lecture 05

{marginaleffects}

{marginaleffects}; information criteria; zero inflation

review

example exam questions

Mini Exam

Paper and Workshop

For the prospectus: “as close to a complete paper as you can make it.”

Must haves:

- A clear statement of the descriptive claim (or perhaps question). Put this in bold if you want
- A clear argument that the pattern is important, even if the relationship isn’t causal.
- A demonstration that any data collection is feasible. Complete is idea.
- A initial data analysis. Averages and/or scatterplots are fine for now, perhaps even preferable.

{marginaleffects}

We understand how things work.

elegant, powerful theoretical framework

probability model

ML estimates

Fisher information

invariance property

delta method

quantities of interest

We understand how things work.

computation

```
# ---- fit logit model with optim() ----

# data
devtools::install_github("jrnold/ZeligData")
turnout <- ZeligData::turnout

# formula
f <- vote ~ age + educate + income + race

# ---- create a function to fit the model ----
# log-likelihood function
logit_ll <- function(beta, y, X) {
  linpred <- X%*%beta  # perhaps denoted eta
  p <- plogis(linpred) # pi is special in R, so use p
  ll <- sum(dbinom(y, size = 1, prob = p, log = TRUE))
  return(ll)
}

# function to fit model
est_logit <- function(f, data) {

  # make X and y
  mf <- model.frame(f, data = data)
  X <- model.matrix(f, data = mf)
  y <- model.response(mf)

  # create starting values
  par_start <- rep(0, ncol(X))

  # run optim()
  est <- optim(par_start,
               fn = logit_ll,
               y = y,
               X = X,
               hessian = TRUE, # for SEs!
               control = list(fnscale = -1),
               method = "BFGS")

  # check convergence; print warning if not
  if (est$convergence != 0) print("Model did not converge!")

  # create list of objects to return
  res <- list(beta_hat = est$par,
              var_hat = solve(-est$hessian))

  # return the list
  return(res)
}
# fit model
fit <- est_logit(f, data = turnout)
print(fit, digits = 2)
```

model

ML

Fisher info

```
# ---- compute first difference ----

# make X_lo
X_lo <- cbind(
  "constant" = 1, # intercept
  "age"      = quantile(turnout$age, probs = 0.25), # 31 years old
  "educate"  = median(turnout$educate),
  "income"   = median(turnout$income),
  "white"    = 1 # white indicators = 1
)

# make X_hi by modifying the relevant value of X_lo
X_hi <- X_lo
X_hi[, "age"] <- quantile(turnout$age, probs = 0.75) # 59 years old

# function to compute first difference
fd_fn <- function(beta, hi, lo) {
  plogis(hi%*%beta) - plogis(lo%*%beta)
}

# invariance property
fd_hat <- fd_fn(fit$beta_hat, X_hi, X_lo)
```

invariance property

```
# delta method
grad <- grad(
  func = fd_fn,
  x = fit$beta_hat,
  hi = X_hi,
  lo = X_lo)
se_fd_hat <- sqrt(grad %*% fit$var_hat %*% grad)
```

delta method

```
# estimated fd
fd_hat

# estimated se
se_fd_hat

# 90% ci
fd_hat - 1.64*se_fd_hat  # lower
fd_hat + 1.64*se_fd_hat  # upper
```

quantities of interest

But how can we do it

easily?

But how can we do it

robustly?

When you write lots of code to do common tasks, you run a **major risk of mistakes**.

Where possible, use **old**, **well-tested**, **widely used** code **written by developers**.

