Discrete Outcome Regressions

Joseph Nathan Cohen

10/27/2019

Contents

Congratulations on Completing the Course's First Half

Congratulations on completing your mid-term. Over the first half of the semester you learned to use R. You now have a sense of its grammar, understand how to import packages, learn to read help files and get help online (e.g., StackOverflow). You have a basic sense of how to manage data, create visualizations, and perform basic statistical operations. The program is very powerful (with limits well beyond what any of us would do in applied social research), and it's free. Software price and capability should no longer be an issue for you in your career as an analyst. Insofar as programming is concerned, you can self-teach going forward, using books and listservs and such.

Second, you took your first steps past the point at which most quantitative analysis curricula end: a basic execution and interpretation of a linear model. That is generally where an undergraduate curriculum ends. The vast majority of people who take undergrad stats classes aren't going to be *making* models. They just need to know how to *use* the models. In learning regression diagnostics, we took an additional step: learning how to assess and improve the quality of a regression model's estimates. You move from a number interpreter to someone who designs the algorithm that crunches the numbers.

Moving forward, we're going to learn that there are different kinds of models, besides OLS, that can be used to identify relationships encoded in data. It is possible to tease information from data sets using a world of tools that are just one R package download away. Over the remainder of the course, we are going to focus on the specialty topics that commonly relevant to our graduates' work after graduation. Today's lesson looks at how to create models with discrete outcomes.

Modeling Discrete Outcomes: Example of Challenge and Redress

Recall from the last lesson (on regression diagnostics) that ordinary least squares ("linear") regression works if several assumptions are met. One of those assumptions conveyed to you is that the dependent variable be continuous. Allow me to illustrate using simulated data.

I create a group of 1000 people in groups "A" and "B", and group "A" will earn $30000 a year more INCOME, with estimates that tend to be off by a standard deviation of $5000.

Now let's say we want to use income to predict whether someone is part of Group A as opposed to B. If we run an OLS, diagnostics will reveal problems:

```r
MODEL.OLS <- lm(GROUP.A ~ INCOME, data = EX1DATA)
summary(MODEL.OLS)
```

```
##
## Call:
## lm(formula = GROUP.A ~ INCOME, data = EX1DATA)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.78420 -0.21136 -0.00952  0.19351  0.65951
##
## Coefficients:
##                Estimate   Std. Error t value Pr(>|t|)
## (Intercept) -0.992757659  0.102056466  -9.728 4.72e-16 ***
## INCOME       0.000022069  0.000001624  13.587  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2818 on 98 degrees of freedom
## Multiple R-squared:  0.6532, Adjusted R-squared:  0.6497
## F-statistic: 184.6 on 1 and 98 DF,  p-value: < 2.2e-16
```

If you were to interpret these results as you've been instructed, how would you mean it. How do we interpret the intercept term? A person with zero income is -1.22 of *what*? What does it mean if someone who earns $50,000 has a predicted value of 0.16, while one who earns $90,000 have one of 1.3? You can cut and paste this code into your R console and run diagnostics on it. The model fails handily.

GLM is a model that allows users to map their discrete data onto unbounded continuous scales to to which linear regression can be applied. With binary data, GLM converts the probability of a unit being in Group A *versus* Group B into a latent (invisible) unbounded scale where a more negative number suggests lower odds of being in Group A, and a higher one suggests higher odds. The distribution might look something like Figure 1 (below). It plots the relationship between a continuous x variable and a discrete y variable.

The logit model reconceptualizes these probabilities as logged odds. We map predictor values onto the following function. Let p be the probability that a subject will be in Group A:

$$odds = \frac{p}{(1-p)}$$

$$logit(p) = \log(odds) = \alpha + \beta \times x + \epsilon$$

"Odds" are the relative probability of a variable taking one score (e.g, Group A) versus another (Group B). Its log transformation maps very low odds into very negative numbers.

```r
#Let's run the logit model (more below)
MODEL.LOGIT <- glm(GROUP.A ~ INCOME, data = EX1DATA, family = "binomial")

#Predicted values are expected log odds
EX1DATA$PREDICTED <- predict(MODEL.LOGIT)

#Exponentiate to get odds
EX1DATA$ODDS <- exp(EX1DATA$PREDICTED)

#Reconstruct p:
EX1DATA$PROB <- exp(EX1DATA$PREDICTED) / (1 + exp(EX1DATA$PRE))
```

