SOC542 Statistical Methods in Sociology II

Binary outcomes I

Thomas Davidson

Rutgers University

March 21, 2022

Course updates

- Identify suitable paper for replication
  - Email me your choice of replication paper by Friday 3/25
- Homework 2 grades and comments released
- Next homework released on Wednesday, due 4/1
  - Logistic regression
  - Interaction terms

Plan

- Linear probability model
- Logistic regression
- Probit regression

Linear probability model

Definition

- A binary outcome variable y consists of two possible values, 0 or 1.
  - e.g. $y \sim Binomial(n, p)$ is a sequence of n observations, where $P(y_i = 1) = p$.
- The **linear probability model (LPM)** is used to model the *probability* of binary dependent variables as a *linear* function of predictors.

Linear probability model

Specification

- The LPM is estimated using OLS:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + u$$

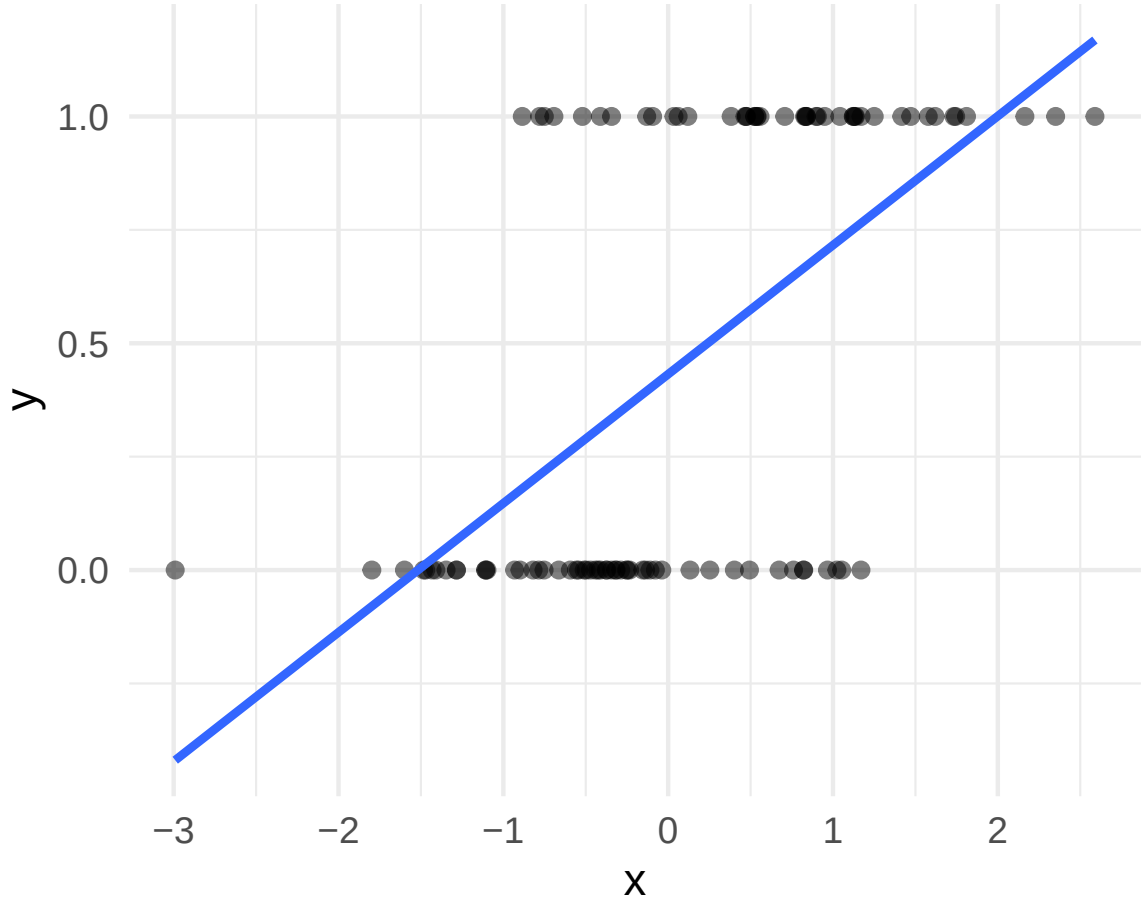
- Thus,

$$P(y = 1|x_1, x_2, ...x_k) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + u$$

- The coefficient β_i represents the change in probability that $y = 1$ associated with a unit-change in x_i, holding other regressors constant.

Linear probability model

Fitting a line to a simulated binary outcome



Linear probability model

Example: Diffusion of Microfinance[1]

- Survey data from 75 villages in Karnataka, India
  - Focus only on women and 72 villages
  - Listwise deletion used to drop respondents missing key variables
  - N = 9064
- Dependent variable:
  - Membership in a micro-finance Self-Help Group (SHG), N = 3357
- Independent variables:
  - Age (continuous) and age squared
  - Caste (dummy, low/high)
- Fixed-effects:
  - Village (dummy)

[1] Data from Banerjee, A., A. G. Chandrasekhar, E. Duflo, and M. O. Jackson. 2013. "The Diffusion of Microfinance." *Science* 341 (6144): 1236498–1236498. Link to paper. Harvard Dataverse link

Linear probability model

Example: Predicting Self-Help Group membership[2]

```
lpm <- lm(shg ~ age + I(age^2) + caste +
              as.factor(village), data = data)
```

[2] Based on a propensity score model in Davidson, Thomas, and Paromita Sanyal. 2017. "Associational Participation and Network Expansion: Microcredit Self-Help Groups and Poor Women's Social Ties in Rural India." *Social Forces* 95 (4): 1695–1724. https://doi.org/10.1093/sf/sox021.