Rather than this...

```r
# ---- fit logit model with optim() ----

# data
devtools::install_github("jrnold/ZeligData")
turnout <- ZeligData::turnout

# formula
f <- vote ~ age + educate + income + race

# ---- create a function to fit the model ----

# log-likelihood function
logit_ll <- function(beta, y, X) {
  linpred <- X%*%beta  # perhaps denoted eta
  p <- plogis(linpred) # pi is special in R, so I use p
  ll <- sum(dbinom(y, size = 1, prob = p, log = TRUE))
  return(ll)
}

# function to fit model
est_logit <- function(f, data) {

  # make X and y
  mf <- model.frame(f, data = data)
  X <- model.matrix(f, data = mf)
  y <- model.response(mf)

  # create starting values
  par_start <- rep(0, ncol(X))

  # run optim()
  est <- optim(par_start,
               fn = logit_ll,
               y = y,
               X = X,
               hessian = TRUE, # for SEs!
               control = list(fnscale = -1),
               method = "BFGS")

  # check convergence; print warning if not
  if (est$convergence != 0) print("Model did not converge!")

  # create list of objects to return
  res <- list(beta_hat = est$par,
              var_hat = solve(-est$hessian))

  # return the list
  return(res)
}

# fit model
fit <- est_logit(f, data = turnout)
print(fit, digits = 2)
```

```r
# ---- compute first difference ----

# make X_lo
X_lo <- cbind(
  "constant" = 1, # intercept
  "age"      = quantile(turnout$age, probs = 0.25), # 31 years old
  "educate"  = median(turnout$educate),
  "income"   = median(turnout$income),
  "white"    = 1 # white indicators = 1
)

# make X_hi by modifying the relevant value of X_lo
X_hi <- X_lo
X_hi[, "age"] <- quantile(turnout$age, probs = 0.75) # 59 years old

# function to compute first difference
fd_fn <- function(beta, hi, lo) {
  plogis(hi%*%beta) - plogis(lo%*%beta)
}

# invariance property
fd_hat <- fd_fn(fit$beta_hat, X_hi, X_lo)

# delta method
grad <- grad(
  func = fd_fn,
  x = fit$beta_hat,
  hi = X_hi,
  lo = X_lo)
se_fd_hat <- sqrt(grad %*% fit$var_hat %*% grad)

# estimated fd
fd_hat

# estimated se
se_fd_hat

# 90% ci
fd_hat - 1.64*se_fd_hat  # lower
fd_hat + 1.64*se_fd_hat  # upper
```

….do this.

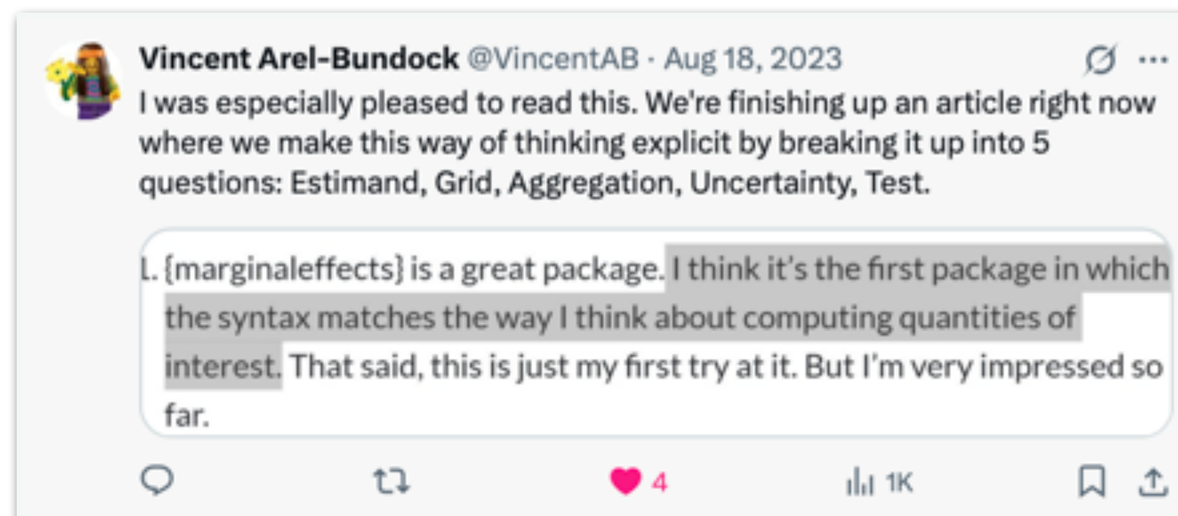
```r
# data
devtools::install_github("jrnold/ZeligData")
turnout <- ZeligData::turnout

# fit model
f <- vote ~ age + educate + income + race
fit <- glm(f, family = binomial, data = turnout)

# compute qi
comparisons(fit,
            variables = list(age = "iqr"),
            newdata = datagrid(FUN_numeric = median))
```

use {marginaleffects}

relatively new ⚠️

well documented ✅

widely used ✅



warning

“Easy-to-use” software is less likely to have bugs.

