第21届国际关系研究方法研讨班

Logit & Probit 回归模型

陈冲
清华大学国际关系学系
cc458.github.io
2023年6月28日

Agenda

- Linear probability model
- Logit model
- Probit model
- Model Assumptions

Binary Response Variables

- Binary response variables: Variables with only two possible outcomes
  - Country at risk of political violence (Yes/No)
  - Decision to join or not join BRI
- Goals:
  - Estimate probabilities of category membership
  - Relate probabilities of category membership to other variables
    - **Model**: variation in the probability a country is at risk of political violence given values of the predictor variables (`GDP`, `population`, `regime type`, etc.)

Data and Packages

```r
# Load the packages
library(tidyverse)
library(broom)
library(AER) #for robust standard errors
library(dotwhisker)
library(stargazer)
library(knitr)
library(cowplot)
load(url("https://cc458.github.io/files/IRdata.RData"))
# we will use the 2015 data for now
df <- IRdata %>% filter(year == 2015)
```

- `vio`: Y is a binary response variable
  - 1: yes (violence occurrence)
  - 0: no (violence absence)
- `gdppc`, `pop`, `polity2`: X a set of predictors

Recap: Bernoulli distribution

If Y is a random variable with two possible outcomes:

$$Pr(Y = 1) = p = 1 - Pr(Y = 0) = 1 - q$$

The probability mass function f of this distribution, over possible outcomes k

$$f(k, p) = \begin{cases} p, \ if \ k = 1 \\ q = 1 - p, \ if \ k = 0 \end{cases}$$

This can also be expressed as

$$f(k, p) = p^k (1 - p)^{1-k}, \ for \ k \in \{0, 1\}$$

$$0 <= p <= 1$$

Modeling Binary DVs

Three common models for binary response variables:

$$y_i \sim Bernoulli(p_i), y \in \{0, 1\}$$

where,

$$p_i = f(x_i\beta)$$

- Linear Probability Model via OLS
  - Identity link function for p
- Non-linear probability model via MLE
  - Logistic Regression: Logit link function for p
  - Probit Regression: Probit link function for p

Linear Probability Model

Linear Probability Model

$$y_i \sim Bernoulli(p_i), y \in \{0, 1\}$$

$$p_i = x_i \beta$$

- This seems like a strange choice
- It is nonetheless useful to explore its meaning and properties

Interpretation

The LPM has an identity link for p, so the coefficients can be interpreted directly in terms of probabilities.