Linear probability model

Example: Predicting Self-Help Group membership

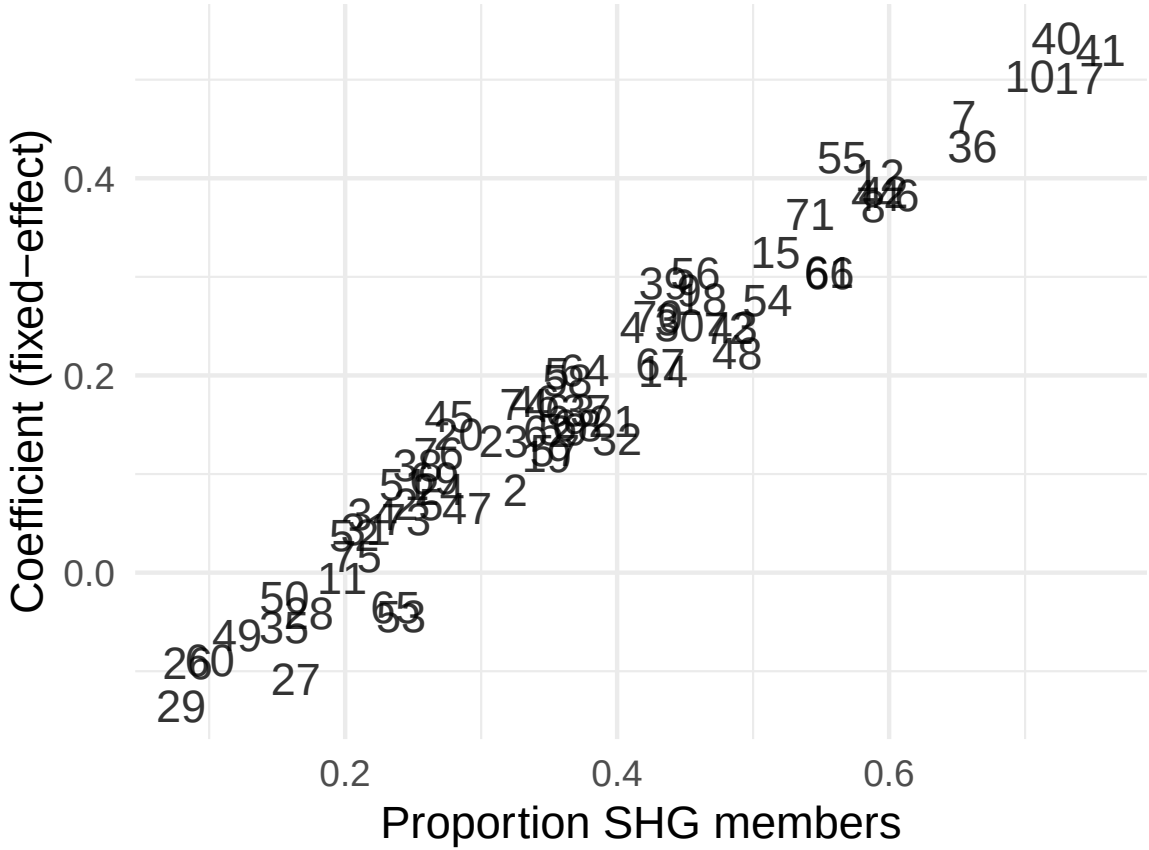
	Model 1
(Intercept)	-0.702***
	(0.057)
Age	0.047***
	(0.002)
Age^2	-0.001***
	(0.000)
Caste (Lower)	0.060***
	(0.011)
Num.Obs.	9064
Log.Lik.	-5430.095
F	24.515

Village fixed-effects omitted.

+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

Linear probability model

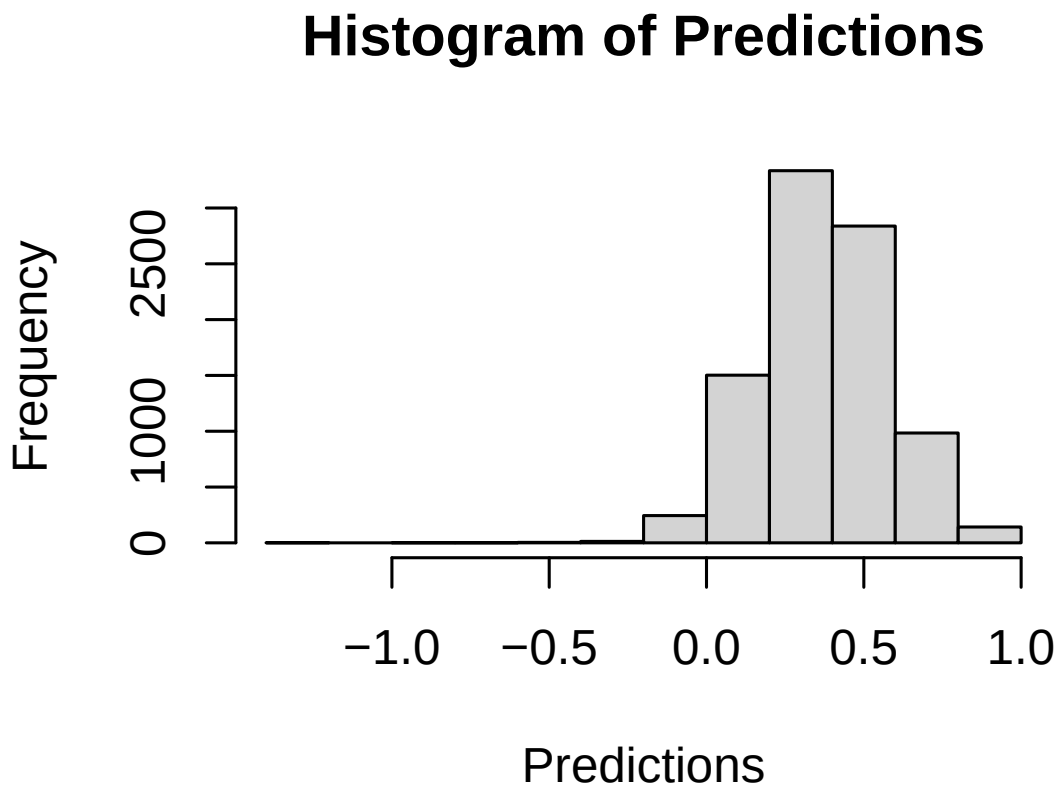
Village fixed-effects



Village 1 is reference category.

