Gov 2001: Section 7

I. Ordered Categorial Variables
II. Binomial Variables
III. Diagnostics

Gov 2001

March 11, 2010

Ordered categorical variables

Suppose our dependent variable is an ordered scale. For example:

- Customers tell you how much they like your product on a 5-point scale from "a lot" to "very little."
- Voters identify their ideology on a 7-point scale: "very liberal," "moderately liberal," "somewhat liberal," "neutral," "somewhat conservative," "moderately conservative," and "very conservative."
- Survey respondents answer on a Likert scale from "strongly agree" to "strongly disagree."

We can use a generalization of the binary model to study these processes.

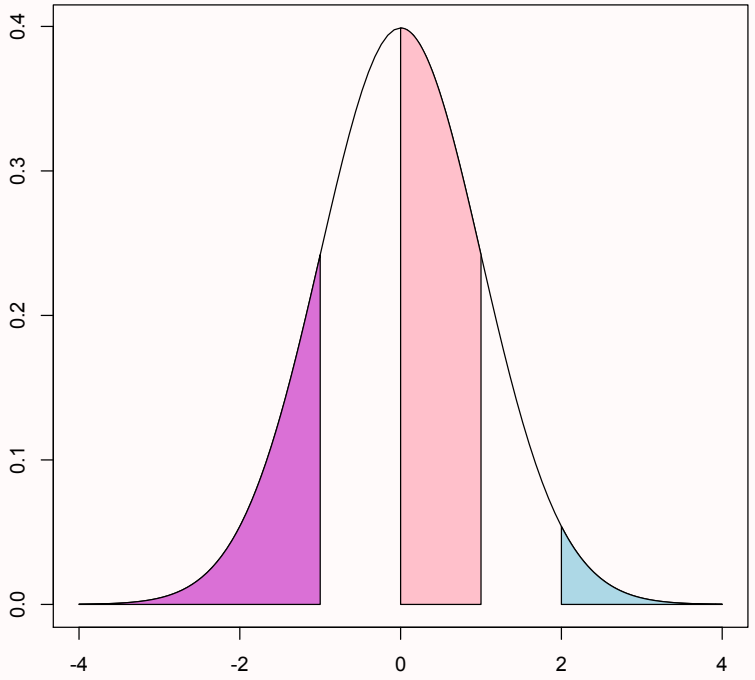
Ordered categorical variables (ctd)

How does this work?

- Suppose there is a latent (unobserved) data distribution, $Y^* \sim f_{stn}(y^*|\mu_i)$.
- This latent distribution has a systematic component, $\mu_i = x_i\beta$.
- Any realizations, y_i, are completely unobserved.
- What you *do* observe is whether y_i is between some threshold parameters.

Ordered categorical variables (ctd)

- Let's define the threshold parameters τ_j for $j = 1, ..., m$, such that $\tau_1 = -\infty$ and $\tau_m = \infty$
- Although y_i^* is unobserved, we do observe which of the m categories it falls into.



Ordered categorical variables (ctd)

Here's what we're working with:

$$y_{ij} = \begin{cases} 1 & \text{if } \tau_{j-i} < y_i^* \leq \tau_j \\ 0 & \text{otherwise} \end{cases}$$

You use this to derive the likelihood that y_i^* will fall into category j:

$$\begin{aligned} Pr(Y_{ji} = 1) &= Pr(\tau_{j-1} < y_i^* < \tau_j) \\ &= \int_{\tau_{j-1}}^{\tau_j} f(y_i^* | \mu_i) dy_i^* \\ &= F(\tau_j | x_i\beta) - F(\tau_{j-1} | x_i\beta) \end{aligned}$$

where F is the cumulative normal density with variance 1.

Ordered categorical variables (ctd)

But this is the likelihood of one observation falling in one of the categories. We want to generalize to all observations and all categories

$$L(\tau, \beta|y) = \prod_{i=1}^{n}\{\prod_{j=1}^{m}[F(\tau_j|x_i\beta) - F(\tau_{j-1}|x_i\beta)]^{y_{ji}}\}$$

where τ is a vector of threshold parameters that you'll have to estimate.