$$p(y_i = 1) = x_i\beta$$

$$p(y_i = 0) = 1 - x_i\beta$$

Estimating LPM in R

$$violence = \beta_0 + \beta_1 \times \text{GDP per capita} + \epsilon$$

```
model <- lm(vio ~ gdppc, data = df)
tidy(model) %>% kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	0.938	0.200	4.681	0.000
gdppc	-0.082	0.023	-3.518	0.001

Robust standard errors

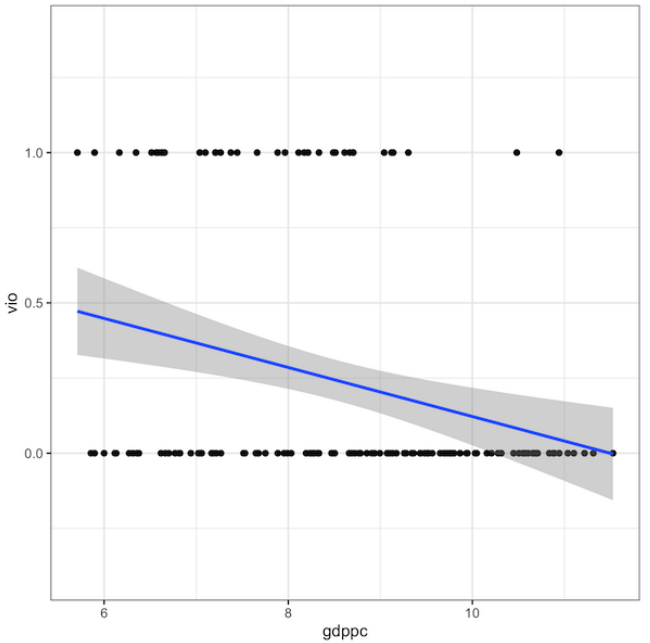
- OLS standard errors are biased in the presence of **heteroscedasticity**, hypothesis tests are wrong
- It is essential to use robust standard errors since the ϵ_i in a linear probability model are always heteroskedastic (non-constant variance for ϵ_i), e.g., σ^2 is proportional to the value of X_i).

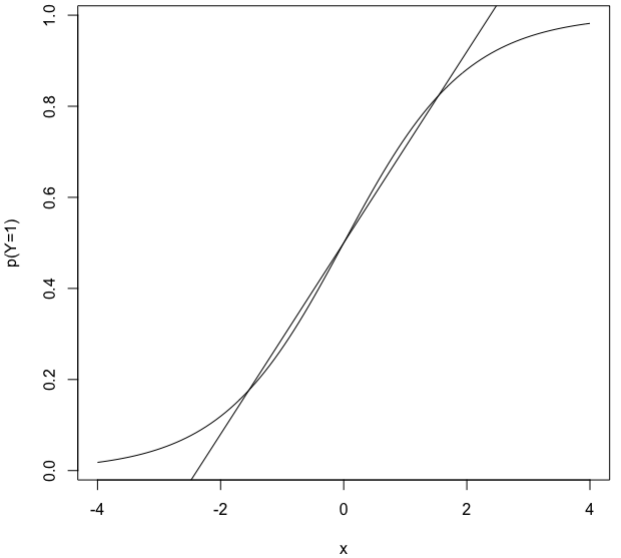
```
# print robust coefficient summary
coeftest(model, vcov. = vcovHC, type = "HC1") %>%
  tidy() %>%
  kable(format = "markdown", digits = 5)
```

term	estimate	std.error	statistic	p.value
(Intercept)	0.93791	0.20333	4.61276	0.00001
gdppc	-0.08159	0.02197	-3.71437	0.00029

Problems with LPM

- The LPM imposes linearity on what is necessarily a non-linear data generating process
  - Probabilities are bounded by 0 and 1, but the LPM places no such restrictions on the model
  - It also overestimates changes in p as a function of X near 1 and 0





Non-Linear Link Functions

Allowing for Non-Linearity

y_i are independent, identically distributed (i.i.d.) Bernoulli trials:

$$y_i \sim Bernoulli(p_i), y \in \{0, 1\}$$

$$p_i = g(x_i\beta)$$

Suppose that $Y = (y_1, y_2, \cdots, y_n)$ represents the outcomes of n independent Bernoulli trials, each with success probability p.

Allowing for Non-Linearity

The likelihood for p based on X is defined as the joint probability distribution of $Y_1, Y_2, \cdots, Y_n$. Since $Y_1, Y_2, \cdots, Y_n$ are i.i.d. random variables, the joint distribution is given as:

$$L(P|Y) = \prod g(x_i\beta)^{y_i}(1 - g(x_i\beta))^{(1-y_i)}$$

The log likelihood is (take log on both sides) :

$$LL(P|Y) = \sum y_i ln(g(x_i\beta)) + (1 - y_i) ln(1 - g(x_i\beta))$$

Logit and Probit Link Functions: g( )

$$y_i \sim Bernoulli(p_i), y \in \{0, 1\}$$

$$p_i = g(x_i\beta)$$

- Logit regression:

$$p_i = logistic(x_i\beta),\ logit(p_i) = x_i\beta$$

- Probit regression:

$$p_i = \Phi(x_i\beta),\ probit(p_i) = x_i\beta$$

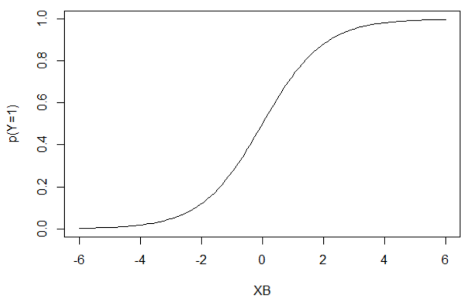
Logistic and Logit Functions

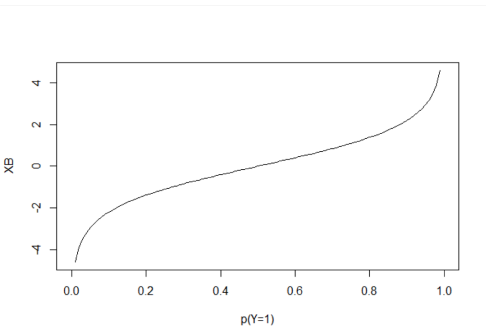
- Logistic function = standard cumulative logistic distribution

$$\frac{exp(x)}{1 + exp(x)}$$

- Logit function = log odds = inverse cumulative logistic:

$$log(\frac{x}{1 - x})$$





Logistic Regression Model

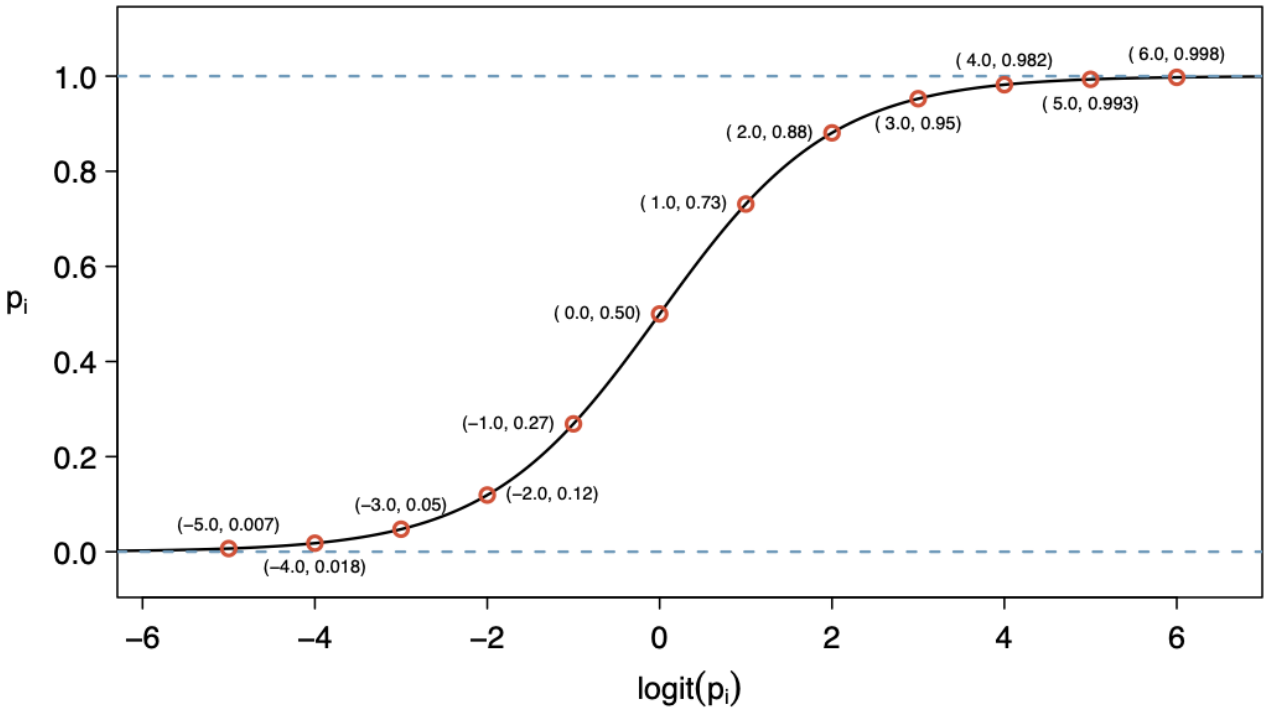
- Suppose $P(y_i = 1|x_i) = p_i$ and $P(y_i = 0|x_i) = 1 - p_i$
- The logistic regression model is

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_i$$

- $\log\left(\frac{p_i}{1-p_i}\right)$ is called the logit function

Logit function

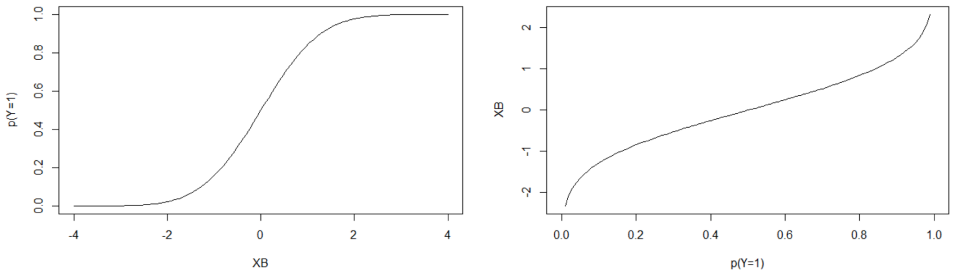
$$0 \leq p \leq 1 \quad \Rightarrow \quad -\infty < \log\left(\frac{p}{1-p}\right) < \infty$$



OpenIntro Statistics, 4th ed (pg. 373)

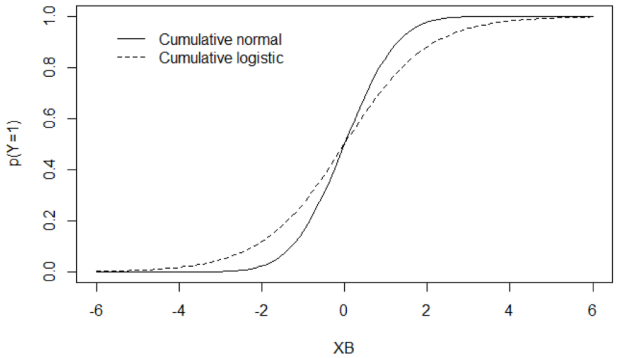
Cumulative Normal and Probit Functions

- Φ = cumulative standard normal
- Probit function = inverse cumulative normal
- These do not have closed-form solutions and are approximated by software. This is the primary reason why logit has been more popular until recently: it was easy to calculate.



- left: $p_i = \Phi(x_i\beta)$; right: $probit(p_i) = x_i\beta$

Latent Variable Intuition for Binary DVs



$$y_i = \begin{cases} 1, \ if \ y^* > \tau \\ 0, \ if \ y^* \leq \tau \end{cases}$$

$$y_i^* = x_i\beta + \epsilon_i, \ \epsilon_i \sim g(0, \sigma^2)$$

Logit: assume $\tau = 0, \epsilon_i \sim logistic(\mu = 0, s = 1)$ Probit: assume $\tau = 0, \epsilon_i \sim N(0, 1)$

Latent Variable Intuition for Binary DVs

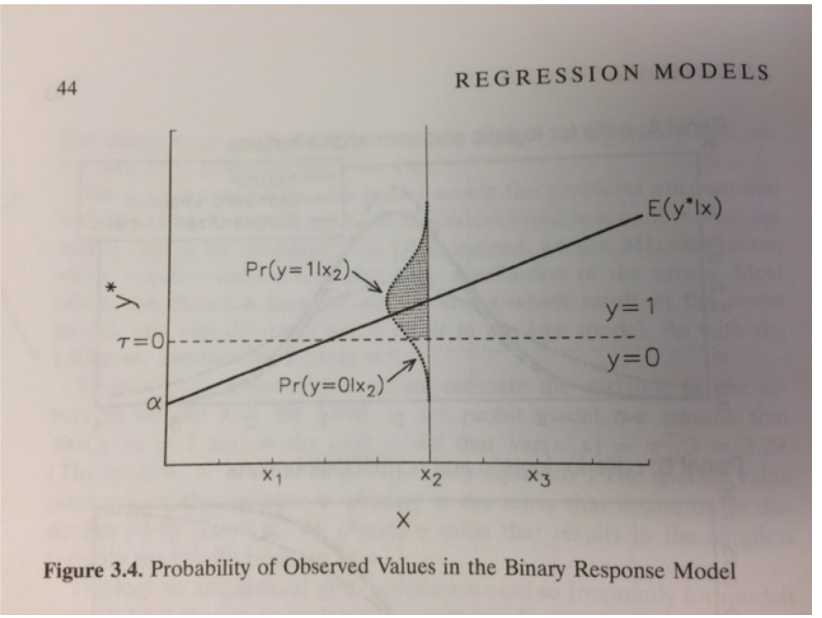


Figure 3.4. Probability of Observed Values in the Binary Response Model

- Long (1997). *Regression Models for Categorical and Limited Dependent Variables*

- Find the point $X\beta$ on y^*, and then take a random draw from a standard logistic or standard normal and add it to $X\beta$: if you are above 0, then $y = 1$, if you are below zero, the $y = 0$.

Translating Back to Probability Space

- Logit:

$$p_i = p(x_i\beta + \epsilon_i > 0) = p(-\epsilon_i < x_i\beta) = logistic(x_i\beta)$$

- Probit:

$$p_i = p(x_i\beta + \epsilon_i > 0) = p(-\epsilon_i < x_i\beta) = \Phi(x_i\beta)$$

- Estimating the coefficients
  - Estimate coefficients using **maximum likelihood estimation**
  - Basic Idea:
    - Find values of $\hat{\beta}_0$ and $\hat{\beta}_1$ that give observed data the maximum probability of occurring

Estimating Logit in R

```
m_logit <- glm(vio ~ gdppc, data = df, family = binomial(link = "logit"))
tidy(m_logit, conf.int = TRUE) %>% kable(format = "markdown", digits = 5)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.hig
(Intercept)	2.83420	1.19397	2.37376	0.01761	0.54264	5.2514
gdppc	-0.48064	0.14675	-3.27532	0.00106	-0.78219	-0.203