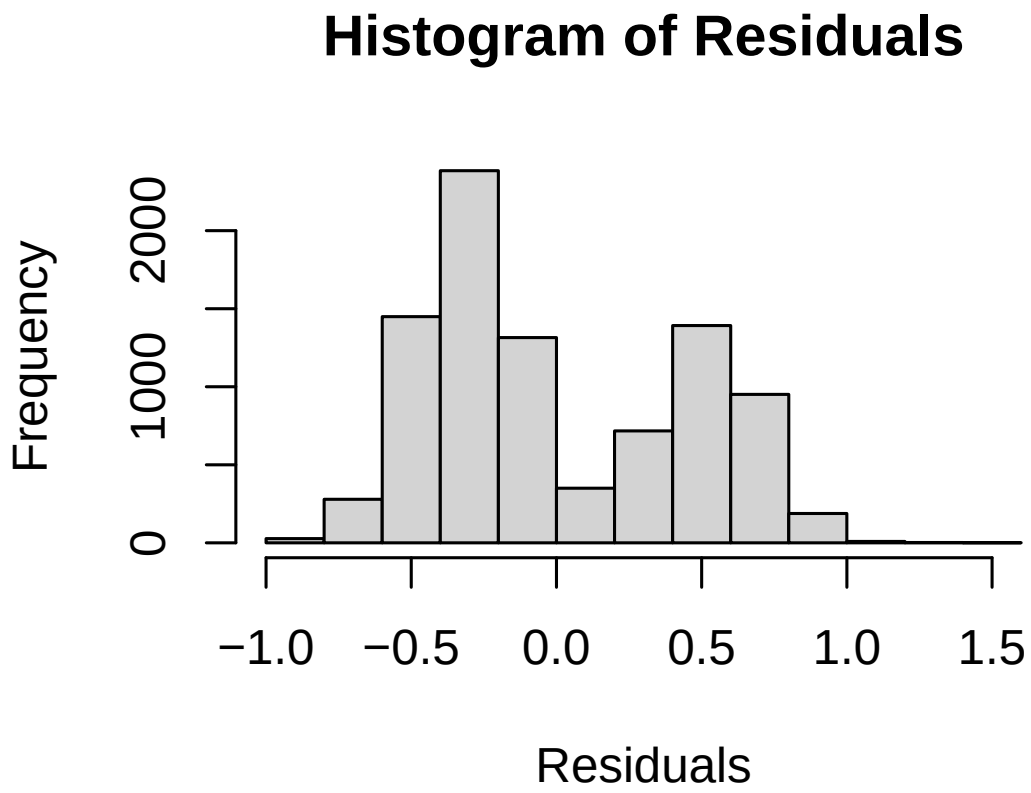
Linear probability model

Predicted values



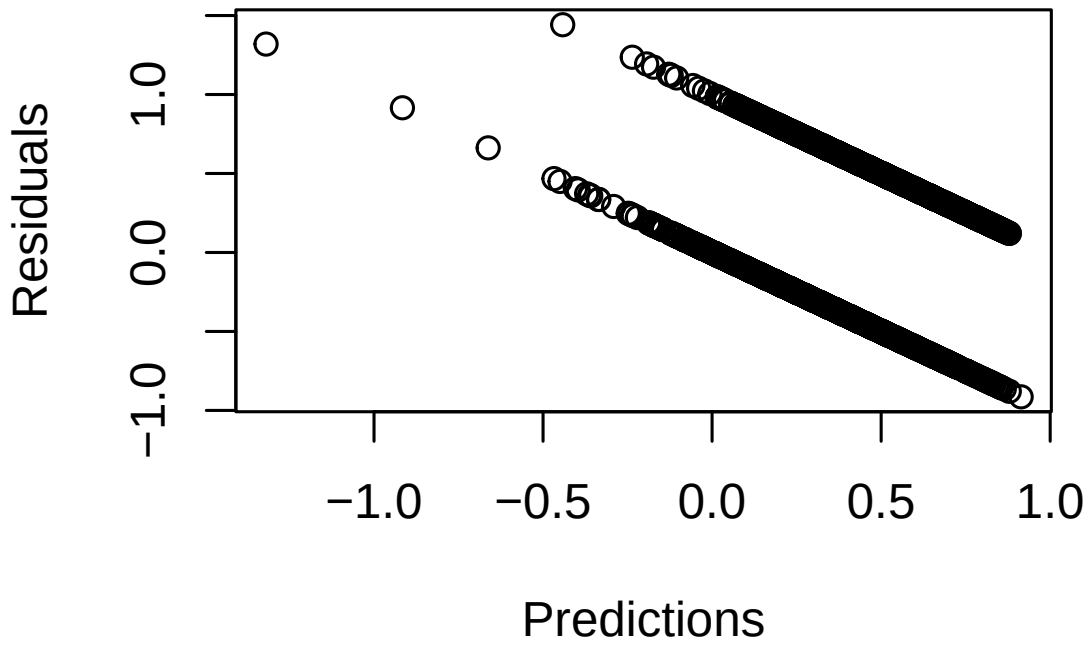
Linear probability model

Residuals



Linear probability model

Predicted values and residuals



Linear probability model

Limitations

- Unrealistic predictions
  - Nothing constrains predictions to be probabilities bounded by [0,1] so the model can make unrealistic predictions
- Heteroskedastic errors
  - Requires the use of heterosekasticity-robust standard errors[3]
- R^2 no longer reliable
  - Under what circumstances could $R^2 = 1$ be achieved with a binary outcome?