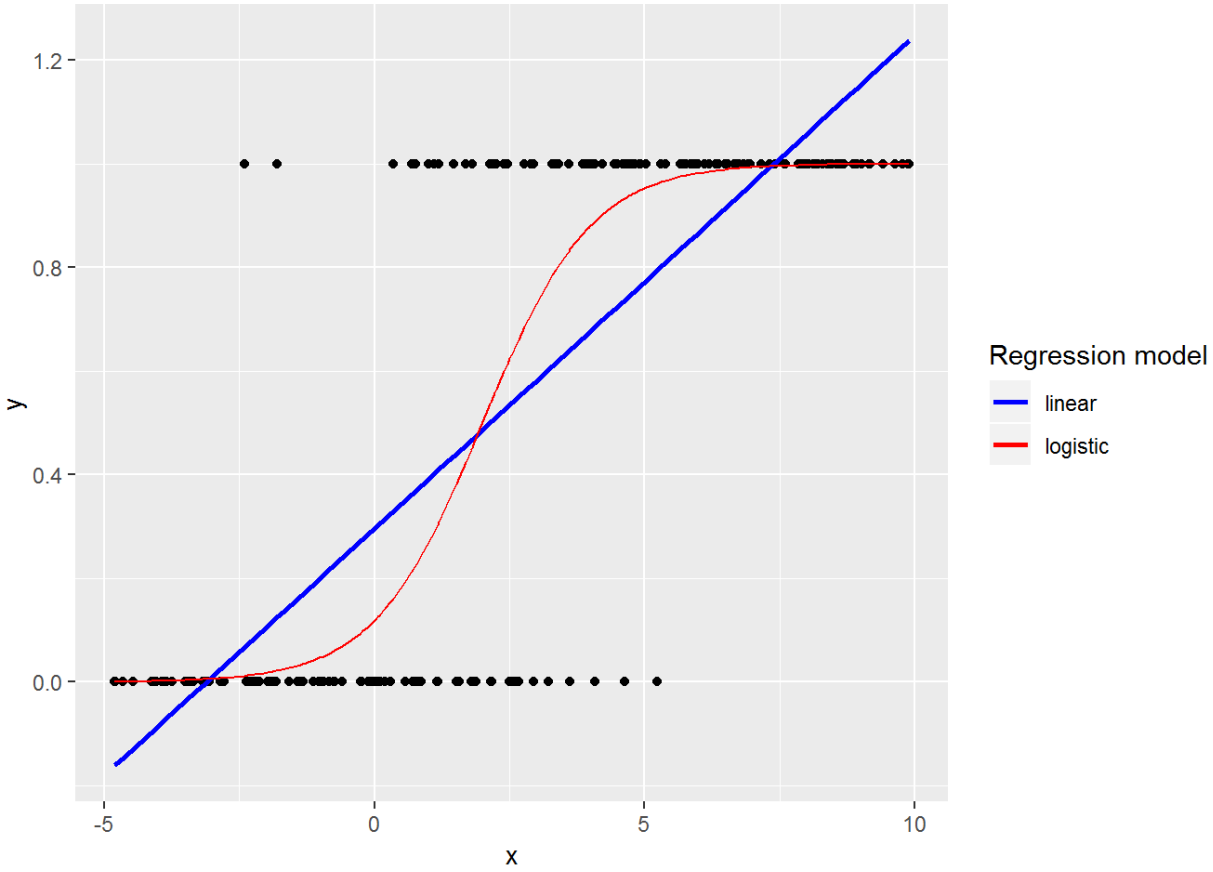

Figure 1: Logit Distribution

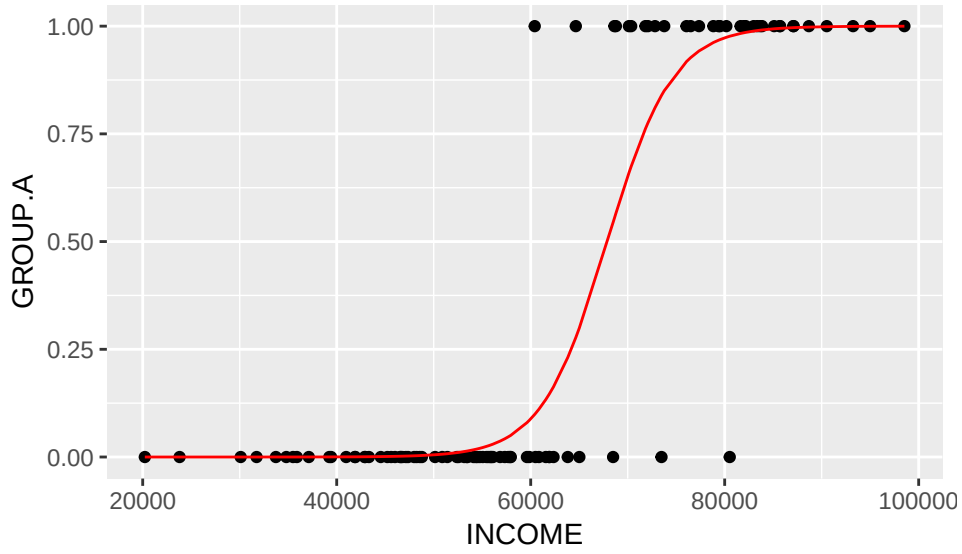

Figure 2: Adds in Example Model #2

```
library(ggplot2)
```

```
ggplot(EX1DATA, aes(x = INCOME)) + geom_point(aes(y = GROUP.A)) +
  geom_line(aes(y = PROB), color = 'red')
```

The model suggests the probability of being in Group A starts starts picking up at around $50,000 in income, and bec9omes a near certainty once people start earning more than $80,000. A few clever moves and we are now able to model binary variables. Let's take a closer look at how to execute and interpret these models

This is an example of a **logit** model, which we use to model binary variables. It is one of the models we will learn today. The others are the **ordered logit** for ordinal outcomes, the **multinomial logit** for multichotomous variables, and the **Poisson** regression for counts.

Preparing the Data

In this lesson, I'm going to show you the code I used to clean your data. Today, we are working with data from the Generalized Social Survey. However, today we are working with the raw variable and codebooks that you download from Berkeley SDA.

There's a new wrinkle here. Check out the lines where I use an **lapply()** command on a list of variables. You can use it to perform the same operation on multiple variables. There are other **apply()** functions that I'm sure will make their way into our lessons. For more, check out https://www.guru99.com/r-apply-sapply-tapply.html

```
dat <- read.csv("GSS Politics.csv")
dat <- dat[-c(8, 13, 33)] #Eliminating variables without data or variation

dat$AGE <- ifelse(dat$AGE > 97, NA, dat$AGE)
dat$SEX <- factor(dat$SEX, labels = c("Male", "Female"))
dat$RACE <- ifelse(dat$RACE == 0, NA, dat$RACE)
dat$RACE <- factor(dat$RACE, labels = c("White", "Black", "Other"))
temp <- c("DEGREE", "PADEG", "MADEG")
dat[temp] <- lapply(dat[temp], function(x) ifelse(x %in% c(7:9), 0, x))
dat[temp] <- lapply(dat[temp], function(x) factor(x,
                                                  labels = c("Below High School",
                                                             "High School",
                                                             "Junior College",
                                                             "Bachelors", "Graduate"),
                                                  ordered = T))