Logit: $\hat{P(vio|gdppc)} = logit(2.83420 - 0.48064 \times gdppc)$

Estimating Probit in R

```
m_probit <- glm(vio ~ gdppc, data = df, family = binomial(link = "probit"))
tidy(m_probit, conf.int = TRUE) %>% kable(format = "markdown", digits = 5)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.hig
(Intercept)	1.67285	0.69474	2.40786	0.01605	0.31968	3.066
gdppc	-0.28559	0.08367	-3.41336	0.00064	-0.45534	-0.123

Probit: $\hat{P(vio|gdppc)} = \Phi(1.67285 - 0.28559 \times gdppc)$

Interpreting the intercept: β_0

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_i$$

- When $x = 0$, **log-odds** of y are β_0
  - Won't use this interpretation in practice
- **When $x = 0$, odds of y are $\exp\{\beta_0\}$**

$$p_i = \Phi(\beta_0 + \beta_1 x_i)$$

- When $x = 0$, z-score of y are β_0
- The probit regression coefficients give the change in the z-score or probit index for a one unit change in the

Interpreting logistic slope coefficient β_1

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_i$$

If x is a quantitative predictor

- As x_i increases by 1 unit, we expect the log-odds of y to increase by β_1
- **As x_i increases by 1 unit, the odds of y multiply by a factor of $\exp\{\beta_1\}$**

Interpreting logistic slope coefficient β_1

If x is a categorical predictor. Suppose $x_i = k$

- The difference in the log-odds between group k and the baseline is β_1
- **The odds of y for group k are $\exp\{\beta_1\}$ times the odds of y for the baseline group.**

We interpret the *sign* of the coefficient but not the *magnitude*

- We should not compare the magnitude of the coefficients among different models because different models have different scales of coefficients.

Inference for coefficients

- The standard error is the estimated standard deviation of the sampling distribution of $\hat{\beta}_1$
- We can calculate the confidence interval based on the large-sample Normal approximations
- **CI for β_1:**

$$\hat{\beta}_1 \pm z^* SE(\hat{\beta}_1)$$

- **CI for $\exp\{\beta_1\}$:**

$$\exp\{\hat{\beta}_1 \pm z^* SE(\hat{\beta}_1)\}$$

Odds and Log Odds

Response Variable, Y

- $\text{Mean}(Y) = p$
  - p is the proportion of "yes" responses in the population
  - $\hat{p}$ is the proportion of "yes" responses in the sample
- $\text{Variance}(Y) = p(1-p)$
  - Sample variance: $\hat{p}(1-\hat{p})$
- $\text{Odds(Y=1)} = \frac{p}{1-p}$
  - Sample odds: $\frac{\hat{p}}{1-\hat{p}}$

Odds

- Given p, the population proportion of "yes" responses (i.e. "success"), the corresponding odds of a "yes" response is

$$\omega = \frac{p}{1-p}$$

- The *sample odds* are $\hat{\omega} = \frac{\hat{p}}{1-\hat{p}}$

- Ex: Suppose the sample proportion $\hat{p} = 0.3$. Then, the sample odds are

$$\hat{\omega} = \frac{0.3}{1-0.3} = 0.4286 \approx \text{ 2 in 5}$$

Properties of the odds

- odds ≥ 0
- If $p = 0.5$, then odds $= 1$
- If odds of "yes" $= \omega$, then the odds of "no" $= \frac{1}{\omega}$
- If odds of "yes" $= \omega$, then $p = \frac{\omega}{(1+\omega)}$

Odds, log odds, probability

```
# function probability -> log-odds
getlogit <- function(p) { log(p/(1-p)) }
# function log-odds -> probability
getpro <- function(x) { 1/(1 + exp(-x))}
# set odds
odds <- c(0.1, 0.2, 0.5, 1.0, 2.0, 5.0, 10.0)
# get log odds (logit)
(log_odds <- log(odds))
```

```
## [1] -2.3025851 -1.6094379 -0.6931472  0.0000000  0.6931472  1.6094379  2.302585
```

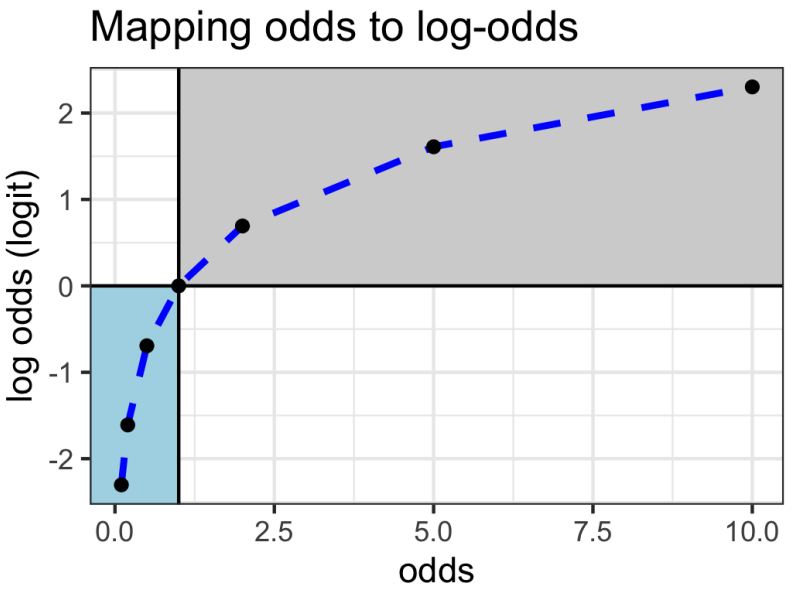
```
# convert from log-odds to probability
(probs <- getpro(log_odds))
```

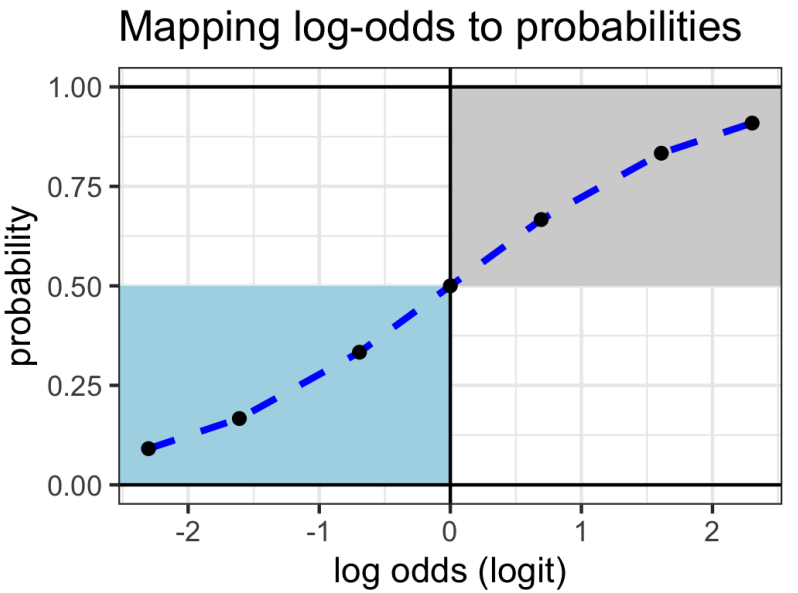
```
## [1] 0.09090909 0.16666667 0.33333333 0.50000000 0.66666667 0.83333333 0.9090909
```

```
# convert from probability to log-odds
getlogit(probs)
```

```
## [1] -2.3025851 -1.6094379 -0.6931472  0.0000000  0.6931472  1.6094379  2.302585
```

Plotting





- lightblue region: Y=0 more likely
- lightgray region: Y =1 more likely

Response Variable, `vio`

```
## # A tibble: 2 × 3
##     vio     n proportion
##   <dbl> <int>      <dbl>
## 1     0   112      0.757
## 2     1    36      0.243
```

- $\hat{p} = 0.243$
- Sample variance = 0.243 * (1- 0.243) = 0.183951
- Odds(Y = 1) = 0.243/(1 - 0.243) = 0.321004
- Odds(Y = 0) = 1 / 0.321004 = 3.1152263

Regressions with multiple predictors

- y: Whether a country in the sample is high risk of having violence.
- $p_i = P(y_i = 1 | \text{gdppc}_i, \text{population}_i, \text{regime type}_i)$: probability a country i is high risk for violence given their GDP per capita, population, and regime type.

Let's consider fitting a multiple linear regression model. Below are 3 possible response variables.

Model 1: $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 \text{gdppc} + \hat{\beta}_2 \text{pop} + \hat{\beta}_3 \text{regime}$

Model 2: $log(\frac{\hat{p}_i}{1-\hat{p}_i}) = \hat{\beta}_0 + \hat{\beta}_1 \text{gdppc} + \hat{\beta}_2 \text{pop} + \hat{\beta}_3 \text{regime}$

Model 3: $\hat{p}_i = \Phi(\hat{\beta}_0 + \hat{\beta}_1 \text{gdppc} + \hat{\beta}_2 \text{pop} + \hat{\beta}_3 \text{regime})$

```
library(stargazer)
```

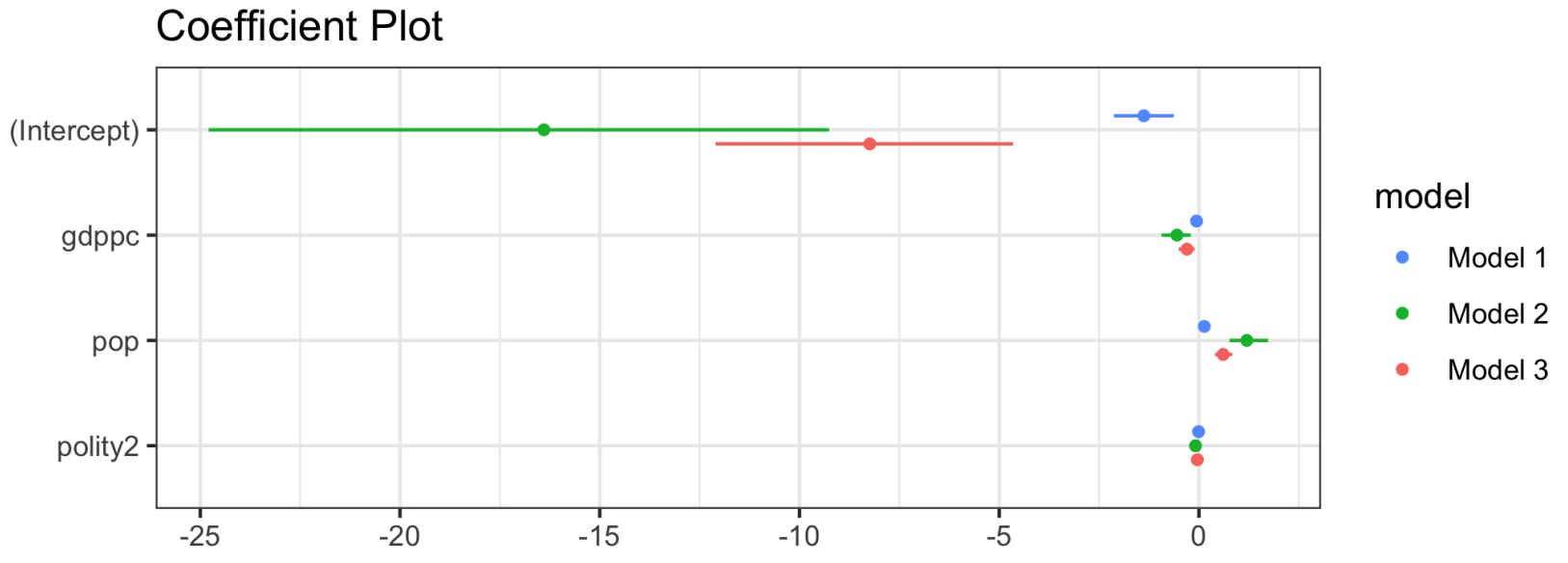
Regression output I: table