[3] We will discuss this topic in more detail in Week 12.

Logistic regression

Addressing the limitations

- We can address the limitations of the LPM by using a different functional form to ensure that predicted values are constrained to the $[0, 1]$ range
- To do this must extend the linear model by using a **link function** to map a linear model onto a non-linear outcome space.
- Such models are known as **generalized linear models (GLM)**.

Logistic regression

The logit function

- The **logit** function takes values in the range $[0, 1]$ and maps them to the range $[-\infty, \infty]$

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

Logistic regression

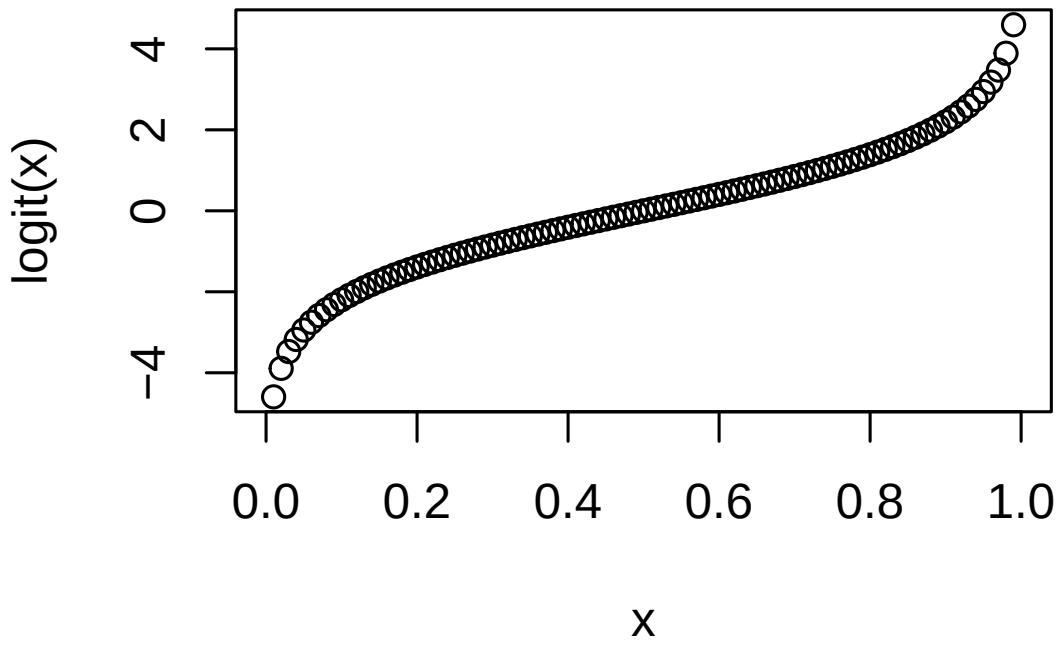
The logit function

```
logit <- function(x) {return(log(x/(1-x)))}
logit(c(0, 0.01, 0.5, 0.99, 1))
```

```
## [1]     -Inf -4.59512  0.00000  4.59512      Inf
```

Logistic regression

The logit function



Logistic regression

The inverse logit function

- The **inverse logit function** reverses this transformation, mapping values back to the $[0, 1]$ range:

$$logit^{-1}(x) = \frac{e^x}{1 + e^x} = \frac{1}{1 + e^{-x}}$$

Logistic regression

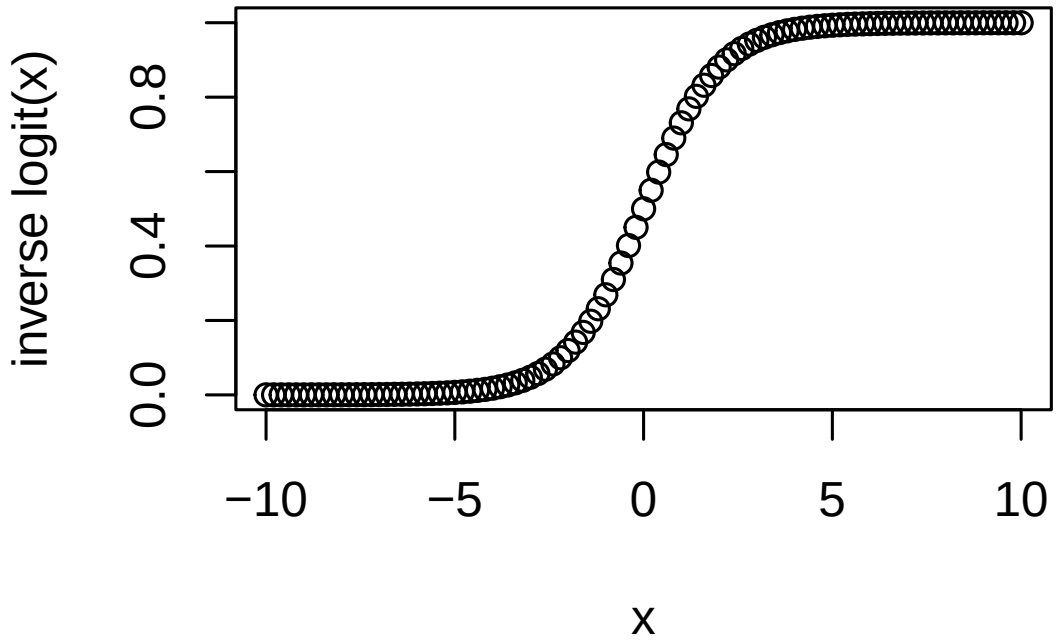
The inverse logit function

```
invlogit <- function(x) {return((exp(1)^x)/(1 + exp(1)^x))}
invlogit(c(-2, -1, 0, 1, 10))
```

```
## [1] 0.1192029 0.2689414 0.5000000 0.7310586 0.9999546
```