But it’s *more* likely to be used incorrectly—
you might not understand what it’s doing.

It’s **critical** to read documentation carefully
and test your understanding.

read documentation carefully

A Conceptual Framework for {marginaleffects}

1. **Quantity**: What is the quantity of interest?
   - Do we want an expected value?
   - Or do we want a *comparison* of expected values (difference, ratio, derivative, etc.)?
2. **Grid**: What predictor values are we interested in?
   - Do we want estimates for the observations in our dataset?
   - Or do we want estimates for hypothetical observations?
3. **Aggregation**: How do we aggregate across the grid, if at all?
   - Do we want estimates for every observation in the grid?
   - Or do we want a summary of the estimates?

Quantity

expected value	first difference, *etc.*
$E(y \mid X_c)$	$E(y \mid X_{hi}) - E(y \mid X_{lo})$
↓	↓
`predictions()`	`comparisons()`

Grid

By default, {marginaleffects} tries to use the observed values.

```
predictions(fit, newdata = x, ...)
```

```
comparisons(fit, newdata = x, ...)
```

`datagrid()` is very powerful.

Aggregation

`predictions()` → `avg_predictions()`

`comparisons()` → `avg_comparisons()`

Examples

```r
# data
devtools::install_github("jrnold/ZeligData")
turnout <- ZeligData::turnout

# fit model
f <- vote ~ age + educate + income + race
fit <- glm(f, family = binomial, data = turnout)
```

```r
# ev as age ranges from 18 to 90; others at mean/mode
predictions(fit, newdata = datagrid(age = 18:90))

# fd as age moves across iqr; others at every observed value
comparisons(fit, variables = list(age = "iqr"))

# avg of the fds above
avg_comparisons(fit, variables = list(age = "iqr"))
```

things to tinker with…

- `predictions()` or `comparisons()`
  (i.e., quantity, part 1)

- `newdata` argument
  (i.e., the grid)

- `comparison` argument (`comparisons()` only)
  (i.e., quantity, part 2)

- `avg_*()` variants
  (i.e., aggregation)

some notes

`predictions()` and `comparisons()`

- always return a data frame
  - use your data wrangling skills on the output
  - use your ggplot2 skills on the output
- always return the grid, so check that you did what you think you did.

Example

https://gist.github.com/carlislerainey/507332fe1f30ea097f3513ad2d195404

information criteria

$$-2\ell(\hat{\theta}) + \underbrace{\boxed{[\text{constant} \times k]}}_{\text{complexity penalty}}$$

Full Name	Short	constant
Akaike information criterion	AIC	2
Bayesian information criterion	BIC	log(*n*)

```r
# data
devtools::install_github("jrnold/ZeligData")
turnout <- ZeligData::turnout

# fit model
f <- vote ~ age + educate + income + race
fit <- glm(f, family = binomial, data = turnout)

# fit model
f <- vote ~ poly(age, 3) + educate + income + race
fit3 <- glm(f, family = binomial, data = turnout)

# create table
modelsummary(list("Linear" = fit, "Cubic" = fit3),
             shape = term ~ model + statistic)
```

	Linear		Cubic	
	Est.	S.E.	Est.	S.E.
(Intercept)	-3.034	0.326	-1.706	0.240
age	0.028	0.003		
educate	0.176	0.020	0.179	0.020
income	0.177	0.027	0.154	0.028
racewhite	0.251	0.146	0.268	0.147
poly(age, 3)1			21.665	2.685
poly(age, 3)2			-9.406	2.442
poly(age, 3)3			-2.570	2.387
Num.Obs.	2000		2000	
AIC	2034.0		2022.4	
BIC	2062.0		2061.6	
Log.Lik.	-1011.991		-1004.210	
RMSE	0.41		0.41	