dat$WORDSUM <- ifelse(dat$WORDSUM %in% c(-1, 99), NA, dat$WORDSUM)
dat$WRKSTAT <- ifelse(dat$WRKSTAT %in% c(0,9), NA, dat$WRKSTAT)
dat$WRKSTAT <- factor(dat$WRKSTAT, labels = c("Working Full-Time", "Working Part-Time",
                                              "Temp Not Working","Unemployed", "Retired",
                                              "School", "Keeping House", "Other"))
dat$MARITAL <- ifelse(dat$MARITAL == 9, NA, dat$MARITAL)
dat$MARITAL <- factor(dat$MARITAL, labels = c("Married", "Widowed", "Divorced",
                                              "Separated", "Never Married"))
dat$CHILDS <- ifelse(dat$CHILDS == 9, NA, dat$CHILDS)
dat$AGEKDBRN <- ifelse(dat$AGEKDBRN > 97, NA, dat$AGEKDBRN)
dat$REGION <- ifelse(dat$REGION == 0, NA, dat$REGION)
dat$REGION <- factor(dat$REGION, labels = c("New England", "Mid Atlantic", "EN Central",
                                            "WN Central","South Atlantic", "ES Central",
                                            "WS Central", "Mountain", "Pacific"))
dat$PARTYID <- ifelse(dat$PARTYID %in% c(7:9), NA, dat$PARTYID)
dat$PARTYID <- factor(dat$PARTYID, labels = c("Strong Dem", "Mod Dem", "Leans Dem",
                                              "Indep.", "Leans Rep", "Mod Rep",
                                              "Strong"))
dat$POLVIEWS <- ifelse(dat$POLVIEWS %in% c(8,9), NA, dat$POLVIEWS)
temp <- names(dat)[15:29]
dat[temp] <- lapply(dat[temp], function(x) ifelse(x %in% c(0, 8, 9), NA, x))
dat[temp] <- lapply(dat[temp], function(x) factor(x,  labels = c("Too Little",
                                                                 "About Right",
                                                                 "Too Much")))
temp <- c("COLATH", "COLRAC", "COLMIL")
dat[temp] <- lapply(dat[temp], function(x) ifelse(x %in% c(0, 8, 9), NA, x))
dat[temp] <- lapply(dat[temp], function(x) factor(x, labels = c("Allowed",
                                                                "Not Allowed")))
temp <- names(dat)[34:46]
dat[temp] <- lapply(dat[temp], function(x) ifelse(x %in% c(0, 8, 9), NA, x))
```

```
dat[temp] <- lapply(dat[temp], function(x) factor(x, labels = c("A Great Deal",
                                                                "Only Some", "Hardly Any")))
rm(temp)
saveRDS(dat, file="GSS Politics.RDS")
```

