UNIVERSITY
OF MANNHEIM
School of Social Sciences

Advanced Quantitative Methods in Political Science: Models for Binary Dependent Variables

Thomas Gschwend | Oliver Rittmann | Viktoriia Semenova

Week 6 - 23 March 2022

Leftovers from last week: Implementation in R

Infrastructure of "Advanced Quantitative Methods" Course

Three steps to come up with a suitable ML Estimator for your research question

1. Formulate a suitable probability model of the data-generating process including assumptions of how Y is distributed (i.e., stochastic component) and a parametrization of stuff that gets estimated (i.e., systematic component).
2. Write down the (log-)likelihood function based on your parametrization and assumptions.
3. Maximize the Log-Likelihood, analytically (often hard, even impossible) or numerically (use functions in R).

There are two more things we need to talk about this semester:

- Interpretation of estimation results through simulating quantities of interest (*you have seen this last semester as well as in the lab*)
- How to check whether the assumed model does fit the data? (*Coming soon!*)

Then, we can apply this infrastructure to any existing model or come-up with our own model.

Implementation in R

- Let's estimate a linear regression model via maximum likelihood instead of using ordinary least squares
- *Step 1:* Assume the following model:
  $$\begin{aligned} Y_i &\sim f_N(y_i|\mu_i, \sigma^2) && \text{stochastic} \\ \mu_i &= X\beta \; (= \beta_0 + \beta_1 x_i) && \text{systematic} \end{aligned}$$
- The parameters we are going to estimate using the above parameterization are $\theta = (\mu_i, \sigma^2) = (\beta_0, \beta_1, \sigma^2)$
- We further assume that y_i is iid.

Implementation in R

- *Step 2:* Using our assumptions about the model and the chosen parameterization of the systematic component, we can set up the likelihood function as follows:

$$L(\beta,\sigma^2|y) = (2\pi\sigma^2)^{-N/2}\ exp\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \beta_0 - \beta_1 x_i)^2\right]$$

- Then (although this is optional) we can take the log of the likelihood function, because it simplifies the next step (i.e. maximization):

$$\begin{aligned}
logL(\beta,\sigma^2|y) &= -\frac{N}{2}log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i-\beta_0-\beta_1x_i)^2 \\
&= -\frac{N}{2}log(2\pi) - \frac{N}{2}log(\sigma^2) - \frac{1}{2}\sum_{i=1}^{N}\frac{(y_i-\beta_0-\beta_1x_i)^2}{\sigma^2} \\
&= -\frac{N}{2}log(\sigma^2) - \frac{1}{2}\sum_{i=1}^{N}\frac{(y_i-\beta_0-\beta_1x_i)^2}{\sigma^2}
\end{aligned}$$

Implementation in R

- Now, let’s write the log-likelihood as a R-function `lm.lik`:

```
 lm.lik <- function(theta, y, x) {
  beta0 <-  theta[1]
  beta1 <-  theta[2]
  gamma <-  theta[3]
# Parametrize sigma2 to be non-negative
  sigma2 <- exp(gamma)
# Residual
  e <- y - beta0 - beta1*x
# Log lik function for one observation
  logl <- -1/2*log(sigma2) - 1/2*(e^2/(sigma2))
# Log lik function is sum over N observations
  logl <- sum(logl)
  return(logl)
 }
```

Implementation in R

- Here is a slightly more general code of the same likelihood:

```
lm.lik1 <-function(theta,y,X){
     N<-nrow(X) # number of observations
     k<-ncol(X) # number of parameters
 # Supstring paramters theta
     beta<-theta[1:k]
     gamma<-theta[k+1]
 # Parametrize sigma2 to be non-negative
     sigma2 <- exp(gamma)
 # Residual
     e<- y-(X%*%beta)
 # Log lik function fover N observations
     logl <- - 1/2*N*log(sigma2)-1/2*((t(e)%*%e)/(sigma2))
 return(logl)
   }
```

Implementation in R

- *Step 3*: Maximize the log-likelihood numerically. Of course, we could do it analytically (see last week). Now we let the computer do all the work for us.
- R provides a tool named `optim()` which maximizes arbitrary functions numerically if we specify `control=list(fnscale=-1)` (`optim()` tries to minimize by default).
- To maximize our likelihood function, we need to feed `optim()` with a set of starting values (the `optim(stval, ...)`'s first guesses for the parameters).

```
          stval <- c(1,1,1)
```

- Then we simply call `optim()` to maximize a likelihood function (`fn=lm.lik`), with particular starting values (`stval`) and data (`y=y, x=x`)

```
res <- optim(stval, fn=lm.lik, control=list(fnscale=-1),
             y=y, x=x, hessian=TRUE)
> res$par
[1] 49.708304  1.125821 10.378797
> sqrt(diag(solve(-1 * res$hessian)))
[1] 1.6249732 0.4578586 3.7924240
```

- Take some data and see how our $\hat{\theta}_{ML}$ compares to $\hat{\theta}_{OLS}$!

