PPOL561 | Summary
Ordered Dependent Variables

Overview

The following offers a summary of main points when dealing with ordered dependent variables. These notes summarize some of the main points from the Long reading and the lectures.

Ordered Responses

Ordered dependent variables are outcomes that retain some inherent but discrete ordering. These sorts of dependent variables are encountered often in survey data (e.g. "Strongly Disagree", "Disagree", "Somewhat Disagree", "Somewhat Agree", ...).

The distance between each category is of different unknown sizes. For example, "Somewhat Disagree" or "Somewhat Agree" may be very close (i.e. there is little difference between the two choices on the scale), or may be far apart (i.e. there is a huge difference between agreeing or disagreeing even somewhat)

More example of these types of ordered scales:

- Likert Scales: disagree, neutral, agree
- Policy options: privatize social security, paritally privatize, leave unchanged
- Ranks: some high School, high school grad, some college, college grad, etc.

One thing to keep in mind is that just because an outcome *can* be treated as an ordinal variable, doesn't mean it *should* be analyzed as an ordinal variable. When the proper ordering of a variable is ambiguous, multinomial models should be considered.

Why not OLS

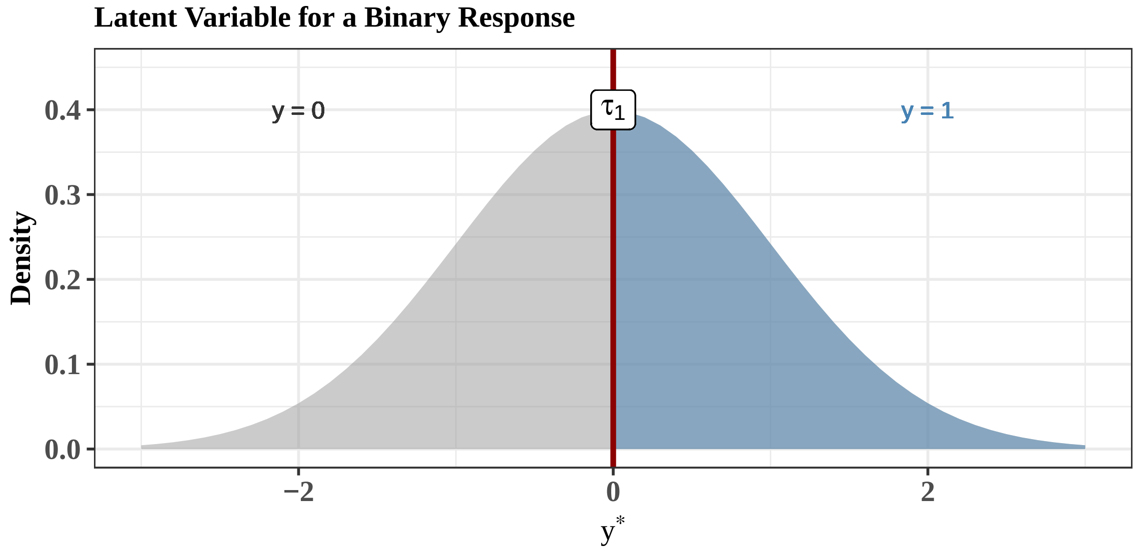
- Encounter the same problems as when using regression with a binary outcome.
  - heteroskedastic
  - non-sensical predictions: predictions for y can fall outside a plausible range.
- Requires us to assume that the distance between categories is equal. That is, a one unit change moves us from one category to the next at an equal rate.
- Difficult to interpret: what is a unit change in x with respect to y? E.g. if y is an ordered outcome in the model $y = \beta_0 + \beta_1 x + \epsilon$, what does $beta_1 = 1.2$ mean? A change in x corresponds with a 1.2 change in a category ranking?