```
stargazer(m1, m2, m3, header = FALSE, type = "html", single.row=TRUE, ci=TRUE, c
          dep.var.labels.include = FALSE)
```

	Dependent variable:		
	OLS	*logistic*	*probit*
	(1)	(2)	(3)
gdppc	-0.060*** (-0.101, -0.020)	-0.551*** (-0.914, -0.188)	-0.300*** (-0.497, -0.102)
pop	0.134*** (0.095, 0.172)	1.198*** (0.721, 1.675)	0.608*** (0.372, 0.843)
polity2	-0.009* (-0.019, 0.002)	-0.087* (-0.174, 0.001)	-0.040* (-0.088, 0.008)
Constant	-1.380*** (-2.124, -0.636)	-16.397*** (-24.094, -8.699)	-8.239*** (-12.202, -4.277)
Observations	148	148	148
R^2	0.313		
Adjusted R^2	0.299		
Log Likelihood		-53.284	-54.195
Akaike Inf. Crit.		114.568	116.391
Residual Std. Error	0.360 (df = 144)		
F Statistic	21.906*** (df = 3; 144)		
Note:			*p<0.1; **p<0.05; ***p<0.01

Regression output II: graph

```
library(dotwhisker)
dwplot(list(m1, m2, m3), conf.level = .95, show_intercept = TRUE) +
  theme_bw() + ggtitle("Coefficient Plot")
```



Prediction

Using the model for prediction

- We are often interested in predicting whether a given observation will have a "yes" response
- To do so
  - Use the logistic regression model to calculate the predicted log-odds that an observation has a "yes" response
  - Then, use the log-odds to calculate the predicted probability of a "yes" response
  - Then, use the predicted probabilities to classify the observation as having a "yes" or "no" response

Calculating the predicted probability

$$\hat{p}_i = \frac{\exp\{\hat{\beta}_0 + \hat{\beta}_1 x_i\}}{1 + \exp\{\hat{\beta}_0 + \hat{\beta}_1 x_i\}}$$

$$\log\left(\frac{\hat{p}_i}{1 - \hat{p}_i}\right) = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

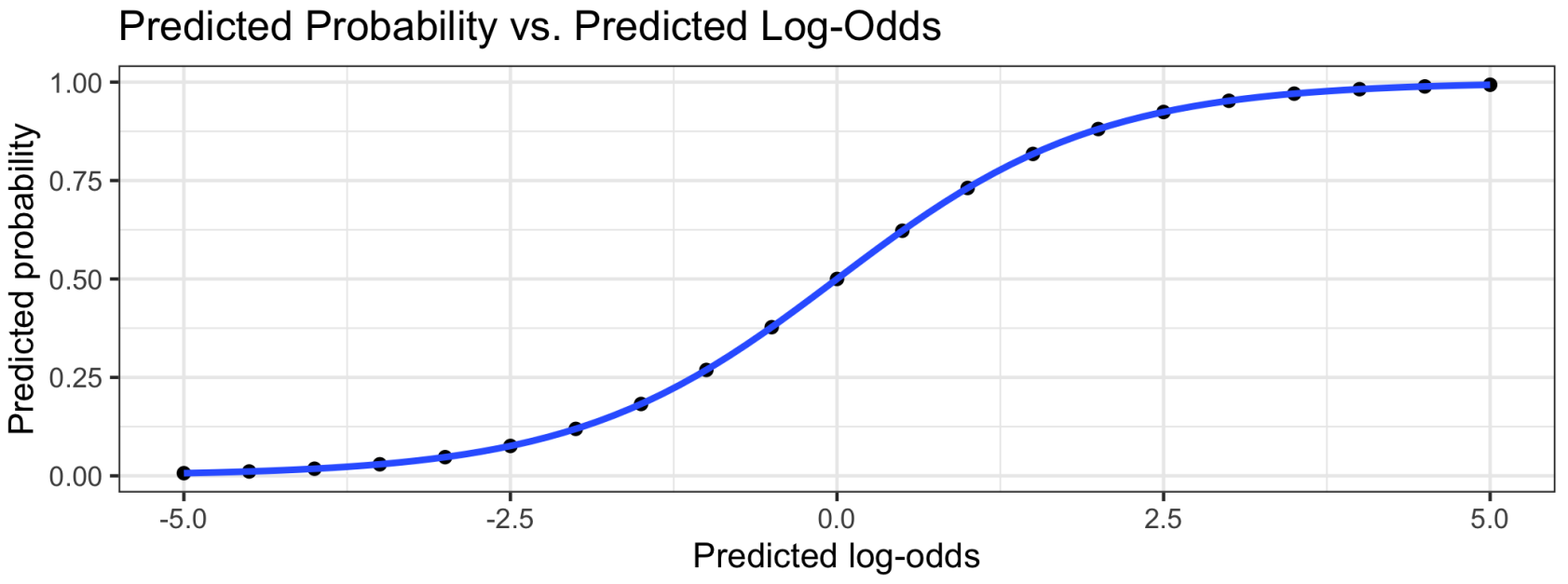
$$\Rightarrow \exp\left\{\log\left(\frac{\hat{p}_i}{1 - \hat{p}_i}\right)\right\} = \exp\{\hat{\beta}_0 + \hat{\beta}_1 x_i\}$$

$$\Rightarrow \frac{\hat{p}_i}{1 - \hat{p}_i} = \exp\{\hat{\beta}_0 + \hat{\beta}_1 x_i\}$$

$$\Rightarrow \hat{p}_i = \frac{\exp\{\hat{\beta}_0 + \hat{\beta}_1 x_i\}}{1 + \exp\{\hat{\beta}_0 + \hat{\beta}_1 x_i\}} = \frac{1}{1 + exp(-\{\hat{\beta}_0 + \hat{\beta}_1 x_i\})}$$

$\hat{p}$ vs. $\widehat{\text{log-odds}}$

$$\hat{p}_i = \frac{\exp(\hat{\beta}_0 + \hat{\beta}_1 x_i)}{1 + \exp(\hat{\beta}_0 + \hat{\beta}_1 x_i)} = \frac{\exp(\widehat{\text{log-odds}})}{1 + \exp(\widehat{\text{log-odds}})}$$



Predicted violence for a country

- Suppose a country whose gdppc is 10, pop is 11, and regime type is 8
- Predicted log-odds that this country is high risk for violence:

$$\log\left(\frac{\hat{p}}{1-\hat{p}}\right) = -16.39683 - 0.55098 \times \text{gdppc} + 1.19806 \times \text{pop} - 0.086$$

$$\log\left(\frac{\hat{p}}{1-\hat{p}}\right) = -16.39683 - 0.55098 \times 10 + 1.19806 \times 11 - 0.$$

- The probability this country is high risk for violence:

$$\hat{p}_i = \frac{\exp\{-9.42173\}}{1+\exp\{-9.42173\}} = 8.093931e-05$$

Predictions in R

```
newdf <- data.frame(gdppc = 10, pop = 11, polity2 = 8,

vio = as.factor(0))
```

- Predicted log-odds

```
predict(m2, newdf)
```

```
##          1
## -9.421727
```

- Predicted probabilities

```
predict(m2, newdf, type = "response")
```

```
##             1
## 8.093953e-05
```

Predicted probabilities with CIs

```
pred <- predict(m2, newdf, se.fit = TRUE)
upr <- pred$fit + (1.96*pred$se.fit)
lwr <- pred$fit - (1.96*pred$se.fit)
fit <- pred$fit
getpro(fit);getpro(upr);getpro(lwr)
```