Heteroskedastic Regression

Heteroskedastic Regression

- Now, what if we instead relax the homoskedasticity assumption?
- *Step 1:* Assume the following model:

$$\begin{array}{lcll} Y_i & \sim & f_N(y_i|\mu_i,\sigma_i^2) & \text{stochastic} \\ \mu_i & = & X\beta\ (=\beta_0+\beta_1 x_i) & \text{systematic} \\ \sigma_i^2 & = & exp(\gamma Z)\ \ (=exp(\gamma_0+\gamma_1 z_i)) & \text{systematic} \end{array}$$

- The parameters we are going to estimate using the above parametrization of the model's systematic component are $\theta = (\beta_0, \beta_1, \gamma_0, \gamma_1)$
- We further assume that the y_i are independently distributed.
- Thus, we get the following log-likelihood function:

$$\begin{aligned} logL(\theta|y) &= -\frac{N}{2}log(2\pi) - \frac{1}{2}\sum_{i=1}^{N} log(\sigma_i^2)\ -\frac{1}{2}\sum_{i=1}^{N}\frac{(y_i-\beta_0-\beta_1 x_i)^2}{\sigma_i^2} \\ &= \qquad -\frac{1}{2}\sum_{i=1}^{N}(\gamma_0+\gamma_1 z_i)\ -\frac{1}{2}\sum_{i=1}^{N}\frac{(y_i-\beta_0-\beta_1 x_i)^2}{exp(\gamma_0+\gamma_1 z_i)} \end{aligned}$$

Heteroskedastic Regression - Implementation in R

- Lets write the LL as a R-function `hetero.lik`, but this time with four arguments (θ, y, x, z):

```
hetero.lik <- function(theta, y, x, z) {
    beta0 <-  theta[1]
    beta1 <-  theta[2]
    gamma0 <- theta[3]                # This line is new
    gamma1 <- theta[4]                # This line is new
# Residual
    e <- y - beta0 - beta1*x
# Variance parameterization
    sigma2 = exp(gamma0 + gamma1*z)  # This line is new
# Log lik function for one observation
    logl <- -1/2*log(sigma2) - 1/2*(e^2/(sigma2))
# Log lik function is sum over N observations
    logl <- sum(logl)
    return(logl)
  }
```

- Note, we need to feed `optim()` with four starting values!

Heteroskedastic Regression - Implementation in R

```
  # start values for maximization algorithm  - now we need 4 values
  stval <- c(0,0,0,0)

  # maximize the likelihood function numerically using optim()

res2 <- optim(stval,                # starting values
     fn=hetero.lik,                 # the likelihood function
     control=list(fnscale=-1),      # maximize rather than minimize funct
     y=y, x=x, z=z,                 # the data
     hessian=TRUE)                  # return numerical Hessian

cat("MLE Betas\n", res2$par[1:2], "\n\n")
cat("MLE Gammas\n", (res2$par[3:4]), "\n\n")
cat("Hessian\n")
print(res2$hessian)
cat("\n MLE St. Errors\n", sqrt(diag(solve(-1*res2$hessian))), "\n\n")
```

Intro

What should you take home from this class today?

- You will see three equivalent justifications for logit and probit models.
- Buckle up! We expand our toolbox. You will learn many more models for binary dependent variables (through a different link function). Thus, same stochastic but different systematic component.
- We will learn some general strategies to check whether the assumed model actually fits the data.

Models for Binary Dependent Variables

Binary Response Models

There are many social outcomes that are binary, e.g.

- A war is fought or not
- A coalition dissolves or not
- A respondent reports to vote or not
- A MP votes in favor of a proposal or not

What would happen if we run OLS in such a situation (aka *linear probability model*)?

The Linear Probability Model

Given that Y_i is Bernoulli, we get (remember?)

$$E(Y_i) = 1 \cdot Pr(Y_i = 1) + 0 \cdot Pr(Y_i = 0) = Pr(Y_i = 1)$$

Thus,

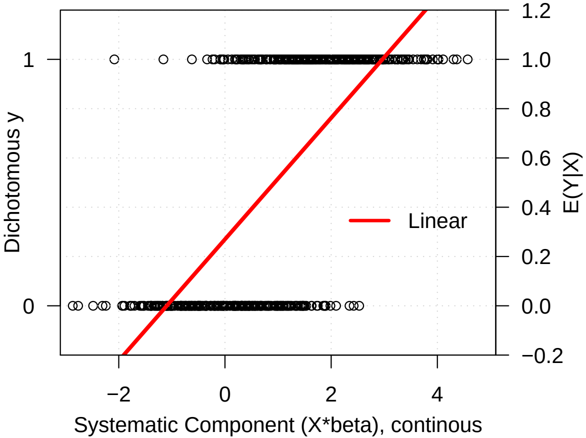
$$E(Y_i) = Pr(Y_i = 1) = \pi_i = X_i\beta = linear(X_i)$$

and (remember?)