Ordered Dependent Variable Model

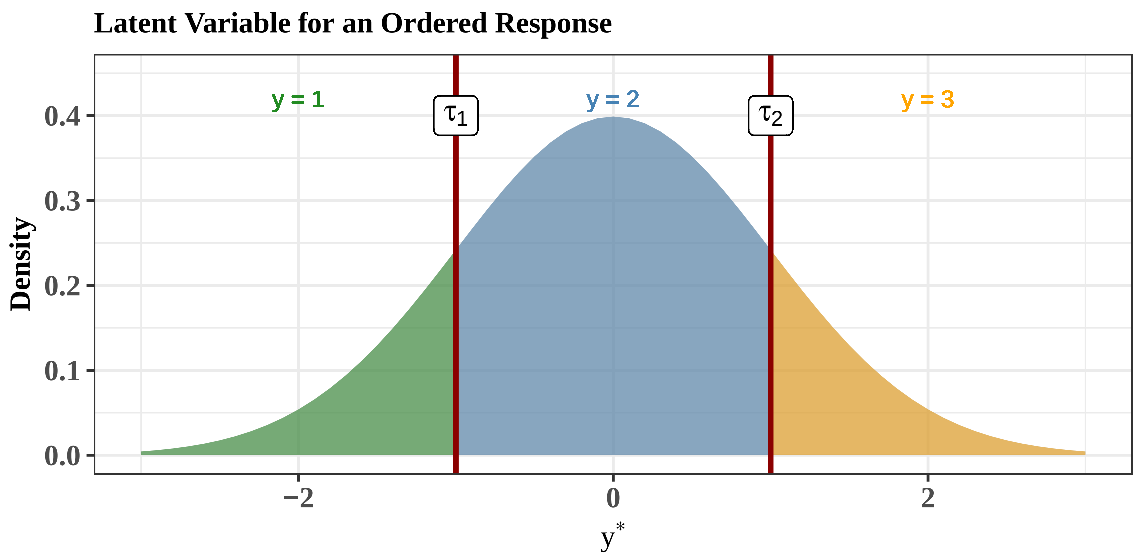
When discussing binary models, we spoke about things in terms of a **latent variable** y^*.

$$y_i^* = \beta_0 + \beta_1 x_i + \epsilon_i$$

When y^* is above a threshold τ_1 we observe $y = 1$, otherwise we observe $y = 0$. In this model, we assumed that this cutpoint was 0 ($\tau_1 = 0$).



Let's now consider the relationship between some observed variable $y \in \{1, 2, 3\}$ (i.e. some ordered outcome as outlined above) and some underlying latent variable $y*$.



The probability of observing a specific response in the ordering depends on which cut points y^* falls in between.

- when $y^* \leq \tau_1$, we observe $y = 1$

- when $\tau_1 < y^* \leq \tau_2$, we observe $y = 2$
- when $\tau_2 < y^*$, we observe $y = 3$

Note that the the upper and lower bounds are set to $\tau_0 = -\infty$ and $\tau_3 = \infty$. When we have *three* possible ordered outcomes (as is the case in the illustration), we need to estimate *two* cutpoints (τ_1 & τ_2).

Assumptions

For an ordered probit/logit, we need to impose a constraint on the model so that we can identify it. We either need to set the **intercept (β_0) to 0**, or set the τ_0 **(our $-\infty$ cut point) to 0**. Doing this imposes a constrain on one of the parameters, which is necessary to estimate the model. Note that the `polr()` function from the `MASS` package—which we'll use to estimate a ordered probit and logit model—opts to set the intercept to 0 by default, so no intercept will be reported.

Finally, when running the an ordered logit or probit model, we inherently make an assumption about "**parallel regressions**". This means that the effect of β_1 is the same across all cutpoint ranges, i.e. the effect of x on y does not change when we move from one threshold to the next. Put differently, we get one set of coefficients and we assume the same relationship for each pair of outcome categories.

Predicted probabilities

Assuming a probit model, we can estimate the predicted probabilities of any observed outcome as follows. The key is to keep track of where the cutpoints fall.

- $pr(y_i = 1|x_i) = \Phi(\tau_1 - \beta_1 x_i) - 0$
- $pr(y_i = 2|x_i) = \Phi(\tau_2 - \beta_1 x_i) - \Phi(\tau_1 - \beta_1 x_i)$
- $pr(y_i = 3|x_i) = 1 - \Phi(\tau_2 - \beta_1 x_i)$

Recall that this is a *probability distribution*, where the bounds fall between 0 and 1. Thus, why the lower bound in the $pr(y_i = 1|x_i)$ calculation is 0 and the upper bound in the $pr(y_i = 3|x_i)$ calculation is 1.

