432 Class 17

https://thomaselove.github.io/432-2023/

2023-03-07

Today's Topic

Regression Models for Ordered Multi-Categorical Outcomes

- Applying to Graduate School: A First Example
- Proportional Odds Logistic Regression Models
  - Using `polr`
  - 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.

Setup

```
knitr::opts_chunk$set(comment=NA)
options(width = 60)

library(janitor)
library(GGally)
library(scales)
library(knitr)
library(rms)
library(MASS)
library(nnet)
library(tidyverse)

theme_set(theme_bw())
```

Section 1

Applying to Graduate School

These are **simulated** data

This is a simulated data set of 530 students, looking at factors that influence the decision of whether to apply to graduate school.

College juniors are asked if they are unlikely, somewhat likely, or very likely to apply to graduate school. Hence, our outcome variable has three categories. Data on parental educational status, whether the undergraduate institution is public or private, and current GPA is also collected. The researchers have reason to believe that the "distances" between these three points are not equal. For example, the "distance" between "unlikely" and "somewhat likely" may be shorter than the distance between "somewhat likely" and "very likely".

```
gradschool <-
  read_csv("c17/data/gradschool_new.csv",
           show_col_types = FALSE) |>
  type.convert(as.is = FALSE) |>
  clean_names()
```

The `gradschool` data and my **Source**

The **gradschool** example is adapted from this UCLA site.

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

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)

Ensuring that our outcome is 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
```

The `gradschool` tibble

```
gradschool
```

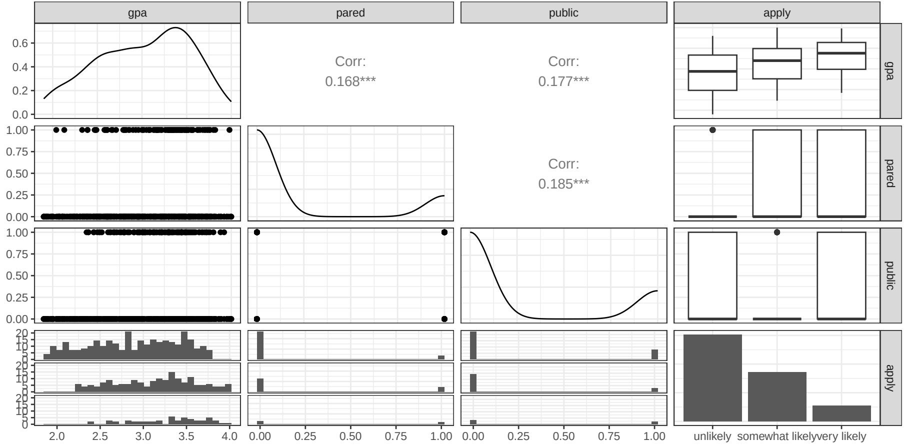
```
# A tibble: 530 x 5
   student apply           pared public   gpa
   <fct>   <ord>           <int>  <int> <dbl>
 1 A001    very likely         0      0  3.41
 2 A002    unlikely            0      0  2.38
 3 A003    somewhat likely     0      0  3.35
 4 A004    unlikely            0      1  3.45
 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
# ... with 520 more rows
```

Numerical Description of `gradschool` data

```
> describe(gradschool)
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      somewhat likely        unlikely     very likely
Frequency              172             303              55
Proportion           0.325           0.572           0.104
------------------------------------------------------------------------
pared
       n  missing distinct     Info      Sum     Mean      Gmd
     530        0        2     0.47      103   0.1943   0.3137

------------------------------------------------------------------------
public
       n  missing distinct     Info      Sum     Mean      Gmd
     530        0        2    0.555      130   0.2453   0.3709

------------------------------------------------------------------------
gpa
       n  missing distinct     Info     Mean      Gmd      .05      .10
     530        0      186        1    3.015   0.5919    2.104    2.279
     .25      .50      .75      .90      .95
   2.610    3.080    3.440    3.660    3.760