Logistic regression

The inverse logit function



Logistic regression

- We can write the following model for a binary outcome, where $p_i = P(y_i = 1|x_{1i}, x_{2i}, ..., x_{ki})$:

$$logit(p_i) = log(\frac{p_i}{1 - p_i}) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + ... + \beta_k x_{ki}$$

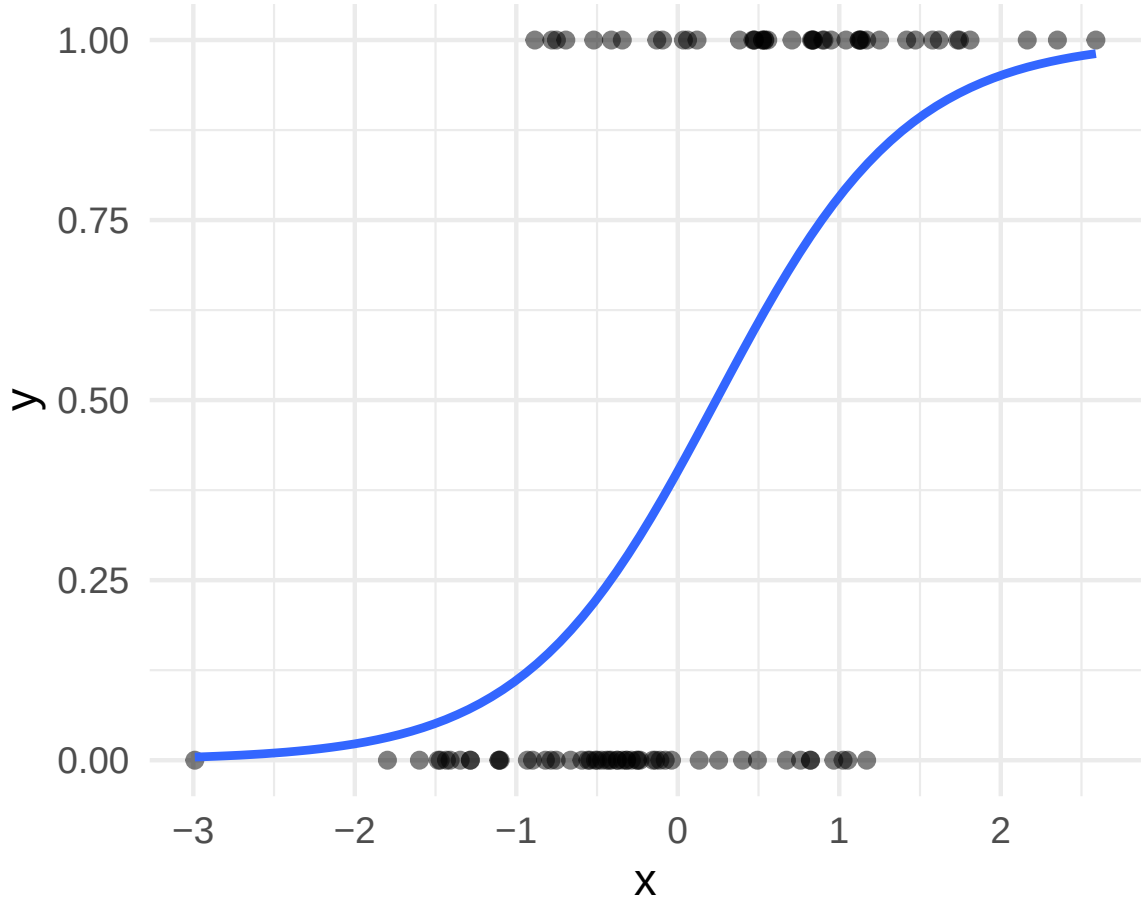
- The model can be expressed in terms of p_i using the inverse-logit function (also known as the logistic function):

$$p_i = logit^{-1}(\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + ... + \beta_k x_{ki})$$

$$p_i = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_{1i} + \beta_2 x_{1i} + ... + \beta_k x_{ki})}}$$

Logistic regression

Fitting a logistic curve to simulated data



Logistic regression

Error terms

- OLS regression has an error-term because we assume that the outcome is normally-distributed with two parameters, μ and σ^2, i.e. $y \sim N(\mu, \sigma^2)$.
- Binary variables follow a Bernoulli distribution,[4] characterized by a single parameter, p, the probability of a success. Thus, $y \sim Bernoulli(p)$.
- Therefore, there is *no error term in logistic regression*.

[4] Recall this is equivalent to $Binomial(1, p)$.

Logistic regression

Estimation

- There is no direct algebraic solution to obtain such estimates (unlike OLS regression, i.e. $(X^T X)^{-1} X^T y$).
- In frequentist statistics, the parameters in a logistic regression (and most other GLMs) are estimated using **Maximum Likelihood Estimation (MLE)**.
  - The MLE estimates have the same general properties as OLS estimates w.r.t standard errors, confidence intervals, and p-values.

Logistic regression

Maximum Likelihood Estimation

- For logistic regression, where β is a vector of coefficients and X is a matrix of predictors, the **likehood** is written as

$$P(y|\beta, X) = \prod_{i=1}^{n} (logit^{-1}(X_i\beta))^{y_i}(1 - logit^{-1}(X_i\beta))^{1-y_i}$$

- The goal of MLE is to find the β that maximizes this function.
  - It finds the parameter values most likely to have produced the observed data.
- A *iterative* algorithm is used to find the parameters that maximize the likelihood function, typically using the logarithm of the likelihood function for computational efficiency.
  - Unlike OLS, models can sometimes fail to converge on an appropriate solution. This can be an issue when trying to fit complex models.