Then we take the log to get the log-likelihood

$$lnL(\tau, \beta|y) = \sum_{i=1}^{n}\sum_{j=1}^{m} y_{ji} ln[F(\tau_j|x_i\beta) - F(\tau_{j-1}|x_i\beta)]$$

Ordered categorical variables (ctd)

- We can operationalize this in `R`
- Let's do this using `Zelig`
- We'll use data on the cost of bilateral sanctions during 1939-1983 (from Martin 1992).

```
> data(sanction)
> head(sanction)
  mil coop target import export cost num          ncost
1   1    4      3      1      1    4  15    major loss
2   0    2      3      0      1    3   4   modest loss
3   0    1      3      1      0    2   1 little effect
4   1    1      3      1      1    2   1 little effect
5   0    1      3      1      1    2   1 little effect
6   0    1      3      0      1    2   1 little effect
```

The `ncost` variable here is an ordered categorical variable.

Ordered categorical variables (ctd)

We estimate the model using the `oprobit` call:

```
> z.out <- zelig(ncost ~ mil + coop,
      model = "oprobit", data = sanction)
```

Note that you could use `model = "ologit"` and get similar inferences.

Ordered categorical variables (ctd)

What does the output look like?

```
> z.out
Call:
zelig(formula = ncost ~ mil + coop, model = "oprobit", data = sanction)

Coefficients:
        mil        coop
-0.03531216  0.58713295

Intercepts:
   net gain|little effect little effect|modest loss    modest loss|major loss
                0.6979813                 2.2498275                 3.1082133

Residual Deviance: 153.5360
AIC: 163.5360
```

These are a little hard to interpret, so we turn to our bag of tricks...

Ordered categorical variables (ctd)

Suppose we want to compare the cost of sanctions when there is or is not military action addition to the sanction.

```
> x.low <- setx(z.out, mil = 0)
> x.high <- setx(z.out, mil = 1)
```

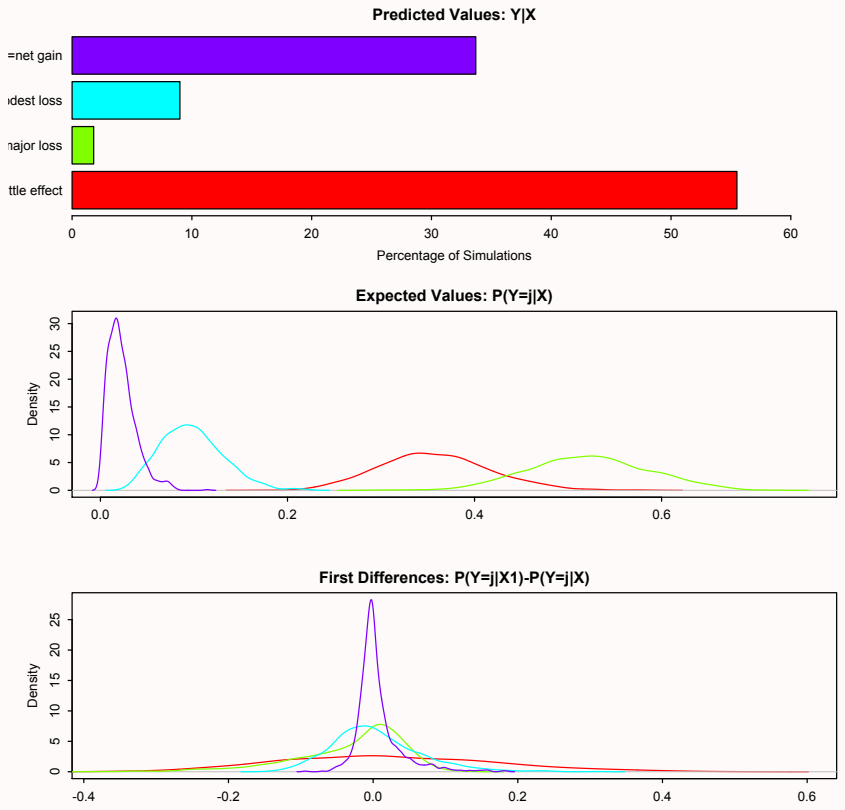
Ordered categorical variables (ctd)