TABLE 6
Grades of Evidence Corresponding to Values of the Bayes Factor for M_2 Against M_1, the BIC Difference and the Posterior Probability of M_2

BIC Difference	Bayes Factor	$p(M_2 \mid D)(\%)$	Evidence
0–2	1–3	50–75	Weak
2–6	3–20	75–95	Positive
6–10	20–150	95–99	Strong
>10	>150	>99	Very strong

of them. Often all the models will be on an equal footing *a priori*, so that $p(M_1) = \ldots = p(M_K) = 1/K$. By the results in Section 4.1, approximately, $p(D|M_k) \propto \exp(-\frac{1}{2}\mathrm{BIC}_k)$ or $\exp(-\frac{1}{2}\mathrm{BIC}'_k)$. Thus

$$p(M_k|D) \approx \exp(-\tfrac{1}{2}\mathrm{BIC}_k) / \sum_{l=1}^{K} \exp(-\tfrac{1}{2}\mathrm{BIC}_\ell). \tag{35}$$

Equation (35) still holds if BIC is replaced by BIC′.

Example

https://gist.github.com/carlislerainey/12b3d0e97b918099573ae2c42cc312eb

models for counts

The Distributive Politics of Enforcement

Alisha C. Holland Harvard University

Why do some politicians tolerate the violation of the law? In contexts where the poor are the primary violators of property laws, I argue that the answer lies in the electoral costs of enforcement: Enforcement can decrease support from poor voters even while it generates support among nonpoor voters. Using an original data set on unlicensed street vending and enforcement operations at the subcity district level in three Latin American capital cities, I show that the combination of voter demographics and electoral rules explains enforcement. Supported by qualitative interviews, these findings suggest how the intentional nonenforcement of law, or forbearance, can be an electoral strategy. Dominant theories based on state capacity poorly explain the results.

In much of the developing world, a source of resources for the poor is the ability to violate property laws without state sanction. Squatters gain rent-free housing if their takings succeed. Street vendors secure a way to earn a living when the government ignores their unlicensed stands. The idea that enforcement has distributive consequences is not new. Yet conventional wisdom is that limited enforcement reflects a weak state unable to implement its laws due to budget constraints or principal-agent problems.

In contrast, this article argues that nonenforcement of law is often intentional—what I call *forbearance*—and explains why some governments tolerate violations of the law by the poor and others do not. The argument is sim-

An intuitive distributive logic thus provides greater leverage to understand enforcement (and its absence) than dominant capacity-based approaches.

Focusing on variation in enforcement against unlicensed street vendors at the city and subcity level, this article tests this electoral theory in two ways. I first examine time-series data on enforcement in a city that constitutes a single electoral district, Bogotá, Colombia. I show that city mayors with nonpoor core constituencies conduct almost five times more enforcement operations against street vendors than those with poor constituencies. Second, I collect original data on enforcement operations and unlicensed street vending in a sample of 89 subcity units, or districts, in three cities. I select cities that vary in tics. The Poisson distribution is appropriate given that enforcement is a count variable with a range restricted to positive integers.[13]

My first hypothesis is that enforcement operations drop off with the fraction of poor residents in an electoral district. So district poverty should be a negative and significant predictor of enforcement, but only in politically decentralized cities. Poverty should have no relationship with enforcement in politically centralized cities once one controls for the number of vendors.