Say, for example, $\tau_1 = -1$, $\tau_2 = 1$, $\beta_1 = .05$. We can calculate the predicted probabilities as follows for $x = 15$

- $pr(y_i = 1|x = 15) = \Phi((-1) - (.05)(15)) = .04$
- $pr(y_i = 2|x = 15) = \Phi((1) - (.05)(15)) - \Phi((-1) - (.05)(15)) = .56$
- $pr(y_i = 3|x = 15) = 1 - \Phi((1) - (.05)(15)) = .40$

In this example, for an observation where $x = 15$, the probability that observed response will be 1 is 4%, 2 is 56%, and 3 is 40%.

Example: Student Support for the Iraq War

Consider the following survey data draw from a sample of 500 college students in 2002 leading up to the Iraq war. The survey asked whether students agreed with the United States entering into the Iraq War. The data contains three variables:

```
dat <- read_csv('student_vote.csv')
dat$warsup = factor(dat$warsup,levels=c('strongly oppose',
                                        "somewhat oppose",
                                        "somewhat support",
                                        "strongly support"))
summary(dat)
```

```
##               warsup          dem             female     
##  strongly oppose :168   Min.   :0.000   Min.   :0.000  
##  somewhat oppose : 98   1st Qu.:0.000   1st Qu.:0.000  
##  somewhat support:126   Median :1.000   Median :1.000  
##  strongly support:108   Mean   :0.536   Mean   :0.576  
##                         3rd Qu.:1.000   3rd Qu.:1.000  
##                         Max.   :1.000   Max.   :1.000
```

- `warsup`: whether the respondent somewhat or strongly oppose/supported the War in Iraq
- `dem`: 1 if the repondent was a democrat, 0 otherwise.
- `female`: 1 if the repondent is female, 0 otherwise.

The dependent variable is `warsup`, and the key independent variable that we're going to look at is `dem`. Our hypothesis is that democratic respondents are less likely to support the Iraq War than non-democratic respondents.

We can estimated the model using the `polr` function from the `MASS` package.

```
war_ordered <- MASS::polr(warsup ~ dem + female, # Model
                          method = "probit", # Probit model
                          Hess = T, # returns the hessian matrix
                          data = dat)
summary(war_ordered,digits = 2)
```

```
## Call:
## MASS::polr(formula = warsup ~ dem + female, data = dat, Hess = T,
##     method = "probit")
##
## Coefficients:
##        Value Std. Error t value
## dem    -1.50       0.11   -13.7
## female -0.18       0.10    -1.7
##
```

```
## Intercepts:
##                                    Value  Std. Error t value
## strongly oppose|somewhat oppose    -1.49   0.11     -13.52
## somewhat oppose|somewhat support   -0.83   0.10      -8.16
## somewhat support|strongly support   0.13   0.10       1.34
##
## Residual Deviance: 1159.039
## AIC: 1169.039
```

Note that the model estimates three cutpoints: τ_1 as `strongly oppose|somewhat oppose`, τ_2 as `somewhat oppose|somewhat support`, and τ_3 as `somewhat support|strongly support`.

We interpret the model as we did with the binary response. That is, they are changes in the log odds. We cannot say much about the marginal effect of `dem` variable, but we talk about the general direction of the coefficients and whether they are statistically significant. Here we see that the coefficient on `dem` is negative and statistically significant. The coefficient on `female` is also negative but is not statistically signficant.

Calculating Predicted Probabilities

To calculate the predicted probability of supporting the war given one's political affiliation, we'll need to manipulate the `dem` variable and calculate the predicted probabilities across each cutpoint.