Now we can simulate values using these hypothetical military involvements:

```
> s.out <- sim(z.out, x = x.low, x1 = x.high)
> summary(s.out)

Values of X
  (Intercept) mil     coop
1           1   0 1.807692

Values of X1
  (Intercept) mil     coop
1           1   1 1.807692

Expected Values: P(Y=j|X)
                    mean         sd        2.5%     97.5%
net gain      0.35728183 0.05914685 0.247131738 0.4797320
little effect 0.52201775 0.06381806 0.401616091 0.6471354
modest loss   0.09787398 0.03330800 0.042606284 0.1718923
major loss    0.02282644 0.01509349 0.002524672 0.0619769

Predicted Values: Y|X
  little effect major loss modest loss net gain
1         0.555      0.018        0.09    0.337

First Differences: P(Y=j|X1)-P(Y=j|X)
                      mean         sd        2.5%      97.5%
net gain       0.020420268 0.15043561 -0.24158255 0.34076705
little effect -0.030199046 0.07645313 -0.23259818 0.07282105
modest loss    0.003883799 0.06188179 -0.10067901 0.14675416
major loss     0.005894979 0.03026963 -0.03706656 0.08792679
```

Ordinal categorical variables (ctd)

And then you can use the `plot(s.out)` command to visualize



Gov 2001: Section 7

I. Ordered Categorial Variables
II. Binomial Variables
III. Diagnostics

Gov 2001

March 11, 2010

Binomial variables (ctd)

Suppose our dependent variable is the number of successes in a series of *independent* trials. For example:

- The number of heads in 10 coin flips.
- The number of times you voted in the last six elections.
- The number of Supreme Court cases the government won in the last ten decisions.

We can use a generalization of the binary model to study these processes.

Binomial variables

- Stochastic Component

$$\begin{aligned} Y_i &\sim Binomial(y_i|\pi_i) \\ P(Y_i = y_i|\pi_i) &= \binom{N}{y_i} \pi_i^{y_i} (1-\pi_i)^{N-y_i} \end{aligned}$$

- $\pi_i^{y_i}$: There are y_i successes each with probability of π_i
- $(1-\pi_i)^{N-y_i}$: There are $N - y_i$ failures each with probability $1 - \pi_i$
- $\binom{N}{y_i}$: Number of ways to distribute y_i successes in N trials; order of successes does not matter.

Binomial variables (ctd)

- Systematic Component

$$\pi_i = [1 + e^{-x_i\beta}]^{-1}$$

Binomial variables (Ctd)

- So we get the following likelihood

$$
\begin{aligned}
L(\pi_i|y_i) &= P(y_i|\pi_i) \\
&= \prod_{i=1}^{n} \binom{N}{y_i} \pi_i^{y_i} (1-\pi_i)^{N-y_i} \\
lnL(\pi_i|y_i) &= \sum_{i=1}^{n} \left[ln\binom{N}{y_i} + ln\pi_i^{y_i} + ln(1-\pi_i)^{N-y_i} \right] \\
&= \sum_{i=1}^{n} [y_i ln\pi_i + (N-y_i) ln(1-\pi_i)]
\end{aligned}
$$

Binomial variables (ctd)

We can operationalize this in R by coding the log likelihood up ourselves. First, let's make up some data to play with:

```
> x1 <- rnorm(1000,0,1)
> x2 <- rnorm(1000,9,.5)
> pi <- inv.logit(-5 + .4*x1 +.6*x2)
> y <- rbinom(1000,10,pi)
```

You can get the `inv.logit` command from the `boot` library.

Binomial variables (ctd)

Next, let's code up the log likelihood:

```
ll.binom <- function(par, N, X, y){
            pi <- 1/(1 + exp(-1*X%*%par))
            out <- sum(y * log(pi) + (N - y)*log(1-pi))
            return(out)
           }
```

Binomial variables (ctd)

And then we can run optim over this

```
> my.optim <- optim(par = c(0,0,0), fn = ll,
    y = y, X = cbind(1,x1,x2), N = 10,
    method = "BFGS", control=list(fnscale=-1), hessian=T)
> my.optim$par
[1] -4.6132799  0.3836413  0.5590359
```

Given that

```
> pi <- inv.logit(-5 + .4*x1 +.6*x2)
```

the output doesn't look too bad.