lowest : 1.90 1.91 1.92 1.93 1.94, highest: 3.95 3.97 3.98 3.99 4.00
```

Scatterplot Matrix (run with `#| message: FALSE`)

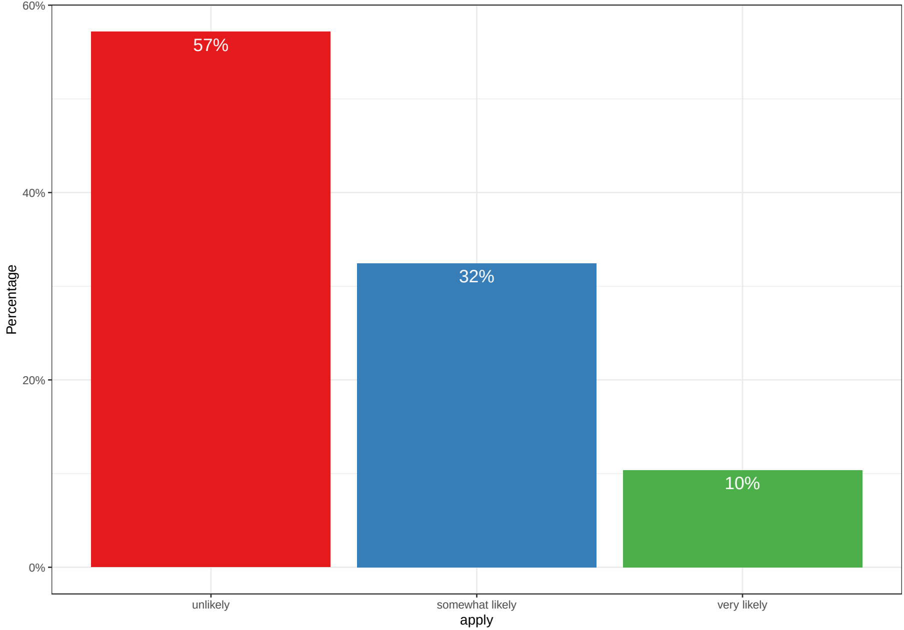
```r
ggpairs(gradschool |> select(gpa, pared, public, apply))
```



Bar Chart of apply classifications with %s (code)

```
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")
```

Bar Chart of `apply` classifications with %s (result)



Data (besides gpa) as Cross-Tabulation

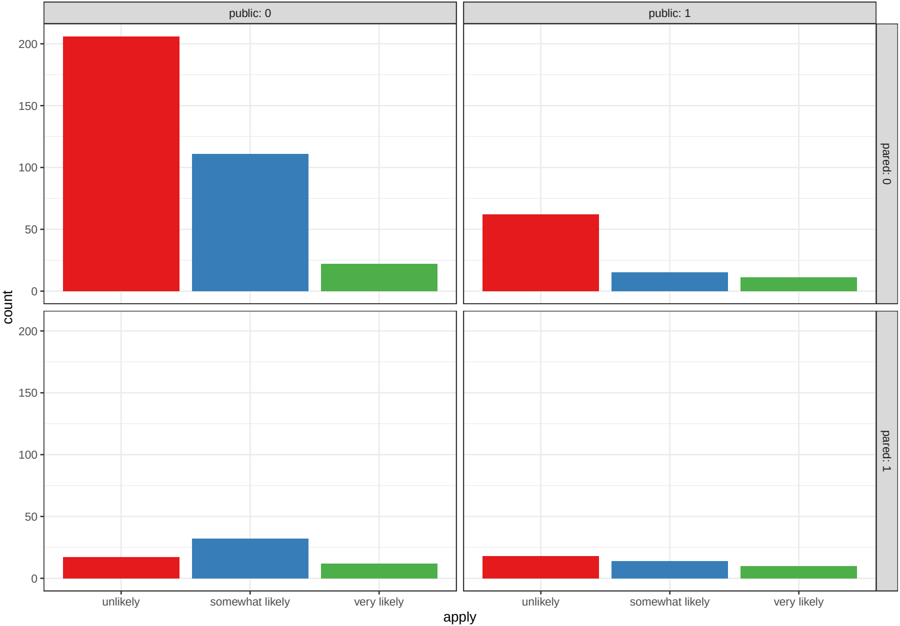
```
ftable(xtabs(~ public + apply + pared, data = gradschool))
```

```
                       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 (code)

```
ggplot(gradschool, aes(x = apply, fill = apply)) +
    geom_bar() +
    scale_fill_brewer(palette = "Set1") +
    guides(fill = "none") +
    facet_grid(pared ~ public, labeller = "label_both")
```