First, let's extract the coefficients and the cut points

```
B = war_ordered$coefficients
B
```

```
##        dem     female
## -1.4998994 -0.1786577
```

```
cuts = war_ordered$zeta
cuts
```

```
##   strongly oppose|somewhat oppose  somewhat oppose|somewhat support
##                        -1.4893715                        -0.8277817
## somewhat support|strongly support
##                         0.1277792
```

Second, let's extract the model matrix (i.e. the data we used to run the model with). We drop the intercept because the intercept is constrained to equal 0 to identify the model.

```
X = model.matrix(war_ordered)[,-1] # drop intercept
head(X)
```

```
##   dem female
## 1   0      0
```

```
## 2    0       1
## 3    0       1
## 4    1       1
## 5    0       0
## 6    0       0
```

Third, maniputate the value of `dem` to be 0 (i.e. the respondent is a republican or independent) and hold all other variables at their **observed values**.

```
X[,1] = 0
```

Fourth, calculate the predicted probabilities for each observation as each cut point

```
pr_strong_opp = pnorm(cuts[1] - X%*%B)
pr_some_opp = pnorm(cuts[2] - X%*%B) - pnorm(cuts[1] - X%*%B)
pr_some_supp = pnorm(cuts[3] - X%*%B) - pnorm(cuts[2] - X%*%B)
pr_strong_supp = 1 - pnorm(cuts[3] - X%*%B)
```

Fifth, calculate the expect (average) probability by taking the mean, and then present as a table.

```
predicted_probs_ndem <-
  tibble(response = c('strongly oppose',
                      "somewhat oppose",
                      "somewhat support",
                      "strongly support"),
         Democrat = 0,
         prob = c(mean(pr_strong_opp),
                  mean(pr_some_opp),
                  mean(pr_some_supp),
                  mean(pr_strong_supp)))
predicted_probs_ndem
```

```
## # A tibble: 4 x 3
##   response         Democrat   prob
##   <chr>               <dbl>  <dbl>
## 1 strongly oppose         0 0.0836
## 2 somewhat oppose         0 0.152
## 3 somewhat support        0 0.356
## 4 strongly support        0 0.409
```

Let's now run through the same steps to calculate the predicted probabilities of being a democrat (`dem = 1`).

```
# Manipulate
X[,1] = 1

# Prediction given cut points
```

```r
pr_strong_opp = pnorm(cuts[1] - X%*%B)
pr_some_opp = pnorm(cuts[2] - X%*%B) - pnorm(cuts[1] - X%*%B)
pr_some_supp = pnorm(cuts[3] - X%*%B) - pnorm(cuts[2] - X%*%B)
pr_strong_supp = 1 - pnorm(cuts[3] - X%*%B)

# Arrange in table
predicted_probs_dem <-
  tibble(response = c('strongly oppose',
                      "somewhat oppose",
                      "somewhat support",
                      "strongly support"),
         Democrat = 1,
         prob = c(mean(pr_strong_opp),
                  mean(pr_some_opp),
                  mean(pr_some_supp),
                  mean(pr_strong_supp)))
predicted_probs_dem
```

```
## # A tibble: 4 x 3
##   response         Democrat   prob
##   <chr>               <dbl>  <dbl>
## 1 strongly oppose         1 0.545
## 2 somewhat oppose         1 0.235
## 3 somewhat support        1 0.178
## 4 strongly support        1 0.0424
```

Print Table.

```r
pprobs <- bind_rows(predicted_probs_dem,predicted_probs_ndem)
pprobs
```

```
## # A tibble: 8 x 3
##   response         Democrat   prob
##   <chr>               <dbl>  <dbl>
## 1 strongly oppose         1 0.545
## 2 somewhat oppose         1 0.235
## 3 somewhat support        1 0.178
## 4 strongly support        1 0.0424
## 5 strongly oppose         0 0.0836
## 6 somewhat oppose         0 0.152
## 7 somewhat support        0 0.356
## 8 strongly support        0 0.409
```

Conclusion: Democratic respondents had a 55% probability of being strongly against the Iraq War whereas their non-democratic counterparts had only a 8% probability of strongly opposing the war. By contrast, Non-democratic respondents had a 41% probability of supporting the War in Iraq. Clearly support for the war fell along partisan lines.