Binomial variables (Beta-Binomial specification)

But what if our trials are not independent? For example:

- The number of members of the House Appropriations Committee that vote yes on a bill.
- The number of OPEC members that want an increase in oil production.
- The number of Supreme Court Justices who vote in favor of the government.

These "trials" are in no way independent, so we must generalize our framework.

Binomial variables (Beta-Binomial specification, ctd)

- This model allows for dependence among the N trials and heterogeneity in π_i across the N trials.
- The Stochastic Component comes from the Beta-Binomial (see UPM 45-48)

$$Y_i \sim EBB(y_i|\pi_i, \gamma)$$

- where γ is an ancillary parameter
- The Systematic Component is the same

$$\pi_i = [1 + e^{-x_i\beta}]^{-1}$$

- Note that γ governs the degree to which π varies across the binary variables.
- When $\gamma = 0$, then the distribution reduces to binomial. Larger amounts of variation in π lead to larger values of γ.

Binomial variables (Beta-Binomial specification, ctd)

- The Likelihood

$$
\begin{aligned}
L(\pi_i, \gamma | y_i) &= P(y_i | \pi_i, \gamma) \\
lnL(\pi_i, \gamma | y_i) &= \sum_{i=1}^{n} \left\{ \sum_{j=0}^{y_i - 1} ln[\pi_i + \gamma j] + \sum_{j=0}^{N - y_i - 1} ln[\pi_i + \gamma j] - \sum_{j=0}^{N-1} ln[1 + \gamma j] \right\}
\end{aligned}
$$

- And then we substitute our systematic parameterization:

$$
lnL(\pi_i, \gamma | y_i) = \sum_{i=1}^{n} \left\{ \sum_{j=0}^{y_i - 1} ln[[1 + e^{-x_i \beta}]^{-1} + \gamma j] + \sum_{j=0}^{N - y_i - 1} ln[[1 + e^{-x_i \beta}]^{-1} + \gamma j] - \sum_{j=0}^{N-1} ln[1 + \gamma j] \right\}
$$

Binomial or Beta-Binomial?

- Which should we use?
- If you think your data are *not* independent, the Beta-Binomial is probably better.
- Tradeoffs
  - Because it has one fewer parameter to estimate, the binomial is (A) more restrictive but (B) more precise.
  - Estimates from the Binomial are consistent but inefficient if the model is wrong (i.e. biased standard errors).
  - Because it has another parameter to estimate, the Beta-Binomial is (A) less restrictive but (B) less precise.
- Alternative: estimate both, compare results, assess model fit

Gov 2001: Section 7

I. Ordered Categorial Variables
II. Binomial Variables
III. Diagnostics

Gov 2001

March 11, 2010

Diagnostics (Logit Example)

- Let's do some more with `Zelig`.
- Here, I work with some U.S. voter turnout data that come with the `Zelig` library.
- The outcome variable, `vote` is binary.

```
> summary(turnout)
     race           age          educate          income            vote
 others: 292   Min.   :17.0   Min.   : 0.00   Min.   : 0.000   Min.   :0.000
 white :1708   1st Qu.:31.0   1st Qu.:10.00   1st Qu.: 1.744   1st Qu.:0.000
               Median :42.0   Median :12.00   Median : 3.351   Median :1.000
               Mean   :45.3   Mean   :12.07   Mean   : 3.887   Mean   :0.746
               3rd Qu.:59.0   3rd Qu.:14.00   3rd Qu.: 5.233   3rd Qu.:1.000
               Max.   :95.0   Max.   :19.00   Max.   :14.925   Max.   :1.000
```

Diagnostics (ctd)

Because the outcome is binary, I'll fit a logit model:

```
> my.z

Call:  zelig(formula = vote ~ age + race + educate,
model = "logit", data = turnout)

Coefficients:
(Intercept)          age    racewhite       educate
   -3.04750      0.02750      0.37758      0.22320

Degrees of Freedom: 1999 Total (i.e. Null)
Null Deviance:     2267
Residual Deviance: 2072  AIC: 2080
```