```
##             1
## 8.093953e-05
```

```
##           1
## 0.002260058
```

```
##             1
## 2.892599e-06
```

- Is this country high risk?

The probability the country is at risk for violence is 0.

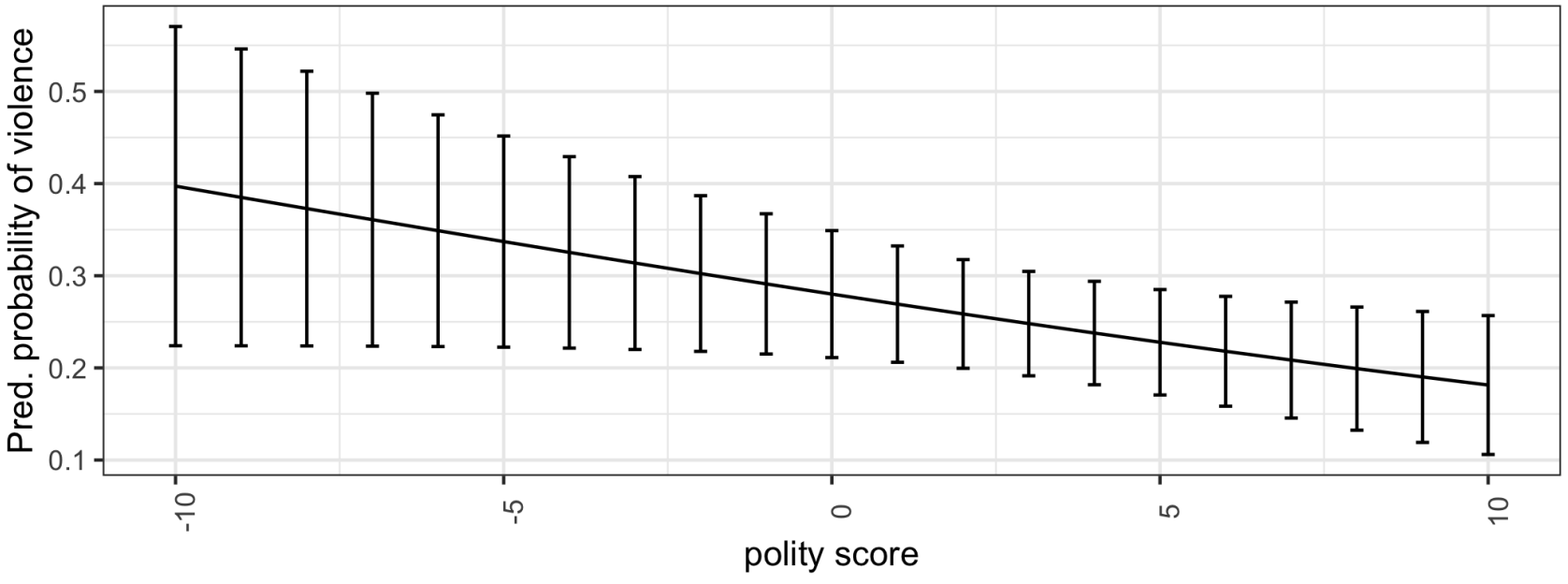
Visual representation of results

```
library(margins)
library(prediction)

predict_m2 <- prediction::prediction(m2,
            at = list(polity2 = unique(model.frame(m2)$polity2)))

summary(predict_m2)
```

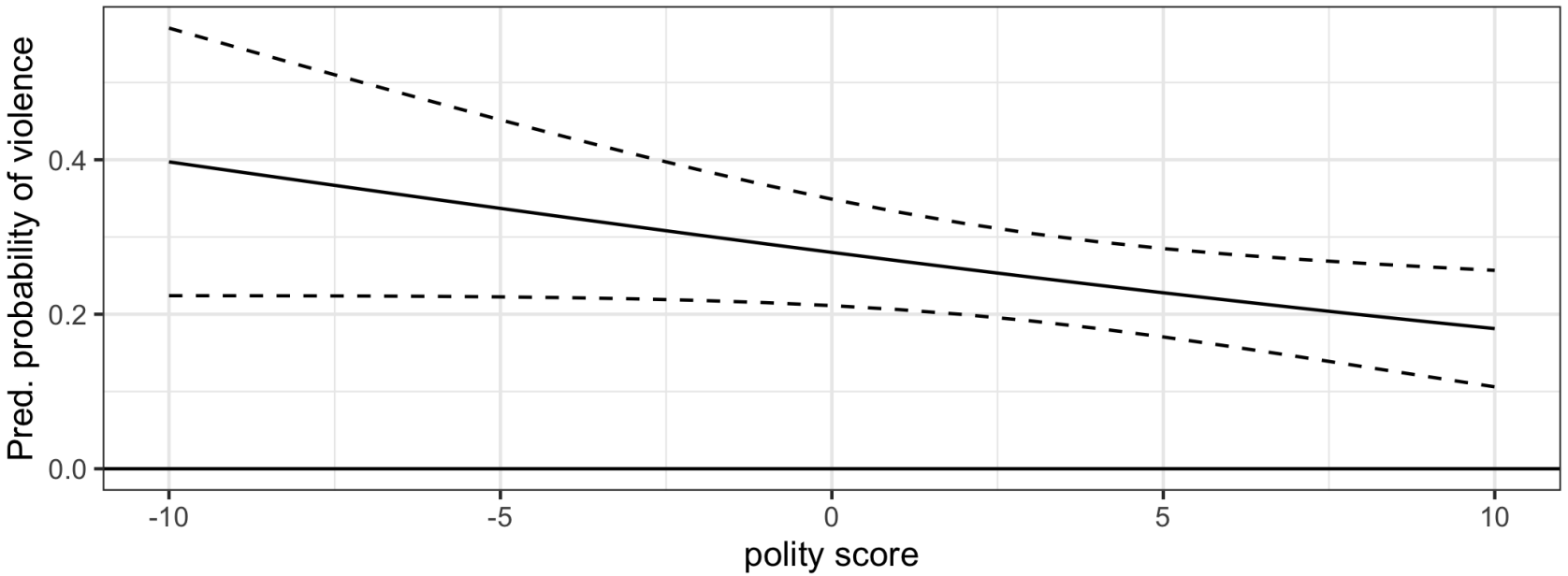
```
##  at(polity2) Prediction      SE     z          p  lower  upper
##          -10     0.3972 0.08838 4.495 6.960e-06 0.2240 0.5705
##           -9     0.3850 0.08217 4.686 2.792e-06 0.2240 0.5461
##           -8     0.3729 0.07604 4.904 9.411e-07 0.2238 0.5219
##           -7     0.3608 0.07002 5.153 2.560e-07 0.2236 0.4981
##           -6     0.3489 0.06414 5.439 5.349e-08 0.2232 0.4746
##           -5     0.3371 0.05845 5.766 8.099e-09 0.2225 0.4516
##           -4     0.3254 0.05301 6.138 8.339e-10 0.2215 0.4293
##           -3     0.3138 0.04786 6.556 5.517e-11 0.2200 0.4076
##           -2     0.3024 0.04311 7.015 2.298e-12 0.2179 0.3869
##           -1     0.2912 0.03883 7.498 6.493e-14 0.2150 0.3673
##            0     0.2801 0.03516 7.967 1.624e-15 0.2112 0.3490
##            1     0.2692 0.03220 8.360 6.281e-17 0.2061 0.3323
##            2     0.2585 0.03009 8.591 8.616e-18 0.1996 0.3175
##            3     0.2481 0.02890 8.583 9.273e-18 0.1914 0.3047
##            4     0.2378 0.02863 8.306 9.881e-17 0.1817 0.2939
##            5     0.2278 0.02919 7.806 5.926e-15 0.1706 0.2850
##            6     0.2180 0.03040 7.171 7.423e-13 0.1584 0.2776
```

Graphs

```
ggplot(summary(predict_m2), aes(x = `at(polity2)`, y = Prediction,
                                ymin = lower, ymax = upper,group = 1)) +
  geom_line() +geom_errorbar(width = 0.2) +
  theme(axis.text.x = element_text(angle = 90)) +
  labs(x = "polity score",y = "Pred. probability of violence")
```

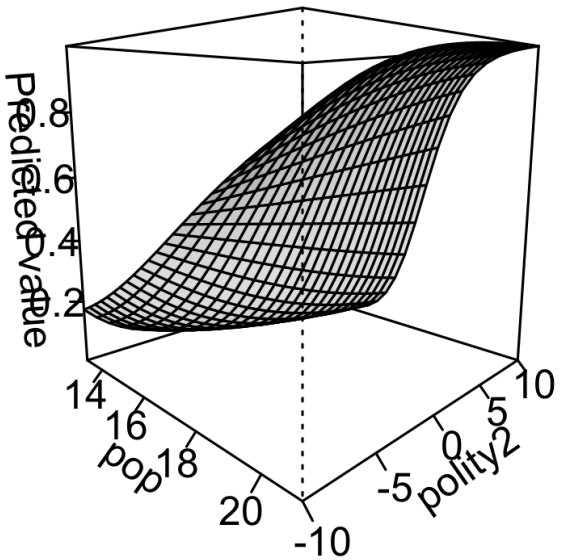


```
ggplot(summary(predict_m2), aes(x = `at(polity2)`)) +
    geom_line(aes(y = Prediction)) +
    geom_line(aes(y = upper), linetype = 2)+
    geom_line(aes(y = lower), linetype = 2) +
    geom_hline(yintercept = 0) +
    labs(x = "polity score",
    y = "Pred. probability of violence")
```



Interaction

```
int1 <- glm(vio ~ gdppc + pop*polity2, data = df, family = binomial(link = "lo
library(margins)
persp(int1, "pop", "polity2", what = "prediction", type = "response")
```



Panel data for logit and Probit models: `bife`

- Stammann, A., F. Heiss, and D. McFadden (2016). "Estimating Fixed Effects Logit Models with Large Panel Data". Working paper.
- `formula must be of type y ~ x | id where the id refers to an individual identifier (fixed effect category)`

```
library(bife)
m1 <- glm(vio ~ gdppc + pop + polity2, data = IRdata, family = binomial(link =
m2 <- glm(vio ~ gdppc + pop + polity2, data = IRdata, family = binomial(link =
m_logit1 <- bife(vio ~ gdppc + pop + polity2 | year, model = "logit", data = I
m_probit1 <- bife(vio ~ gdppc + pop + polity2 | year, model = "probit", data =
m_logit2 <- bife(vio ~ gdppc + pop + polity2 | ccode, model = "logit", data =
m_probit2 <- bife(vio ~ gdppc + pop + polity2 | ccode, model = "probit", data
```