Simulating Confidence Intervals

One issue with the above predictions is that we have no sense of how certain we are about these predictions. As we did with binary response models, we can estimated our uncertainty around our predictions via monte carlo simulation.

Below I'll use the `obsval` package to calculate the 95% confidence interval around the predicted effects. The process of doing this manually is a little involved as we saw last time with binary responses, but it follows the same basic steps (try doing it for yourself!)

```r
require(obsval) # Load the package

# Re-estimate the model using obsval
war_ordered2 <-
  obsval(warsup ~ dem + female,
         data = dat,
         ci = .95,
         n.draws = 1000,
         reg.model = "oprobit",
         effect.var = "dem",
         effect.vals = c(0,1))

# Check the model summary to make sure everything looks right
summary(war_ordered2$model)
```

```
## Call:
## MASS::polr(formula = fmla, data = data, Hess = TRUE, method = "probit")
##
## Coefficients:
##          Value Std. Error t value
## dem    -1.4999     0.1097 -13.675
## female -0.1787     0.1024  -1.745
##
## Intercepts:
##                                    Value    Std. Error t value
## strongly oppose|somewhat oppose    -1.4894   0.1101   -13.5230
## somewhat oppose|somewhat support   -0.8278   0.1014    -8.1609
## somewhat support|strongly support   0.1278   0.0953     1.3412
##
## Residual Deviance: 1159.039
## AIC: 1169.039
```

Let's summarized the predicted effects. Check out the dimensions of the output: it's stored as an array! 1000 simulations by 4 possible ordered outcome categories by 2 effect conditions (democrat or not). We'll need to clean this up…

```
dim(war_ordered2$preds)
```

```
## [1] 1000    4    2
```

Extract out the upper and lower bound that we want.

```
# Function extracts the 2.5%, mean, and 97.5% interval
extract = function(x) c(quantile(x,.025),ave_pred=mean(x),quantile(x,.975))

# Apply to systematically draw it
non_dem = apply(war_ordered2$preds[,,1],2,extract)
dem = apply(war_ordered2$preds[,,2],2,extract)
```

Reformat as a table.

```
non_dem = data.frame(t(non_dem)) %>%
  rownames_to_column("category") %>%
  mutate(democrat = 0)
dem = data.frame(t(dem)) %>%
  rownames_to_column("category") %>%
  mutate(democrat = 1)
predicted_effects = bind_rows(dem,non_dem)
predicted_effects
```