Logistic regression

Estimation in R

We can easily estimate this using the `glm` function in R. The `family` argument is used to select the appropriate model.

```
logit.mle <- glm(shg ~ age + I(age^2) + caste +
                       as.factor(village),
                 data = data,
                 family = binomial(link = "logit"))
logit.mle$converged # Has model converged?
```

```
## [1] TRUE
```

```
logit.mle$iter # How many MLE iterations used?
```

```
## [1] 5
```

Logistic regression

Comparison with the LPM

	LPM	Logistic
(Intercept)	-0.702***	-7.066***
	(0.057)	(0.358)
Age	0.047***	0.300***
	(0.002)	(0.014)
Age^2	-0.001***	-0.004***
	(0.000)	(0.000)
Caste (Lower)	0.060***	0.315***
	(0.011)	(0.054)
Num.Obs.	9064	9064
Log.Lik.	-5430.095	-5115.536
F	24.515	17.330

Village fixed-effects omitted.

+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

Logistic regression

Bayesian estimation

- Logistic regression can alternatively be estimated using Bayesian methods.
  - We can use the same MCMC approach as used for OLS regression.
- Unlike MLE, Bayesian estimation does not converge on a single maximum likelihood estimate of β, but produces a posterior distribution.
- If a *uniform prior* is used, the posterior density is proportional to the likelihood function and the *mode* is equal to the maximum likelihood estimate.
  - Typically, we can do better by using more informative priors.

Logistic regression

Bayesian estimation in R

```
logit.bayes <- stan_glm(shg ~ age + I(age^2) + caste +
                            as.factor(village),
                        data = data,
                        family = binomial(link = "logit"),
                        refresh = 0, chains = 1, iter = 5000)
```

Note: Bayesian estimation can be considerably slower than MLE. The model requires more iterations than the default to ensure convergence, likely due to the nested structure of the data (more informative priors would likely help).

Logistic regression

Comparing Maximum Likelihood and Bayesian estimates[5]

	MLE	Bayes
(Intercept)	-7.066	-7.053
	[-7.782, -6.378]	[-7.755, -6.340]
Age	0.300	0.300
	[0.273, 0.328]	[0.274, 0.329]
Age^2	-0.004	-0.004
	[-0.004, -0.003]	[-0.004, -0.003]
Caste (Lower)	0.315	0.317
	[0.209, 0.422]	[0.209, 0.424]
Num.Obs.	9064	9064
Log.Lik.	-5115.536	
ELPD		-5192.6

Village fixed-effects omitted.

[5] For comparability, 95% confidence/credibility intervals are shown below coefficients.

Logistic regression

Interpretation: Terminology

- If an outcome occurs with probability p, the **odds** of the outcome are defined as $\frac{p}{1-p}$.
  - If $p = 0.5$, the odds $= \frac{0.5}{1-0.5} = 1$. If $p = \frac{2}{3}$, the odds ≈ 2.
- The **log odds** is the natural logarithm of the odds
  - This is also known as the **logit** function, $log(\frac{p}{1-p})$
- An **odds ratio** is the ratio of two odds.
  - $OR(p, q) = \frac{\frac{p}{1-p}}{\frac{q}{1-q}}$
- Odds ratios can be used as a way to communicate changes in the probability scale:

$$OR(0.6, 0.8) = \frac{\frac{0.6}{1-0.6}}{\frac{0.8}{1-0.8}} = \frac{1.5}{4} = 0.375$$

Logistic regression

Interpretation: Log-odds

- Since the outcome can be expressed as $log(\frac{p}{1-p})$, the coefficients in the regression output are **log odds**.
  - Where $\beta_{age} = 0.3$, a 1 year increase is associated with a 0.3 increase in the log-odds of SHG membership.
  - $\beta_{caste} = 0.315$, implying that belonging to a lower caste group versus a higher caste group changes the log-odds of SHG membership by 0.315.

Logistic regression

Interpretation: Odds-ratios

- We can get the **odds ratio** by exponentiating the coefficients
  - $OR(age) = exp(0.3) = 1.35$. A 1 year increase in age is associated with a 34% increase in the probability of SHG membership.
  - $OR(caste) = \frac{Odds(SHG=1|lower-caste)}{Odds(SHG=1|higher-caste)} = exp(0.315) = 1.37$. This implies that low caste residents are more likely to belong to SHGs (a 37% increase in the probability of SHG membership).

Logistic regression

Interpretation: Intuition

- Why do we get an odds ratio and not an odds when we exponentiate a log odds?

$$exp(\beta_{caste}) = exp(\beta^*_{caste_{low}} - \beta^*_{caste_{high}})$$

$$= exp(log(odds(p|caste_{low})) - log(odds(p|caste_{high})))$$

$$= \frac{exp(log(odds(p|caste_{low}))}{exp(log(odds(p|caste_{high})))}$$

$$= \frac{odds(p|caste_{low})}{odds(p|caste_{high})}$$

Logistic regression

Interpretation: The divide-by-4 rule

- The divide-by-4 rule provides a quick way to assess the effects of predictors in a logistic regression:
  - The logistic curve is steepest at the center, where $\beta_0 + \beta X = 0$ and $logit^{-1}(\beta_0 + \beta X) = 0.5$. The slope (or the derivative of the logistic function) is maximized.
  - At this point, $\frac{\beta e^0}{(1+e^0)^2} = \frac{\beta}{(1+1)^2} = \beta/4$.
- $\beta/4$ is the *maximum* difference in $P(y = 1)$ corresponding to a unit change in x.
  - This provides an simple approximation for the *maximum effect of a predictor*.
- For example, $\beta_{Age}/4 = 0.3/4 = 0.075$. Thus, the maximum effect of a 1-year results in a maximum 7.5% increase in the probability of SHG membership.