```
library(texreg)
htmlreg(list(m1, m2, m_logit1, m_probit1,  m_logit2, m_probit2))
```

Results

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
(Intercept)	-10.40***	-5.72***				
	(0.61)	(0.33)				
gdppc	-0.54***	-0.30***	-0.55***	-0.31***	-0.60**	-0.35***
	(0.03)	(0.02)	(0.03)	(0.02)	(0.19)	(0.10)
pop	0.82***	0.45***	0.83***	0.46***	-0.78	-0.40
	(0.04)	(0.02)	(0.04)	(0.02)	(0.59)	(0.34)
polity2	-0.04***	-0.02***	-0.03***	-0.02***	-0.08***	-0.05***
	(0.01)	(0.00)	(0.01)	(0.00)	(0.02)	(0.01)
AIC	3164.32	3168.54				
BIC	3189.17	3193.39				
Log Likelihood	-1578.16	-1580.27	-1566.71	-1568.82	-727.91	-729.26
Deviance	3156.32	3160.54	3133.41	3137.64	1455.82	1458.52
Num. obs.	3687	3687	3687	3687	1580	1580

***p < 0.001; **p < 0.01; *p < 0.05

Statistical models

Panel data for logit and Probit models: lme4

```
library(lme4)
mlm_1 <- lmer(gdppc ~ miliper + vio + pop + polity2 + (1 | year), data = IRdat
mlm_2 <- glmer(vio ~ gdppc + pop + polity2 + (1 | year), data = IRdata, family
mlm_3 <- glmer(vio ~ gdppc + pop + polity2 + (1 | ccode), data = IRdata, famil
```

```
htmlreg(list(mlm_1, mlm_2, mlm_3))
```

Results

	Model 1	Model 2	Model 3
(Intercept)	5.53***	-10.40***	-18.46***
	(0.27)	(0.61)	(3.24)
miliper	1.05***		
	(0.05)		
vio	-1.01***		
	(0.06)		
pop	0.09***	0.82***	1.49***
	(0.02)	(0.04)	(0.21)
polity2	0.10***	-0.04***	-0.11***
	(0.00)	(0.01)	(0.02)
gdppc		-0.54***	-1.10***
		(0.03)	(0.13)
AIC	12670.87	3166.32	1885.68
BIC	12714.36	3197.38	1916.74
Log Likelihood	-6328.44	-1578.16	-937.84
Num. obs.	3687	3687	3687
Num. groups: year	26	26	
Var: year (Intercept)	0.20	0.00	
Var: Residual	1.77		
Num. groups: ccode			156
Var: ccode (Intercept)			11.92

***p < 0.001; **p < 0.01; *p < 0.05

Statistical models

Confusion Matrix

- We can use the predicted probability to predict the outcome for a given observation
  - In other words, we can classify the observations into two groups: "yes" and "no"
- **How**: Establish a threshold such that $y = 1$ if predicted probability is greater than the threshold $(y = 0 \text{ otherwise})$
- To assess the accuracy of our predictions, we can make a table of the observed (actual) response versus the predicted response. This table is the confusion matrix
- We can use this table to calculate the proportion of observations that were misclassifed. This is the misclassification rate.

Confusion Matrix

Suppose we use 0.3 as the threshold to classify observations

```
threshold <- 0.3
m_aug <- augment(m1, type.predict = "response",
                      type.residuals = "deviance")
```

```
m_aug %>%
  mutate(risk_predict = if_else(.fitted > threshold, "Yes", "No")) %>%
  group_by(vio, risk_predict) %>%
  summarise(n = n()) %>%
  kable(format="markdown")
```

vio	risk_predict	n
0	No	2115
0	Yes	606
1	No	219
1	Yes	747

Confusion matrix

vio	risk_predict	n
0	No	2115
0	Yes	606
1	No	219
1	Yes	747

What proportion of observations were misclassified?

What is the disadvantage of relying on the confusion matrix to assess the accuracy of the model?

Confusion matrix: 2 X 2 table

In practice, you often see the confusion matrix presented as a 2×2 table as shown below:

```
m_aug %>%
  mutate(risk_predict = if_else(.fitted > threshold, "Yes", "No")) %>%
  group_by(vio, risk_predict) %>%
  summarise(n = n()) %>%
  spread(risk_predict, n) %>%
  kable(format="markdown")
```

vio	No	Yes
0	2115	606
1	219	747

Confusion matrix by hand

	Predicted Positive	Predicted Negative	
Actual Positive	TP *True Positive*	FN *False Negative*	Sensitivity $\frac{TP}{(TP+FN)}$
Actual Negative	FP *False Positive*	TN *True Negative*	Specificity $\frac{TN}{(TN+FP)}$
	Precision $\frac{TP}{(TP+FP)}$	Negative Predictive Value $\frac{TN}{(TN+FN)}$	Accuracy $\frac{TP+TN}{(TP+TN+FP+FN)}$

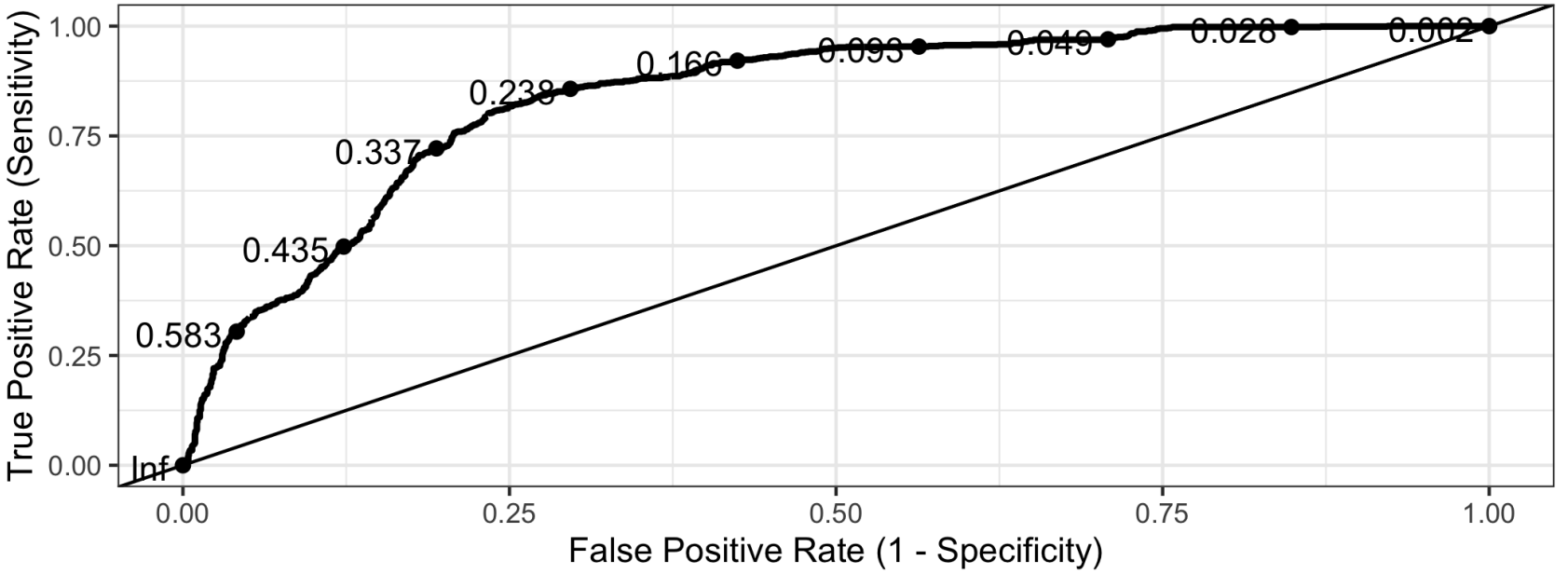
- true positives (TP): These are cases in which we predicted yes, and they do happen.
- true negatives (TN): We predicted no, and they don't happen.
- false positives (FP): We predicted yes, but they don't actually happen. (Also known as a "Type I error.")
- false negatives (FN): We predicted no, but they actually do happen. (Also known as a "Type II error.")

Sensitivity & Specificity

- Sensitivity: Proportion of observations with $y = 1$ that have predicted probability above a specified threshold
  - Called **true positive rate** (y-axis) (also referred to as Recall)
- Specificity: Proportion of observations with $y = 0$ that have predicted probability below a specified threshold
  - (1 - specificity) called **false positive rate** (x-axis)
- Precision: Of the relevant cases identified, how many errors were made? (i.e., how precise are my predictions)
  - *positive predictive value of the classifier*
- What we want:
  - High sensitivity
  - Low values of 1-specificity

Receiver Operating Characteristic (ROC) curve

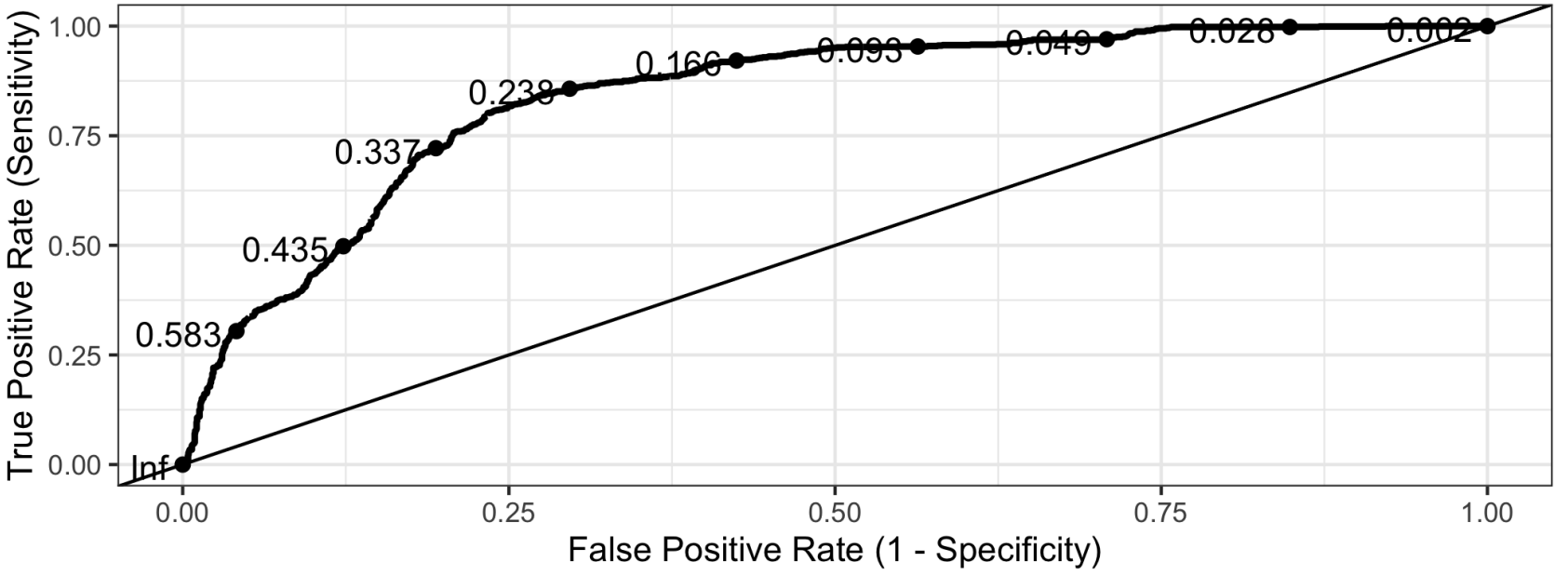
```
library(plotROC) #extension of ggplot2
(roc_curve <- ggplot(m_aug, aes(d = as.numeric(vio) - 1, m = .fitted)) +
  geom_roc(n.cuts = 10, labelround = 3) +
  geom_abline(intercept = 0) +
  labs(x = "False Positive Rate (1 - Specificity)",
       y = "True Positive Rate (Sensitivity)") )
```



Area under curve (AUC)

We can use the area under the curve (AUC) as one way to assess how well the logistic model fits the data

- $AUC = 0.5$ very bad fit (no better than a coin flip)
- AUC close to 1: good fit

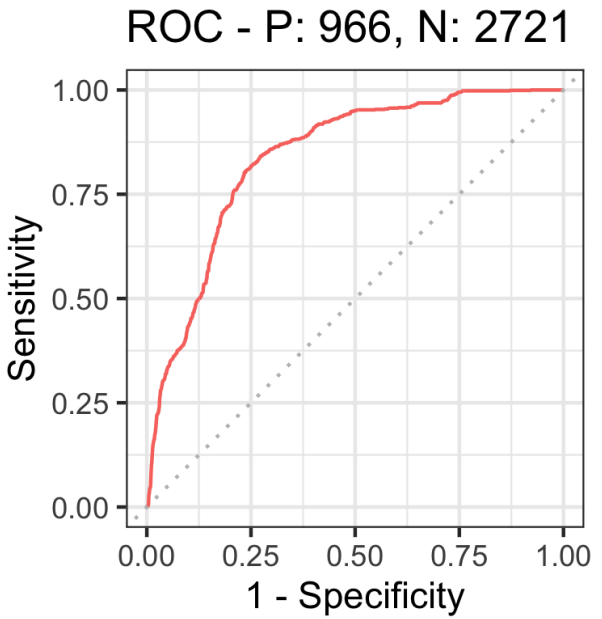


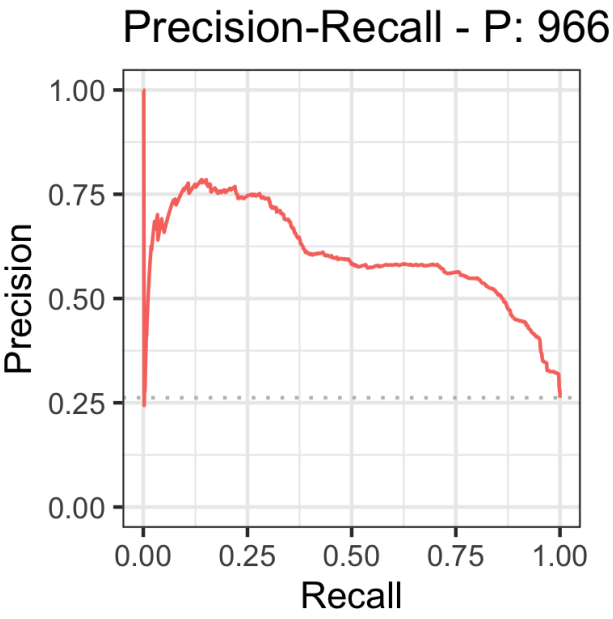