I include the number of vendors as a covariate for the limited purpose of observing the difference depending on whether enforcement policy is locally or centrally

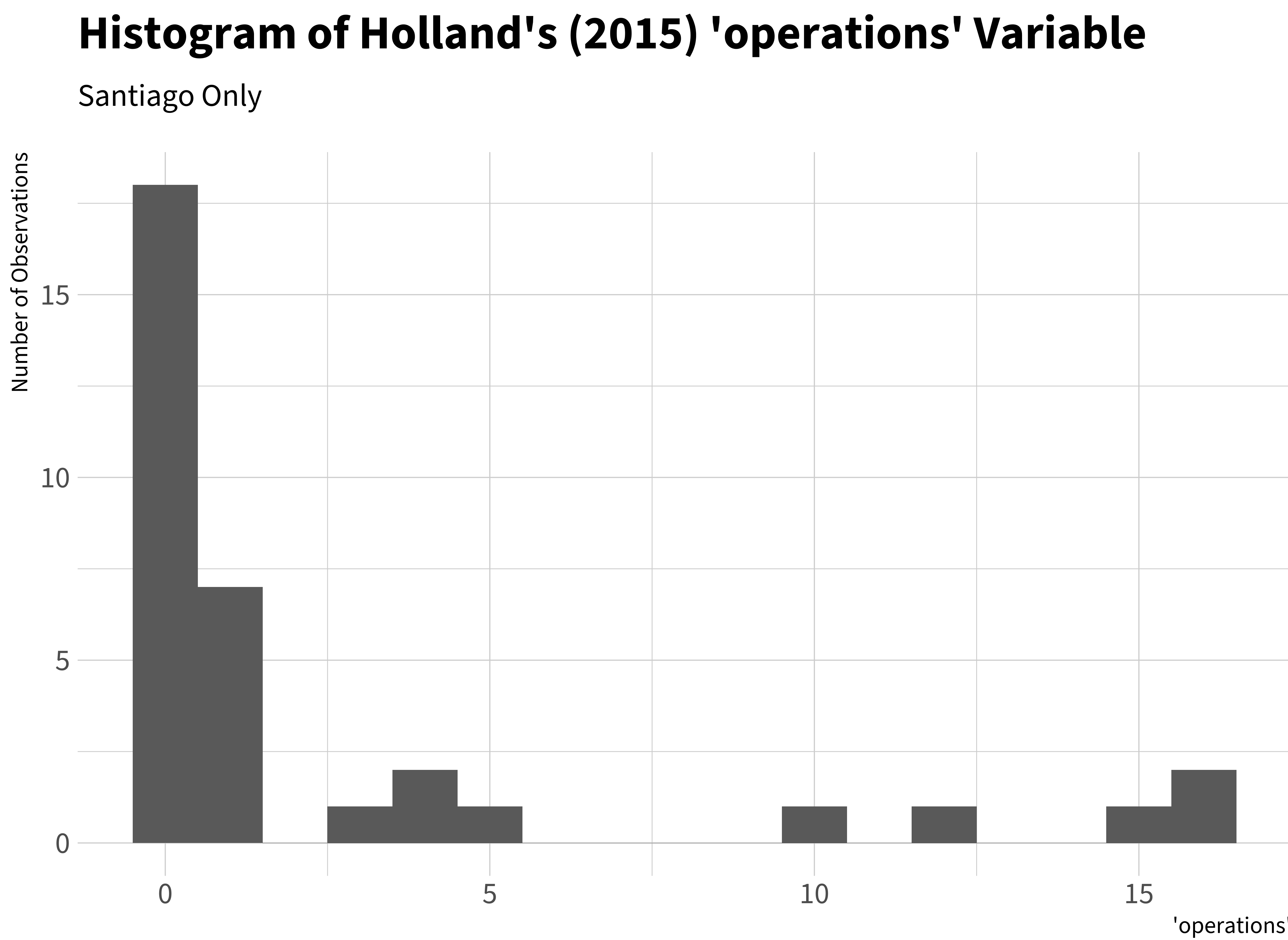
Table 1 Theoretical Hypotheses and Empirical Predictions