Logit for Binary Outcomes

The logit model, presented above, used with binary outcomes. You can run it by using the **glm()** command in the base package, with the **family='binomial'** option. Below, we will create models that predict whether or not someone will report having "hardly any" faith in unions:

```
#Create Dependent Variable
#Equals 1 if "Hardly any" faith in unions, 0 otherwise
dat$NOFAITH.LABOR <- ifelse(dat$CONLABOR == "Hardly Any", 1, 0)

#Execute model
MODEL.LOGIT.2 <- glm(NOFAITH.LABOR ~ factor(PARTYID) + AGE, data = dat,
                     family = "binomial")
summary(MODEL.LOGIT.2)
```

```
##
## Call:
## glm(formula = NOFAITH.LABOR ~ factor(PARTYID) + AGE, family = "binomial",
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2202  -0.7596  -0.6078  -0.4394   2.2234
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -2.836923   0.247094 -11.481  < 2e-16 ***
## factor(PARTYID)Mod Dem     0.278684   0.217973   1.279    0.201
## factor(PARTYID)Leans Dem   0.094469   0.238101   0.397    0.692
## factor(PARTYID)Indep.      0.921461   0.209637   4.396 1.11e-05 ***
## factor(PARTYID)Leans Rep   0.945095   0.224890   4.202 2.64e-05 ***
## factor(PARTYID)Mod Rep     1.079035   0.214607   5.028 4.96e-07 ***
## factor(PARTYID)Strong      1.103248   0.227881   4.841 1.29e-06 ***
## AGE                        0.020604   0.003337   6.175 6.62e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1951.3  on 1795  degrees of freedom
## Residual deviance: 1848.4  on 1788  degrees of freedom
##   (1071 observations deleted due to missingness)
## AIC: 1864.4
##
## Number of Fisher Scoring iterations: 4
```

There is no R-Squared with this model, though it is possible to derive a pseudo-R-Squared.

```
library(DescTools)
PseudoR2(MODEL.LOGIT.2)
```

```
##  McFadden
## 0.0951092
```

```
#A pseudo R-Squared of 0.07
```

Remember: When we read coefficients, we look at the sign, size, and significance. It's the same here. Sign and significance are interpreted similar to OLS. Positive coefficients mean it raises the odds of a dependent variable being scored one as opposed to zero. Significance is similar too – if it isn't significant, we treat it as if there was no relationship. However, the size needs to be interpreted differently.

Interpreting Predictors

Discrete Predictors. Here, the baseline comaprison group are "Strong Democrats". Interpret the coefficient as suggesting that the demarcated group has $100 \times (\exp(\beta) - 1)$ percent change in the odds of being part of the group modeled in the logit. Let's interpret the "Independent" indicator in the model above. Here, the baseline comaprison group are "Strong Democrats":

$$\beta = 0.73354$$

$$100 \times [\exp(0.73345)] = 108$$

Thus, the model predicts that political indpenedents will have **108%** higher odds of not having confidence in unions, compared to the baseline comparison group (Strong Democrats)

Continuous Predictors. This coefficient suggests that adding +1 to a continuous indpendent variable results in a change in the odds of scoring one on our binary outcome variable. So AGE:

$$\beta = 0.02$$

$$100 \times [\exp(0.02) - 1] = 2$$

Thus, each additional year of age increases the odds by **2%** that a respondent will not trust unions.