```
calc_auc(roc_curve)$AUC
```

```
## [1] 0.8394968
```

ROC and Precision-Recall curves

```
library(precrec)
precrec_obj <- evalmod(scores = m_aug$.fitted, labels = as.numeric(m_aug$vio)
autoplot(precrec_obj)
```





Checking Assumptions for Logit model

Assumptions for logistic regression

```
logit_m <- glm(vio ~ dem + gdppc, data = df, family = binomial(link = "log
```

We want to check the following assumptions for the logistic regression model:

- Linearity: Is there a linear relationship between the log-odds and the predictor variables?
- Randomness: Was the sample randomly selected? Or can we reasonably treat it as random?
- Independence: There is no obvious relationship between observations

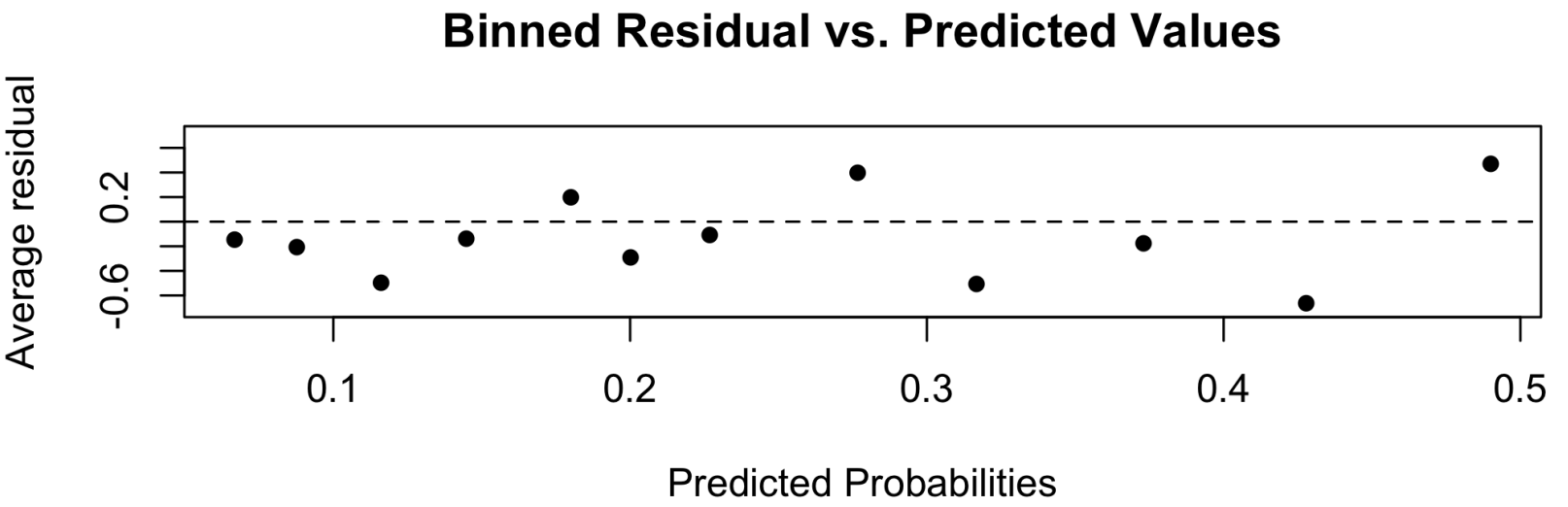
Linearity: binned residual plots

- It is not useful to plot the raw residuals, so we will examine *binned* residual plots
- **When examining binned residuals**
- Plot should have no discernible pattern or trend
  - Nonlinear trend may be indication that squared term or log transformation of predictor variable required
- If bins have average residuals with large magnitude
  - Look at averages of other predictor variables across bins
  - Interaction may be required if large magnitude residuals correspond to certain combinations of predictor variables

Binned plot vs. predicted values

- Use the `binnedplot` function in the **arm** package.
  - *Tip: Don't load the **arm** package to avoid conflicts with tidyverse*