Diagnostics (ctd)

Note: I can also do this using the `glm` function in `R`, which is pre-packaged:

```
> my.glm <- glm(vote ~ age + race + educate,
family = binomial(link = "logit"), data = turnout)
> my.glm

Coefficients:
(Intercept)           age    racewhite       educate
   -3.04750       0.02750       0.37758       0.22320

Degrees of Freedom: 1999 Total (i.e. Null);  1996 Residual
Null Deviance:      2267
Residual Deviance: 2072  AIC: 2080
```

Diagnostics (ctd)

- The first thing we could do is look at what the model tells us for a "known" value.
- Take me as an example – I vote regularly. Will the model predict that I vote?
- My age is 31, my race is "others," and my education level is 19.

```
> my.hypo <- setx(my.z, age = 31, educate = 19,
                  race = "others")
```

This sets the covariate values that correspond to me.

Diagnostics (ctd)

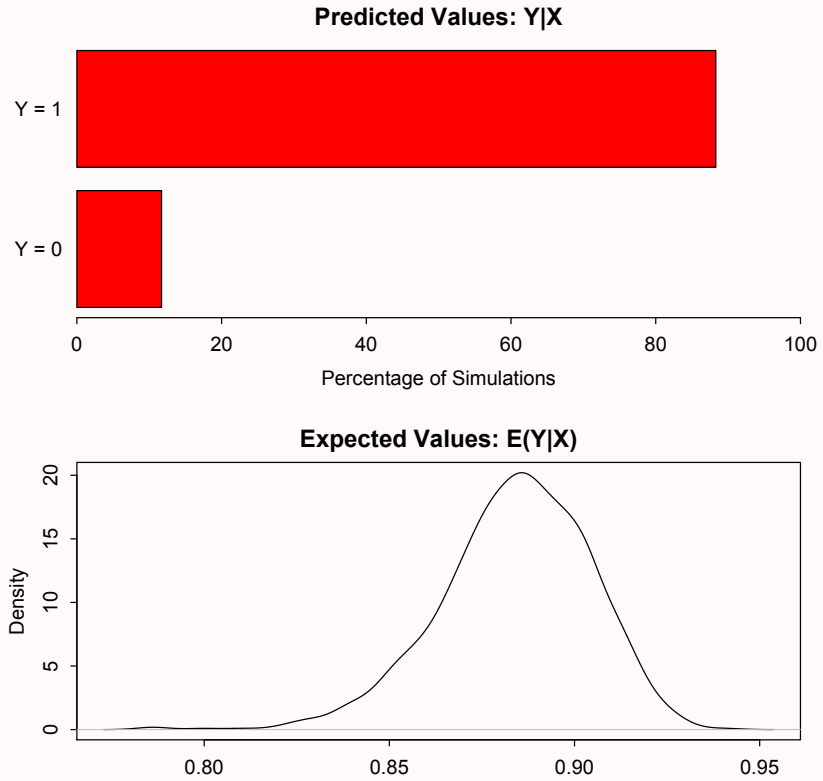
Now, let's simulate the quantities of interest (the probability of voting) from the posterior distribution

```
> my.sim <- sim(my.z, x = my.hypo)
> summary(my.sim)

  Model: logit
  Number of simulations: 1000

Values of X
  (Intercept) age racewhite educate
1           1  31         0      19

Expected Values: E(Y|X)
       mean          sd      2.5%     97.5%
1 0.8840032 0.02038413 0.8403375 0.9186828

Predicted Values: Y|X
      0     1
1 0.117 0.883
```

According to the model, I'm pretty likely to vote.

Diagnostics (ctd)

We can also look at this graphically using `plot(my.sim)`:



Diagnostics (ctd)

Another strategy:

- Let's set aside some ("training") data;
- then fit our model to this training data;
- make predictions;
- and compare our predictions to the rest of the ("test") data.