Logistic regression

Confidence intervals

- The standard formula for calculating confidence intervals assumes normality. This assumption is violated by logistic regression so standard (Wald) confidence intervals are incorrect.[6]
- Instead, confidence intervals for GLMs are calculated by using information from the likelihood function using the **profile likelihood** approach.
  - Note: Profile intervals are considerably slower to compute than Wald intervals.
- For Bayesian models, we can construct credible intervals using the posterior distribution in the same fashion as OLS models.

[6] In practice, the two approaches often produce very similar results, as the following example shows.

Logistic regression

Confidence intervals

The `conf.int` function in R allows us to calculate the correct confidence intervals (in this case for the effect of age). `conf.int` default provides standard confidence intervals. The intervals are exponentiated to get odds ratios.

```
round(exp(confint.default(logit.mle)[2,]),4)
```

```
##  2.5 % 97.5 %
## 1.3130 1.3873
```

```
round(exp(confint(logit.mle))[2,],4)
```

```
##  2.5 % 97.5 %
## 1.3133 1.3877
```

Logistic regression

Model fit: Log-likelihood

- The **log-likelihood** of a model is defined as

$$\sum_{i=1}^{n} log(p_i)y_i + log(1-p_i)(1-y_i)$$

- If $y_i = 1$, we add $log(p_i)$, otherwise we add $log(1-p_i)$.
- It is always negative.[7] A higher score indicates a better fit
  - But like R^2, adding more variables tends to increase the score.
- A related measure known as **deviance** is simply -2 times the log-likelihood.

[7] Recall $log_e(1) = 0$.

Logistic regression

Model fit: Log-likelihood

We can calculate the log-likelihood using the formula above or the `logLik` function:

```
y <- data$shg
pred_probs <- predict(logit.mle, type = "response")
sum(log(pred_probs)*y + log(1 - pred_probs)*(1-y))
```

```
## [1] -5115.536
```

```
print(logLik(logit.mle))
```

```
## 'log Lik.' -5115.536 (df=75)
```

Logistic regression

Model fit: Log-likelihood

	Village	Caste	Age	Full	+Religion
(Intercept)	-1.421	-1.470	-6.981	-7.066	-7.053
	(0.243)	(0.244)	(0.357)	(0.358)	(0.381)
Caste (Lower)		0.286		0.315	0.316
		(0.052)		(0.054)	(0.055)
Age			0.298	0.300	0.300
			(0.014)	(0.014)	(0.014)
Age^2			-0.004	-0.004	-0.004
			(0.000)	(0.000)	(0.000)
Hindu					-0.014
					(0.139)
Log.Lik.	-5431.917	-5417.034	-5132.504	-5115.536	-5115.531

Village fixed-effects omitted.

Logistic regression

Model fit: Log-likelihood (Bayesian)

We can use the same formula to calculate the log-likelihood for a Bayesian model.[8] Do you notice any problems with such an approach?

```
pred_probs.bayes <- predict(logit.bayes, type = "response")
sum(log(pred_probs.bayes)*y + log(1 - pred_probs.bayes)*(1-y))
```

```
## [1] -5115.654
```

[8] The standard 'logLik' function does not work for Bayesian models.

Logistic regression

Model fit: Log-likelihood (Bayesian)

To incorporate the uncertainty in the posterior distribution, we can take the average log-likelihood of each point over all posterior samples S, known as the **log-pointwise-predictive-density**.[9]

```
S <- dim(posterior_pred_probs)[1]
n <- dim(posterior_pred_probs)[2]
LLs <- matrix(nrow = n, ncol = S)
for (i in 1:S) {
    LLs[,i] <-
        posterior_pred_probs[i,]*y +
        (1 - posterior_pred_probs[i,])*(1-y)
}
sum(log(rowSums(LLs)/S))
```

```
## [1] -5115.464
```

[9] See McElreath p. 210

Logistic regression

Model fit: Log-likelihood (Bayesian)

- In this case, the estimates using `predict` and `posterior_predict` are almost identical.
- However, you should always use the full posterior distribution when computing any summary statistics from Bayesian models.

Logistic regression

Model fit: R^2 and fraction correctly predicted

- R^2 is no longer a useful measure of fit for the same reason as the LPM.
- A simple fit measure is the *fraction of cases correctly predicted*, where $\hat{y}_i = 1$ if $p_i > 0.5$ and $\hat{y}_i = 0$ if $p_i \leq 0.5$, but this approach throws out information about the predicted probabilities.

Logistic regression

Model fit: Pseudo-R^2

- There are several different approaches to construct **pseudo-R^2** statistics. These measures approximate an R^2 by ranging between 0 and 1, but do are not equivalent.[10]
- One of the more common variants is McFadden's pseudo-R^2:

$$R^2 = 1 - \frac{LL(M_{full})}{LL(M_{intercept})}$$

- This is the standard formula for R^2, where the log-likelihood of an intercept-only model as the total sum of squares and the fully parameterized model is the sum of squared errors.

[10] See this blog post for a discussion of several different measures.

Logistic regression

Model fit: McFadden's pseudo-R^2

The result is similar to the R^2 obtained from the LPM.

```
logit.mle.i <- glm(shg ~ 1,
                   data = data,
                   family = binomial(link = "logit"))

pR2 <- 1 - (logLik(logit.mle)[1]/logLik(logit.mle.i)[1])
print(round(pR2,3))
```

```
## [1] 0.144
```

```
print(round(summary(lpm)$r.squared, 3))
```

```
## [1] 0.168
```

Logistic regression

Model fit: Bayesian held-out likelihood

- For Bayesian models, we can use the `loo` function to calculate a held-out likelihood score using the entire posterior distribution.
  - The `elpd_loo` score provides an approximation of the LOO-CV expected log-pointwise predictive density.
- This will be slightly more conservative (lower) than the in-sample log-likelihood.