```
m_aug <- augment(logit_m, type.predict = "response",
                         type.residuals = "deviance")

arm::binnedplot(x = m_aug$.fitted, y = m_aug$.resid,
                xlab = "Predicted Probabilities",
                main = "Binned Residual vs. Predicted Values",
                col.int = FALSE)
```


Binned Residual vs. Predicted Values
Average residual
-0.6
0.2
0.1
0.2
0.3
0.4
0.5
Predicted Probabilities

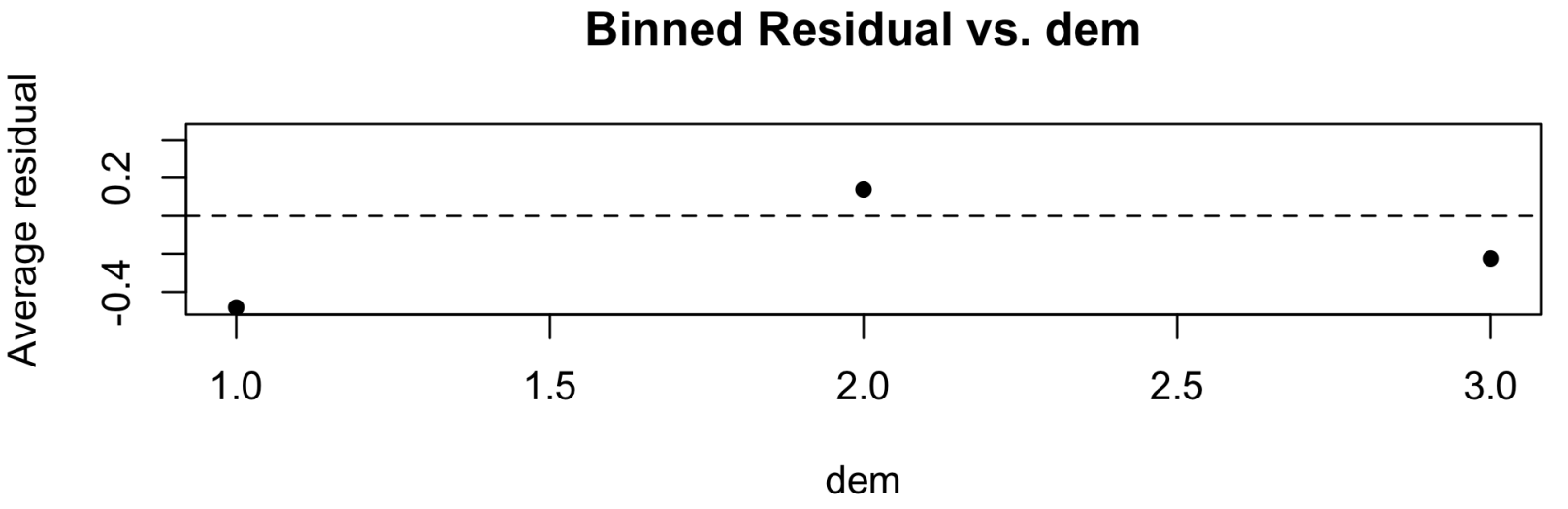
Making binned residual plot

- Calculate raw residuals $(y_i - \hat{\pi}_i)$
- Order observations either by the values of the predicted probabilities (or by numeric predictor variable)
- Use the ordered data to create g bins of approximately equal size. Default value: $g = \sqrt{n}$
- Calculate average residual value in each bin
- Plot average residuals vs. average predicted probability (or average predictor value)

Residuals vs. dem

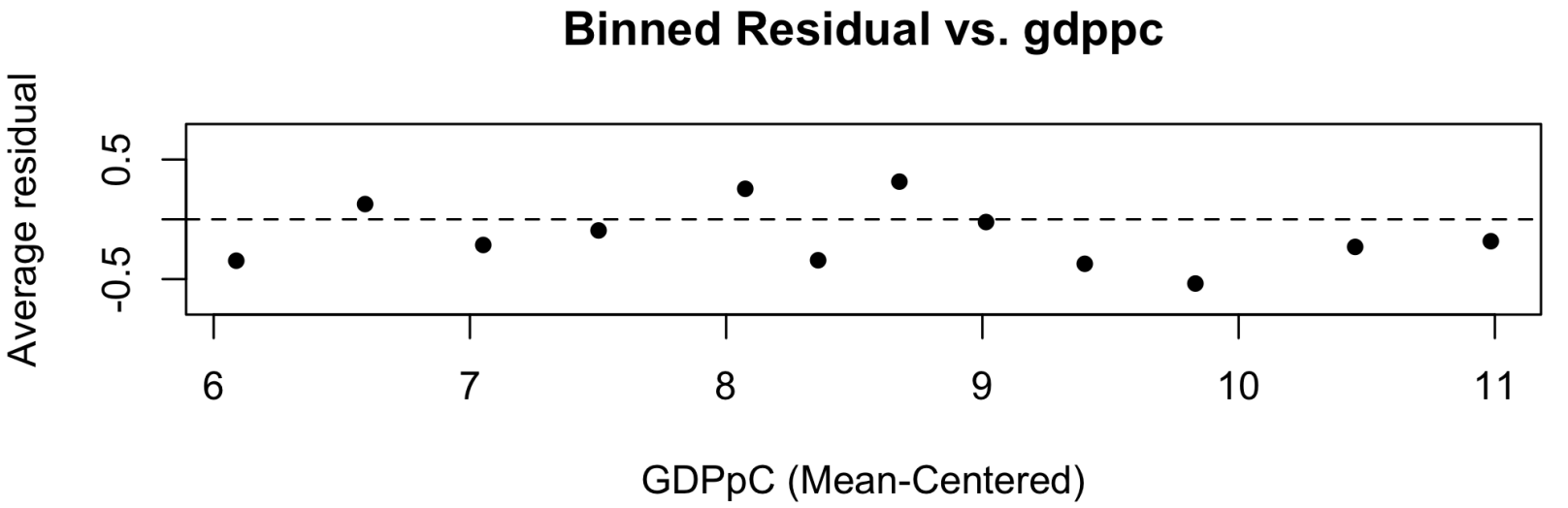
Make binned plot with predictor on x axis

```
arm::binnedplot(x = m_aug$dem,
                y = m_aug$.resid,
                col.int = FALSE,
                xlab = "dem",
                main = "Binned Residual vs. dem")
```



Residuals vs. gdppc

```
arm::binnedplot(x = m_aug$gdppc,
                y = m_aug$.resid,
                col.int = FALSE,
                xlab = "GDPpC (Mean-Centered)",
                main = "Binned Residual vs. gdppc")
```



Residuals vs. categorical predictors

- Calculate average residual for each level of the predictor
  - Are all means close to 0? If not, there is a problem with model fit.

```
m_aug %>%
  group_by(dem) %>%
  summarise(mean_resid = mean(.resid))
```

```
## # A tibble: 3 × 2
##     dem mean_resid
##   <dbl>      <dbl>
## 1     1     -0.482
## 2     2      0.138
## 3     3     -0.224
```

Randomness

Assess randomness based on a description of the data collection

- Was the sample randomly selected?
- If the sample was not randomly selected, is there reason to believe the observations in the sample differ systematically from the population of interest?

What do you conclude about the randomness assumption for our dataset?

Independence

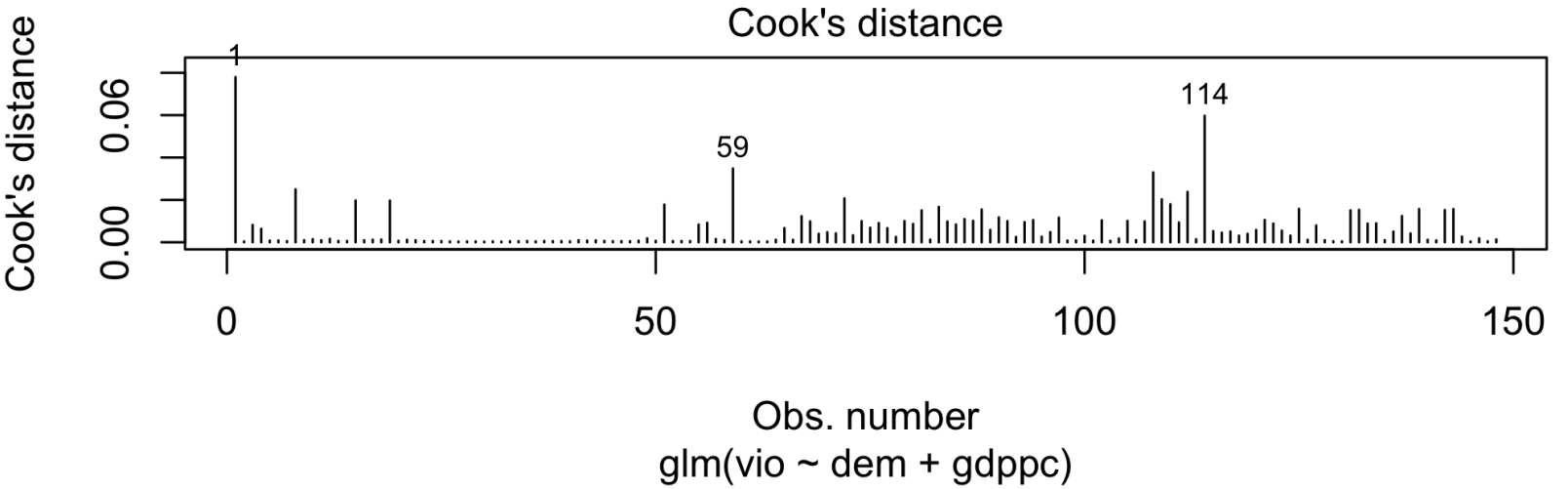
Assess independence based on a description of the data collection

- Is there an obvious relationship between observations?
  - This assumption is most often violated when data was collected over time or there is a spatial relationship between observations?

What do you conclude about the independence assumption for our dataset?

Checking influential values

```
plot(logit_m, which = 4, id.n = 3)
```



- not all outliers are influential observations.

```r
# Extract model results
augment(logit_m) %>%
  mutate(index = 1:n())%>%
  top_n(3, .cooksd)
```

```
## # A tibble: 3 × 10
##     vio   dem gdppc .fitted .resid .std.resid   .hat .sigma .cooksd index
##   <dbl> <dbl> <dbl>   <dbl>  <dbl>      <dbl>  <dbl>  <dbl>   <dbl> <int>
## 1     1     3 10.9   -2.64    2.33       2.35 0.0162   1.00  0.0780     1
## 2     1     1  8.61  -0.648   1.46       1.50 0.0494   1.01  0.0348    59
## 3     1     3 10.5   -2.42    2.24       2.26 0.0154   1.00  0.0597   114
```

Multicollinearity in Logit model

```
car::vif(logit_m )
```

```
##      dem    gdppc
## 1.000259 1.000259
```

- As a rule of thumb, a VIF value that exceeds 5 indicates a problematic amount of collinearity.