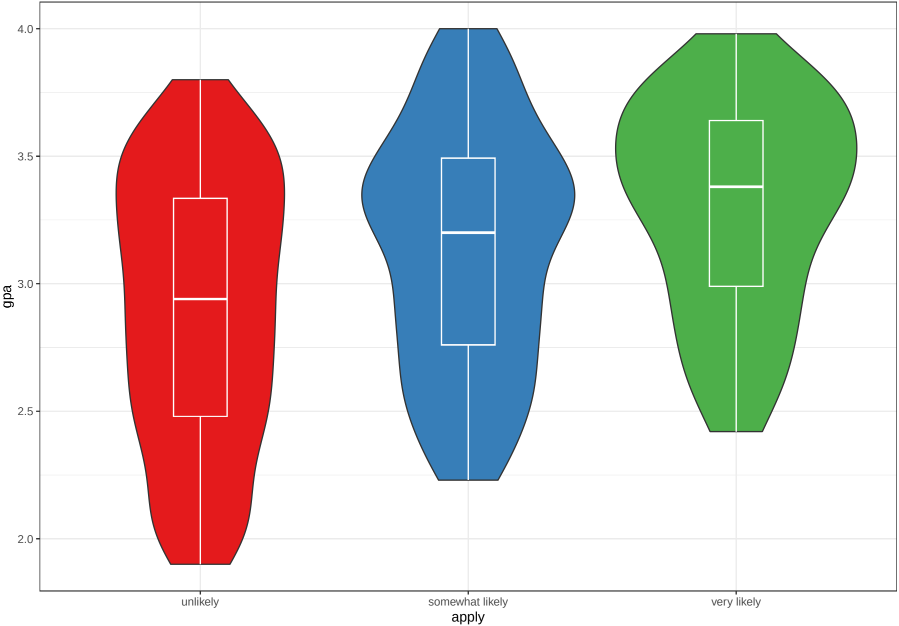
apply percentages by public, pared



Breakdown of gpa by apply (code)

```r
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")
```

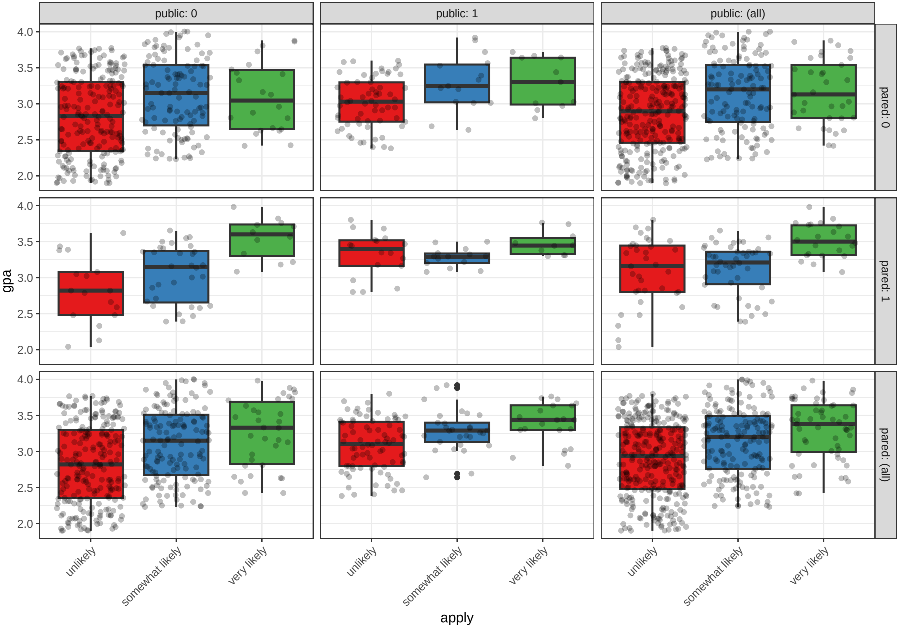
Breakdown of gpa by apply



gpa by three other variables (code)

```r
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))
```

gpa by three other variables



Section 2

Proportional Odds Logit Model via `polr`

Fitting the POLR model with MASS::polr

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.

polr uses the standard formula interface in R for specifying a regression model with outcome followed by predictors. We also specify Hess=TRUE to have the model return the observed information matrix from optimization (called the Hessian) which is used to get standard errors.

Obtaining Predicted Probabilities from `mod_p1`

To start we'll obtain predicted probabilities, which are usually the best way to understand the model.

