmargins(table(predict(mod\_p1), gradschool$apply,   
 dnn = c("predicted", "actual")))

actual  
predicted unlikely somewhat likely very likely Sum  
 unlikely 264 112 29 405  
 somewhat likely 39 60 25 124  
 very likely 0 0 1 1  
 Sum 303 172 55 530

addmargins(table(predict(m1\_multi), gradschool$apply,   
 dnn = c("predicted", "actual")))

actual  
predicted unlikely somewhat likely very likely Sum  
 unlikely 265 115 36 416  
 somewhat likely 34 57 16 107  
 very likely 4 0 3 7  
 Sum 303 172 55 530

* Which looks better?

# Using lrm for Proportional Odds Logistic Regression

## Using lrm to work through this model

d <- datadist(gradschool); options(datadist = "d")  
  
mod <- lrm(apply ~ pared + public + gpa, data = gradschool, x = T, y = T)  
  
mod

Logistic Regression Model  
  
lrm(formula = apply ~ pared + public + gpa, data = gradschool,   
 x = T, y = T)  
  
 Model Likelihood Discrimination Rank Discrim.   
 Ratio Test Indexes Indexes   
 Obs 530 LR chi2 74.22 R2 0.155 C 0.684   
 unlikely 303 d.f. 3 R2(3,530)0.126 Dxy 0.369   
 somewhat likely172 Pr(> chi2) <0.0001 R2(3,412.3)0.159 gamma 0.369   
 very likely 55 Brier 0.216 tau-a 0.206   
 max |deriv| 5e-09   
  
 Coef S.E. Wald Z Pr(>|Z|)  
y>=somewhat likely -3.8728 0.5721 -6.77 <0.0001   
y>=very likely -5.9413 0.6063 -9.80 <0.0001   
pared 1.1525 0.2184 5.28 <0.0001   
public -0.4949 0.2195 -2.25 0.0242   
gpa 1.1416 0.1850 6.17 <0.0001

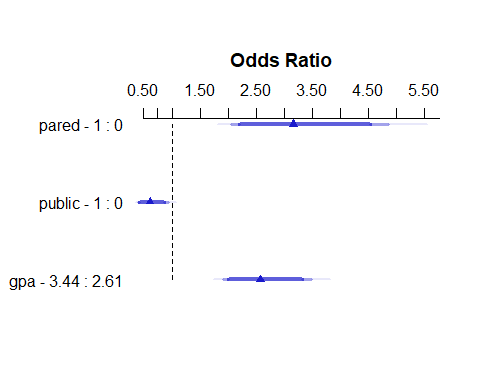
## Validating our mod

set.seed(432); validate(mod)

index.orig training test optimism index.corrected n  
Dxy 0.3687 0.3663 0.3646 0.0017 0.3670 40  
R2 0.1553 0.1528 0.1511 0.0018 0.1536 40  
Intercept 0.0000 0.0000 0.0231 -0.0231 0.0231 40  
Slope 1.0000 1.0000 1.0170 -0.0170 1.0170 40  
Emax 0.0000 0.0000 0.0078 0.0078 0.0078 40  
D 0.1382 0.1359 0.1340 0.0019 0.1363 40  
U -0.0038 -0.0038 -0.4637 0.4599 -0.4637 40  
Q 0.1419 0.1397 0.5978 -0.4581 0.6000 40  
B 0.2155 0.2136 0.2171 -0.0035 0.2190 40  
g 0.8954 0.8833 0.8814 0.0019 0.8934 40  
gp 0.2004 0.1958 0.1975 -0.0016 0.2021 40

## Effects Plot

plot(summary(mod))



## Effects Summary

summary(mod)

Effects Response : apply   
  
 Factor Low High Diff. Effect S.E. Lower 0.95 Upper 0.95  
 pared 0.00 1.00 1.00 1.15250 0.21843 0.72436 1.580600   
 Odds Ratio 0.00 1.00 1.00 3.16600 NA 2.06340 4.857900   
 public 0.00 1.00 1.00 -0.49486 0.21951 -0.92509 -0.064629   
 Odds Ratio 0.00 1.00 1.00 0.60966 NA 0.39650 0.937410   
 gpa 2.61 3.44 0.83 0.94756 0.15354 0.64662 1.248500   
 Odds Ratio 2.61 3.44 0.83 2.57940 NA 1.90910 3.485100

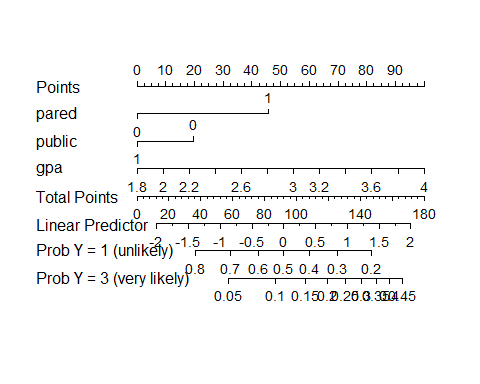
### Coefficients in the mod equation

mod$coef

y>=somewhat likely y>=very likely pared public   
 -3.872786 -5.941317 1.152479 -0.494859   
 gpa   
 1.141633

## Nomogram of mod

fun.1 <- function(x) 1 - plogis(x)  
fun.3 <- function(x)   
 plogis(x - mod$coef[1] + mod$coef[2])  
  
plot(nomogram(mod,  
 fun=list('Prob Y = 1 (unlikely)' = fun.1,   
 'Prob Y = 3 (very likely)' = fun.3)))



1. <http://stats.idre.ucla.edu/r/dae/ordinal-logistic-regression/> [↑](#footnote-ref-22)