Hypothesis	Empirical Prediction
Hypothesis 1: Enforcement decreases with the poverty of an electoral district.	$\beta_{lower} < 0$, $\beta_{vendors\mid lower} \approx 0$ in Lima and Santiago $\beta_{vendors} > 0$ in Bogotá
Hypothesis 2: District demographics are less relevant under limited political competition.	$\beta_{lower\mid vendors} \approx 0$ in Bogotá $\beta_{margin*lower} > 0$ in Lima and Santiago
Hypothesis 3: Politicians enforce less when their core constituents are poor.	$\bar{E}_{nonpoor} > \bar{E}_{poor}$ in Bogotá $\beta_{right} > 0$ in Santiago
Alternative 1: Enforcement decreases with the poverty of a district due to capacity constraints.	$\beta_{lower} < 0$ in all cities $R^2_{budget} > R^2_{lower}$
Alternative 2: Enforcement decreases with the poverty of a district because the police are less responsive.	$\beta_{lower} < 0$ in all cities $\beta^{arrests}_{lower} < 0$ in Santiago
Alternative 3: Politicians enforce in proportion to the number of offenses.	$\beta_{vendor} > 0$ in all cities

Notes: $\bar{E}$ refers to the expected number of enforcement operations.

```
> # data; see ?crdata::holland2015
> holland <- crdata::holland2015 |>
+   filter(city == "santiago")
> glimpse(holland)
Rows: 34
Columns: 7
$ city       <chr> "santiago", "santiago", "santiago", "santiago", "santiago", "santiago", "santi…
$ district   <chr> "Cerrillos", "Cerro Navia", "Conchali", "El Bosque", "Estacion Central", "Huec…
$ operations <dbl> 0, 0, 0, 0, 12, 0, 0, 0, 1, 1, 0, 10, 1, 5, 0, 0, 0, 4, 4, 0, 1, 16, 1, 1, 0, …
$ lower      <dbl> 52.2, 69.8, 54.8, 58.4, 43.6, 58.3, 41.0, 38.3, 36.7, 60.1, 73.8, 16.4, 7.7, 2…
$ vendors    <dbl> 0.50, 0.60, 5.00, 1.20, 1.00, 0.30, 0.05, 1.25, 2.21, 0.70, 1.00, 0.50, 0.05, …
$ budget     <dbl> 337.24, 188.87, 210.71, 153.76, 264.43, 430.42, 312.75, 255.53, 149.48, 164.98…
$ population <dbl> 6.6160, 13.3943, 10.7246, 16.8302, 11.1702, 8.5761, 5.1277, 7.1443, 39.8355, 1…
```

Note: Santiago is “highly decentralized.”


Histogram of Holland's (2015) 'operations' Variable
Santiago Only
Number of Observations
0
5
10
15
0
5
10
15
'operations'

Poisson Regression

Outcome. Counts $0, 1, 2, \ldots$

Model.

$$y_i \sim \text{Pois}(\mu_i), \qquad \mu_i = \exp(X_i \beta).$$

Expected value (choosing X_c).

$$\mathbb{E}[y]_c = \mu_c,$$

$$\mu_c = \exp(X_c \hat{\beta}).$$

First difference (choosing X_{hi} and X_{lo}).

$$\Delta = \mathbb{E}[y]_{\text{hi}} - \mathbb{E}[y]_{\text{lo}} = \mu_{\text{hi}} - \mu_{\text{lo}},$$

$$\mu_{\bullet} = \exp(X_{\bullet} \hat{\beta}).$$

Fit in R.

```
glm(y ~ x1 + x2, family = poisson())
```

Notes.

- Assumes mean and variance are equal.

Negative Binomial Regression

Outcome. Counts $0, 1, 2, \ldots$

Model.

$$y_i \sim \mathrm{NB}(\mu_i, \theta), \qquad \mu_i = \exp(X_i \beta).$$

Expected value (choosing X_c).

$$\mathbb{E}[y]_c = \mu_c,$$

$$\mu_c = \exp(X_c \hat{\beta}).$$

First difference (choosing X_{hi} and X_{lo}).

$$\Delta = \mathbb{E}[y]_{\mathrm{hi}} - \mathbb{E}[y]_{\mathrm{lo}} = \mu_{\mathrm{hi}} - \mu_{\mathrm{lo}},$$

$$\mu_{\bullet} = \exp(X_{\bullet} \hat{\beta}).$$

Fit in R.

```
library(MASS) # watch conflicts w/ tidyverse

glm.nb(y ~ x1 + x2)
```

Notes.