For example, we can 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))
```

Obtaining Predicted Probabilities from mod_p1

Now, make predictions using model mod_p1:

```
newdat_p1 <- cbind(newdat,
                   predict(mod_p1, newdat, type = "probs"))
head(newdat_p1) |> kable(digits = 3)
```

pared	public	gpa	unlikely	somewhat likely	very likely
0	0	1.900	0.846	0.132	0.022
1	0	1.921	0.629	0.302	0.069
0	0	1.942	0.840	0.137	0.024
1	0	1.964	0.617	0.310	0.073
0	0	1.985	0.833	0.142	0.025
1	0	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...

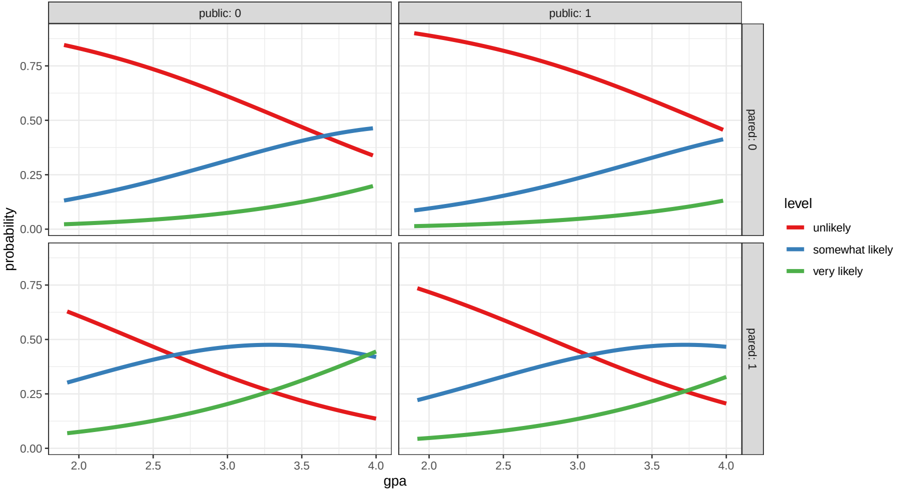
The newdat_long data

```
# A tibble: 1,200 x 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
# ... with 1,190 more rows
```

Plot the prediction results... (code)

```
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")
```

Plot the prediction results...



Cross-Tabulation of Predicted/Observed Classifications

Predictions in the rows, Observed in the columns

```
addmargins(table(predict(mod_p1), gradschool$apply))
```

```
                 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.

Describing the Proportional Odds Logistic Model

Our outcome, `apply`, has three levels. Our model has two logit 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 to estimate the three categories, since Pr(`apply` ≤ 3) $= 1$, because "very likely" is the maximum category for `apply`.

Parameters of the POLR Model

- The parameters to be fit include two intercepts:
  - ζ_1 will be the `unlikely|somewhat likely` parameter
  - ζ_2 will be the `somewhat likely|very likely` parameter

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

- The two logistic equations that will be fit differ only by their intercepts.

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
```

Understanding the Model

$$logit[Pr(apply \leq 1)] = \zeta_1 - \beta_1 pared - \beta_2 public - \beta_3 gpa$$

$$logit[Pr(apply \leq 2)] = \zeta_2 - \beta_1 pared - \beta_2 public - \beta_3 gpa$$

in general. In our setting, we have …

The `mod_p1` equations...

$$logit[Pr(apply \leq unlikely)] = 3.87 - 1.15 pared - (-0.49) public - 1.14 gpa$$

and

$$logit[Pr(apply \leq somewhat)] = 5.94 - 1.15 pared - (-0.49) public - 1.14 gp$$

confint(mod_p1)

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

```
             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

```r
exp(coef(mod_p1))
```

```
    pared    public       gpa 
3.1660446 0.6096623 3.1318247 
```

```r
exp(confint(mod_p1))
```

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

Interpreting the Exponentiated 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.
- 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`?

Comparison to a Null Model

```r
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
1                    1       528   975.1828
2 pared + public + gpa       525   900.9629 1 vs 2     3
  LR stat.      Pr(Chi)
1
2 74.21989 5.551115e-16
```

AIC and BIC are available, too

We could also compare model `mod_p1` to the null model `mod_p0` with AIC or BIC.

```
AIC(mod_p1, mod_p0)
```

```
       df      AIC
mod_p1  5 910.9629
mod_p0  2 979.1828
```

```
BIC(mod_p1, mod_p0)
```

```
       df      BIC
mod_p1  5 932.3273
mod_p0  2 987.7286
```

Testing the Proportional Odds Assumption

One way to test 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.