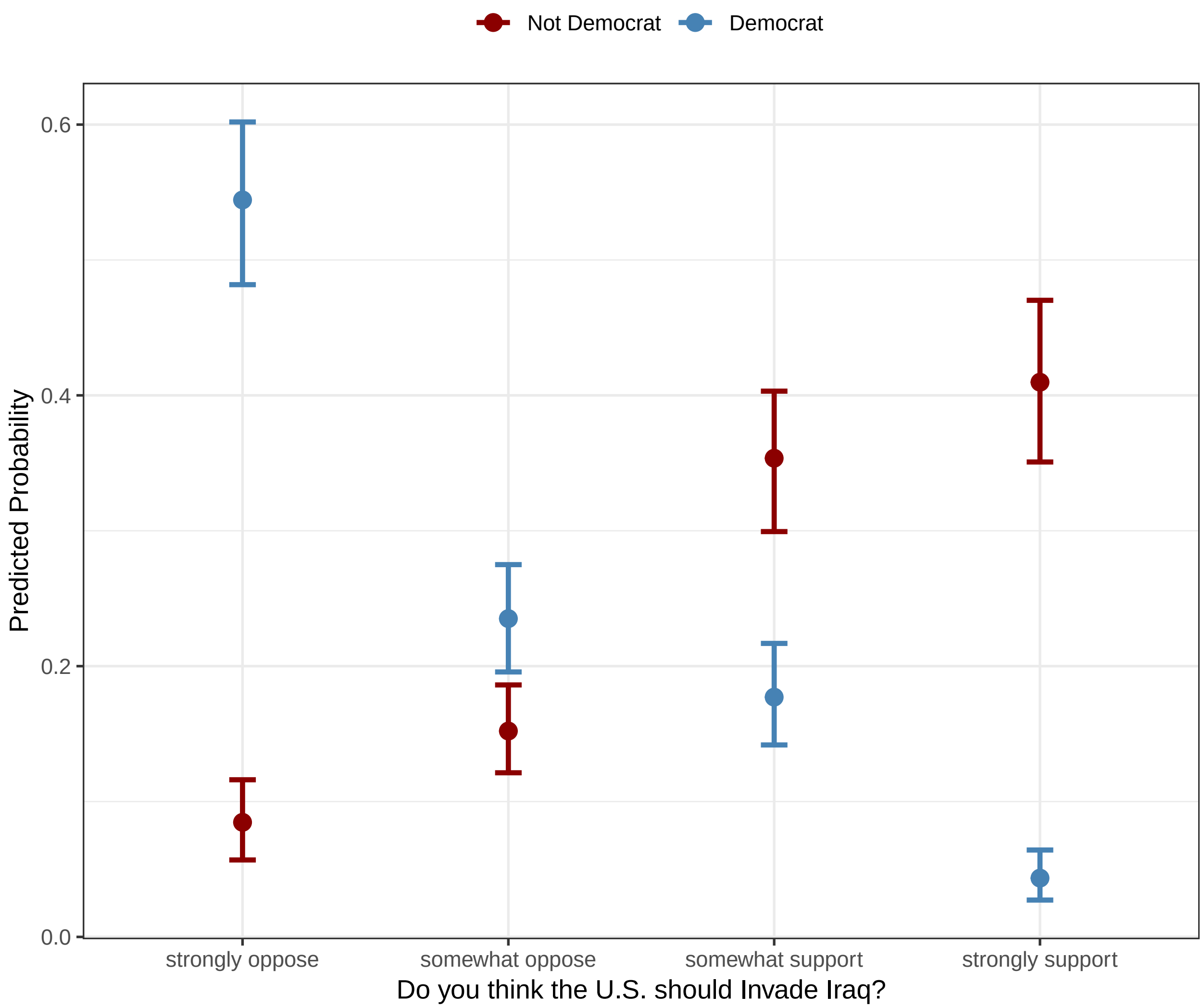
```
##           category      X2.5.   ave_pred     X97.5. democrat
## 1  strongly oppose 0.48175780 0.54433086 0.60191610        1
## 2  somewhat oppose 0.19571675 0.23511298 0.27498620        1
## 3 somewhat support 0.14180100 0.17710785 0.21680088        1
## 4 strongly support 0.02722750 0.04344832 0.06418976        1
## 5  strongly oppose 0.05685161 0.08463486 0.11607832        0
## 6  somewhat oppose 0.12123468 0.15207507 0.18612725        0
## 7 somewhat support 0.29938412 0.35356924 0.40311050        0
## 8 strongly support 0.35079880 0.40972083 0.47020427        0
```

Finally, let's visualize![1]

```
predicted_effects %>%
  mutate(category = factor(category,levels = levels(dat$warsup))) %>%
  ggplot(aes(category,ave_pred,color=factor(democrat))) +
  geom_point(size=3) +
  geom_errorbar(aes(x=category,ymin=`X2.5.`,ymax=`X97.5.`),
                width=.1,size=1) +
  labs(y="Predicted Probability",
       x="Do you think the U.S. should Invade Iraq?",
       color="") +
```

[1] Note that I'm playing with the factor levels in the below plot to ensure the right ordering when visualizing.

```r
  scale_color_manual(values=c("darkred","steelblue"),
                     labels=c("Not Democrat","Democrat")) +
  theme_bw() +
  theme(legend.position = "top")
```



As we can see, the predicted effects are all statistically significant. None of the confidence intervals switch signs, which means we can reject the Null Hypothesis that party affiliation (i.e. self-identifying as a democrat or not) has no effect. Moreover, we can see that the intervals are distinct (i.e. they don't overlap). This means that they are meaignfully different from one another. We could confirm this by taking the discrete difference of dem/not dem across each sub-category.