Ordered Logit for Ordinal Outcomes

Reducing ordinal variables to binary ones results in information loss. It is possible to model an ordinal variable directly by using an ordered logit. We can do this with the **polr()** command in the ***MASS*** library:

```
library(MASS)
MODEL.OLOGIT <- polr(CONLABOR ~  factor(PARTYID) + AGE, data = dat)
summary(MODEL.OLOGIT)
```

```
##
## Re-fitting to get Hessian
```

```
## Call:
## polr(formula = CONLABOR ~ factor(PARTYID) + AGE, data = dat)
##
## Coefficients:
##                            Value Std. Error t value
## factor(PARTYID)Mod Dem   0.35241   0.165220   2.133
## factor(PARTYID)Leans Dem 0.35928   0.172157   2.087
## factor(PARTYID)Indep.    0.90112   0.170697   5.279
```

```
## factor(PARTYID)Leans Rep 0.86367   0.188392   4.584
## factor(PARTYID)Mod Rep   1.05975   0.180663   5.866
## factor(PARTYID)Strong    0.92539   0.198856   4.654
## AGE                      0.01881   0.002842   6.618
##
## Intercepts:
##                         Value   Std. Error t value
## A Great Deal|Only Some -0.4622  0.1882    -2.4555
## Only Some|Hardly Any    2.7414  0.2025    13.5363
##
## Residual Deviance: 3125.886
## AIC: 3143.886
## (1071 observations deleted due to missingness)
```

These results don't give stars, but you can infer them from the t-value column. A t-test value of greater than $t > 3.291$ is significant at $p<0.001$, $t > 2.576$ at $p < 0.01$, and $t > 1.960$ at $p < 0.05$. We interpret the coefficients the same way, but with a slight change. The coefficients express a percentage change in the odds of being in a higher category on the ordinal scale than one would otherwise expect, all other factors remaining constant.

Poisson Model for Counts

A Poisson model is used for counts. It is another GLM, but with a "family = poisson" option in the **glm()** operation. Let's try the CHILDS variable as an outcome, which is a count variable:

```
MODEL.POISSON <- glm(CHILDS ~ factor(PARTYID) + AGE, data = dat, family = poisson)
summary(MODEL.POISSON)
```

```
##
## Call:
## glm(formula = CHILDS ~ factor(PARTYID) + AGE, family = poisson,
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7425  -1.4215  -0.1538   0.6002   3.9766
##
## Coefficients:
##                            Estimate Std. Error z value     Pr(>|z|)
## (Intercept)              -0.3229566  0.0574773  -5.619 0.0000000192 ***
## factor(PARTYID)Mod Dem   -0.0194697  0.0466788  -0.417     0.676606
## factor(PARTYID)Leans Dem -0.1746845  0.0526505  -3.318     0.000907 ***
## factor(PARTYID)Indep.     0.1045591  0.0471372   2.218     0.026542 *
## factor(PARTYID)Leans Rep -0.0433128  0.0547485  -0.791     0.428872
## factor(PARTYID)Mod Rep   -0.0013449  0.0494908  -0.027     0.978321
## factor(PARTYID)Strong    -0.1100670  0.0548637  -2.006     0.044836 *
## AGE                       0.0185114  0.0008066  22.951      < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 4744.8  on 2748  degrees of freedom
## Residual deviance: 4167.8  on 2741  degrees of freedom
##   (118 observations deleted due to missingness)
```

```
## AIC: 9589.7
##
## Number of Fisher Scoring iterations: 5
```

Again, we read the sign, size, and significance of the coefficients. Again, the coefficient size requires a similar interpretation.

Continuous Predictors. AGE has a coefficient of 0.018. We calculate 100 * [exp(0.018) - 1] = 1.8%. Each additional year of age increases your expected number of children by 1.8%.

Discrete Predictors. The "Strong Republican" coefficient is -0.11. We calculate 100 * [exp(-0.11) - 1] = -1.09. A strong Republican is expected to have -1.1% fewer children than a Strong Democrat.