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

Again, the multinomial model is fit here using `multinom()` from the **nnet** package…

```
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 = gradsc

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
```

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 AIC and BIC

```
AIC(mod_p1)
```

```
[1] 910.9629
```

```
AIC(m1_multi)
```

```
[1] 906.8867
```

```
BIC(mod_p1)
```

```
[1] 932.3273
```

```
BIC(m1_multi)
```

```
[1] 941.0697
```

What to do in light of these results...

- A non-significant *p* value here isn't always the best way to assess the proportional odds assumption, but it does provide some evidence of model adequacy.
- The stronger BIC (and only slightly worse AIC) for our POLR model relative to the multinomial gives us some conflicting advice.
  - One alternative would be to fit the multinomial model instead.
  - Another would be to fit a check of residuals (see Frank Harrell's RMS text.)
  - Another would be to fit a different model for ordinal regression. Several are available (check out `orm` in the `rms` package, for instance.)

Section 3

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)
```

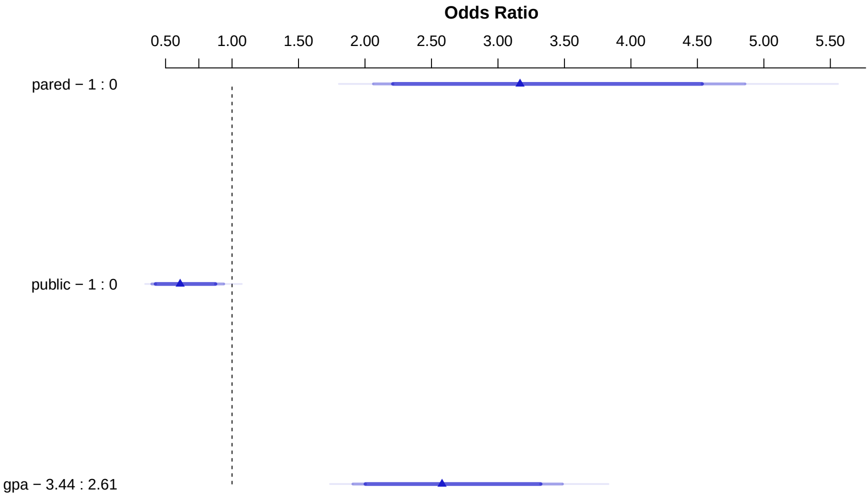
mod output

```
> 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    g        0.895    Dxy     0.369
  somewhat likely172       Pr(> chi2) <0.0001   gr       2.448    gamma   0.369
    very likely     55                          gp       0.200    tau-a   0.206
   max |deriv| 5e-09                            Brier    0.216

                      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
```

summary(mod)

```
             Effects              Response : apply

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

plot(summary(mod))



Coefficients in our equation

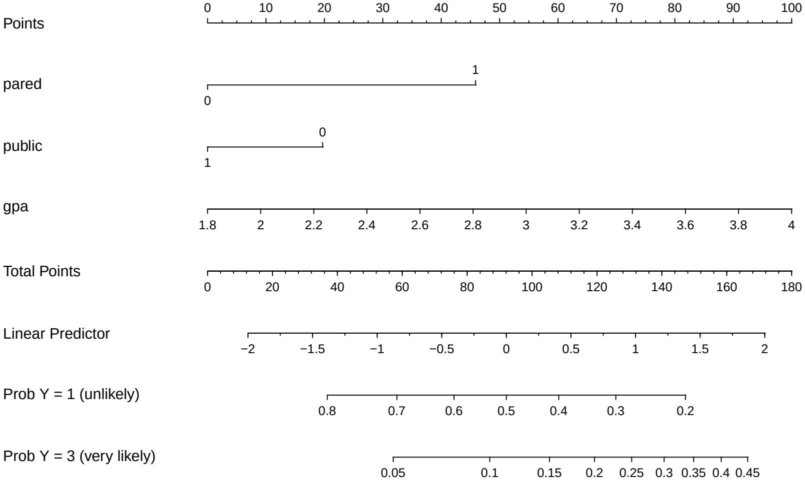
```r
mod$coef
```

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

Nomogram of mod (code)

```
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)))
```

Nomogram of `mod` (result)



set.seed(432); validate(mod)

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

More to come.

We'll share another example of ordinal logistic regression before we're done.