Logistic regression

Model fit: Bayesian held-out likelihood

```
print(loo(logit.bayes))
```

```
##
## Computed from 2500 by 9064 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -5192.6 43.8
## p_loo        77.1  1.0
## looic     10385.2 87.7
## ------
## Monte Carlo SE of elpd_loo is 0.2.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

Probit regression

- The **probit** regression model is similar to logistic regression but uses a cumulative normal function Φ instead of the inverse logistic function:

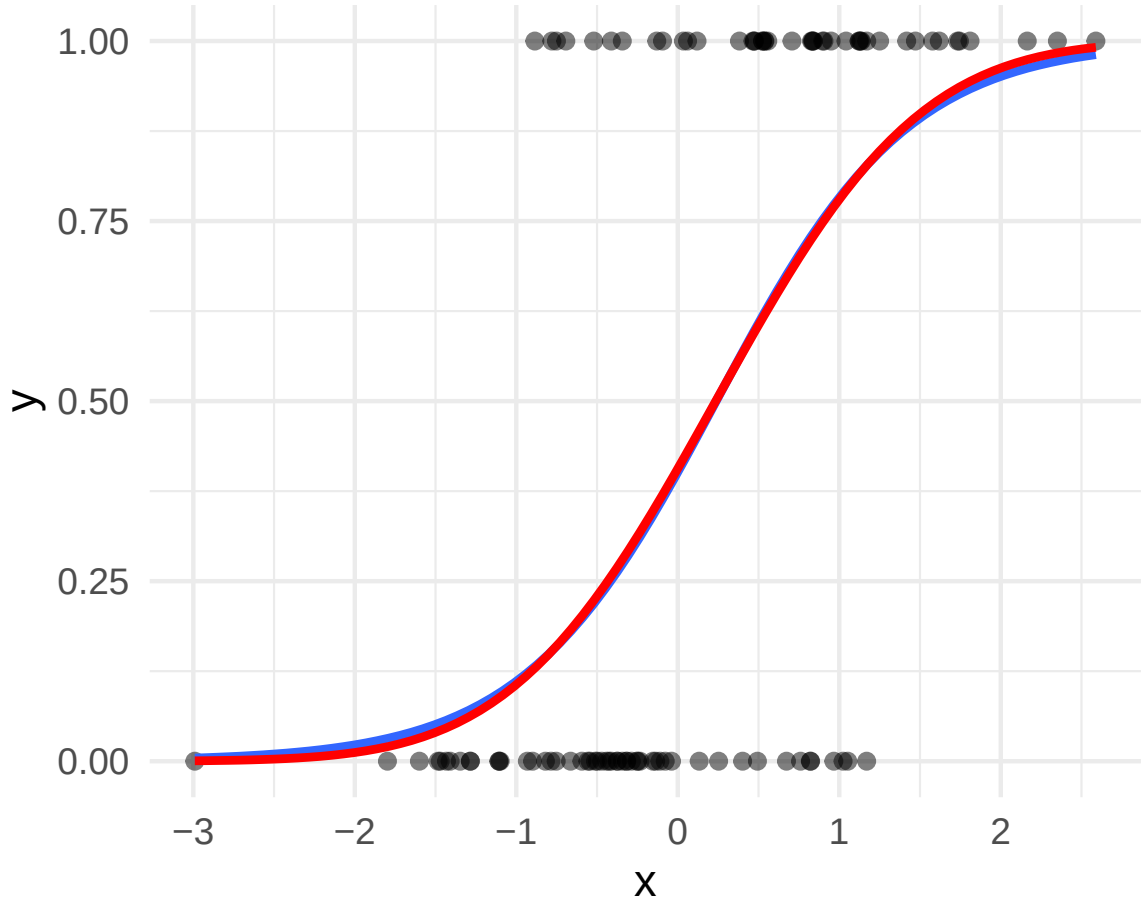
$$P(y = 1|X) = \Phi(X\beta)$$

- Historically, probit has been preferred in some cases for computational reasons, but the two models tend to produce similar results.[11]
- Like logistic regression, probit regression can be estimated using MLE or Bayesian approaches.

[11] See this blog post for an example of a comparison using Monte Carlo simulation.

Probit regression

Probit (red) and logistic (blue) curves



Probit regression

Estimation

The only change to the model is the link function.

```
probit.mle <- glm(shg ~ age + I(age^2) + caste +
                      as.factor(village),
                  data = data,
                  family = binomial(link = "probit"))
```

Probit regression

Comparison with logistic regression

	Logistic	Probit
(Intercept)	-7.066***	-4.064***
	(0.358)	(0.202)
Age	0.300***	0.169***
	(0.014)	(0.008)
Age^2	-0.004***	-0.002***
	(0.000)	(0.000)
Caste (Lower)	0.315***	0.190***
	(0.054)	(0.032)
Num.Obs.	9064	9064
Log.Lik.	-5115.536	-5123.682
F	17.330	19.151

Village fixed-effects omitted.

+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

Probit regression

Interpretation

- The coefficients in a probit regression are more difficult to interpret than OLS or logistic regression.
- In general, positive coefficients indicate increases in the predicted probability of the outcome, while negative coefficients indicate decreases.[12]

[12] See the Stata blog for further discussion.

Summary

- LPM
  - Estimate using OLS
  - Easy to estimate and interpret, but can make bad predictions
- Logistic regression
  - Logistic function used to apply linear model to non-linear outcome
  - Interpret log-odds and odds-ratios
  - Estimate using MLE or Bayes
- Probit regression
  - Cumulative normal distribution as link
  - Similar fit to logistic but more difficult to interpret

Next week

- Logistic regression continued
  - Predictions and marginal effects
  - Interaction terms