Diagnostics (ctd)

To take 100 observations out of the sample:

```
> random <- sample(length(turnout$educate), 100)
> samp <- turnout[random,]
> rest <- turnout[-random,]
```

Now, run the model on this sample:

```
> z.samp <- zelig(vote ~ race + educate + age,
                  model = "logit", data = samp)
```

Diagnostics (ctd)

- Now I want to compare the predictions from this training set to the rest of the data.
- How does the model do in predicting the behavior of 40-year-old high school-educated whites?

```
> x.hypo <- setx(my.z, age = 40, race = "white", educate = 12, data = turnout)
> s.hypo <- sim(my.z, x = x.hypo)
> summary(s.hypo)

Values of X
  (Intercept) age racewhite educate
1           1  40         1      12

Expected Values: E(Y|X)
       mean          sd      2.5%     97.5%
1 0.7519259 0.01139800 0.7307549 0.7735116

Predicted Values: Y|X
      0     1
1 0.242 0.758
```

Diagnostics (ctd)

- So the model predicts about 75% probability that a 40-year old HS-educated white will vote.
- How does this compare with what we see in the population (i.e., the rest of the data)?

```
> mean(rest$vote[rest$age == 40
& rest$race == "white" & rest$educate == 12])
[1] 0.7
```

- Not terrible.
- Note: The best test sets are out of sample. Unfortunately, we almost always don't have access to out-of-sample data.

Diagnostics (ctd)

- Suppose we want to compare two models

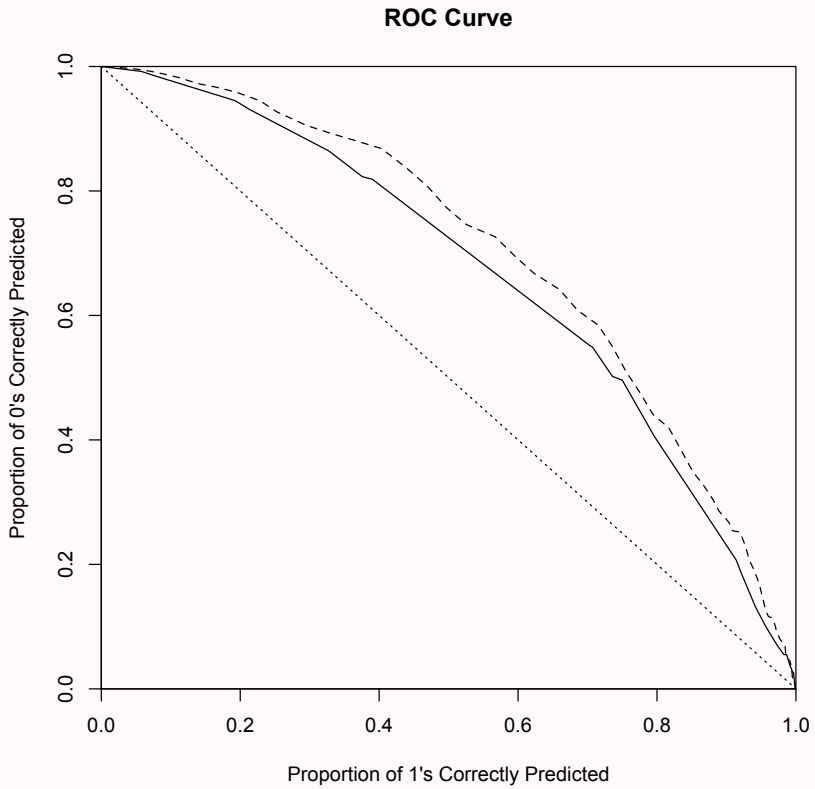
```
> z.out1 <- zelig(vote ~ race + educate,
                  model = "probit", data = turnout)
> z.out2 <- zelig(vote ~ race + educate + age,
                  model = "logit", data = turnout)
```

Diagnostics (ctd)

- We can create an receiver operating characteristic (ROC) curve, which is a graphical representation of sensitivity,
- In our case, it's the ratio of the proportion of 1s correctly predicted to the ratio of 0s correctly predicted
- Using `Zelig`:

```
> rocplot(z.out1$y, z.out2$y,
        fitted(z.out1), fitted(z.out2))
```

Diagnostics(ctd)



The model represented by the dotted line is a better model.