- θ (aka `size`) controls overdispersion in NB2.
- $\mathrm{Var}(y_i) = \mu_i + \mu_i^2/\theta.$

Zero-Inflated Negative Binomial Regression

Outcome. Counts $0, 1, 2, \ldots$

Model.

$$y_i \sim \begin{cases} 0 & \text{w.p. } \pi_i, \\ \mathrm{NB}(\mu_i, \theta) & \text{w.p. } 1 - \pi_i \end{cases} \qquad \mu_i = \exp(X_i \beta), \qquad \pi_i = \mathrm{logit}^{-1}(Z_i \gamma).$$

Expected value (choosing X_c).

$$\mathbb{E}[y]_c = (1 - \pi_c)\, \mu_c,$$

$$\mu_c = \exp(X_c \hat{\beta}), \qquad \pi_c = \mathrm{logit}^{-1}(Z_c \hat{\gamma}).$$

First difference (choosing X_{hi} and X_{lo}).

$$\Delta = \mathbb{E}[y]_{\mathrm{hi}} - \mathbb{E}[y]_{\mathrm{lo}} = (1 - \pi_{\mathrm{hi}})\mu_{\mathrm{hi}} - (1 - \pi_{\mathrm{lo}})\mu_{\mathrm{lo}},$$

$$\mu_{\bullet} = \exp(X_{\bullet} \hat{\beta}), \qquad \pi_{\bullet} = \mathrm{logit}^{-1}(Z_{\bullet} \hat{\gamma}).$$

Fit in R.

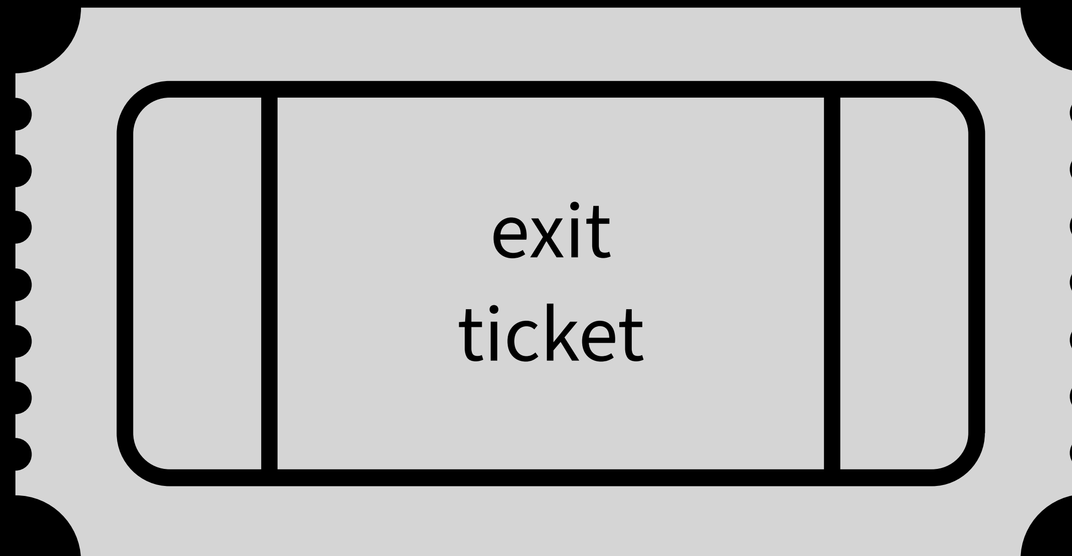
```
library(glmmTMB)

glmmTMB(y ~ x1 + x2, ziformula = ~ x1 + x3, family = nbinom2)
```

Notes.

- θ (aka `size`) controls overdispersion in the NB2 component.
- Zero inflation is on the logit scale via its own design Z_i, which can include intercept only (i.e., constant), the same covariates as X, or different covariates.

Example

https://gist.github.com/carlislerainey/85180ee05c6f4566b2262f0dcc1f9117



List three important ideas from today’s class. For each, briefly connect it to one or more ideas from last week.