$$Var(Y_i) = E(Y_i)(1 - E(Y_i)) = \pi_i \cdot (1 - \pi_i) = X_i\beta \cdot (1 - X_i\beta)$$

- This amounts to fitting an OLS regression ...
  ...with unbiased point estimates $\hat{\beta}$
- The variance, however, varies systematically with X_i (heteroskedasticity).
- Errors can only take two values, $1 - X_i\beta$ or $-X_i\beta$
- Inference from OLS is therefore invalid (non-normal, heteroskedastic errors).

Fitting Binary Data with a Linear Probability Model



Model also yields *out-of-bounds predictions*. Thus we need to transform the systematic component and find a (link) function $g(\cdot)$, such that $0 \leq g(X_i\beta) \leq 1$.

Derivation of Logit and Probit Models

There are three different ways to formulate Logit and Probit Models:

1. Pure Probability Approach
2. Latent Variable Approach
3. Random Utility Approach

All three justifications will lead to the same models.

Pure Probability Approach

1. Pure Probability Approach

- Recall that the Bernoulli would be appropriate if every event had the same chance π chance of occurring.
- Too restrictive for many substantive applications

1. Stochastic Component:

$$Y_i \sim Y_{Bern}(y_i|\pi_i) = \pi_i^{y_i}(1-\pi_i)^{1-y_i} = \begin{cases} \pi_i & for \quad y_i = 1 \\ 1-\pi_i & for \quad y_i = 0 \end{cases}$$

2. Systematic Component:
   The model would not be identified if every observation has its own π_i. Thus, we reduce the number of parameters and allow for substantive explanatory variables through the following parameterization, using a function $g(\cdot)$:

$$E(Y_i) = Pr(Y_i = 1) = \pi_i = g(X_i\beta)$$

3. Y_i and Y_j are independent, conditional on X

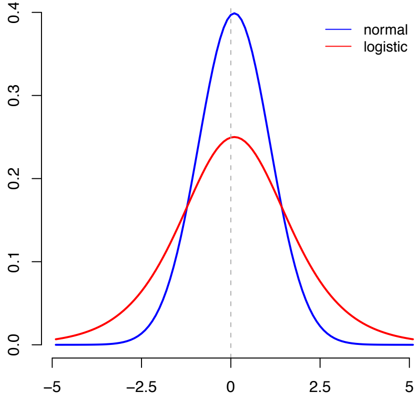
Which link function $g(\cdot)$ should we choose?

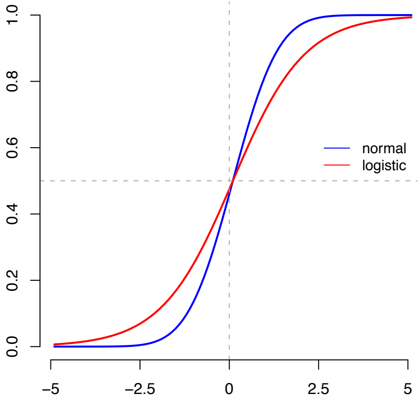
We have seen last semester that …

- Using the *cumulative standard logistic*, we get the *Logit Model*.
- Using the *cumulative standard normal*, we get the *Probit Model*.
- In practice, both model specifications lead to the same results, because the standard normal and logistic distribution are rather similar …

Logistic and standard normal distribution

- The logistic distribution has fatter tails (corresponding to a variance of $\pi^2/3$)
- Logit and probit coefficients differ by a factor of ca. 1.81 ($\pi/\sqrt{3}$). But both models produce the same quantities of interest.





Logit Model

- Taking for $g(\cdot)$ the cumulative standard logistic function $\Lambda(\cdot)$ yields

$$Pr(Y_i = 1) = \pi_i = \Lambda(X_i\beta) = \frac{e^{X_i\beta}}{1 + e^{X_i\beta}} = \frac{1}{1 + e^{-X_i\beta}}$$

- The log-likelihood contribution $L_i(\pi|y)$ of observation i is

$$lnL_i(\pi|y) = y_i \cdot ln(\pi_i) + (1 - y_i) \cdot ln(1 - \pi_i)$$

- Then summing-up all n individual contributions assuming independent realizations

$$lnL(\pi|y) = \sum_{i=1}^{n} (y_i \cdot ln(\pi_i) + (1 - y_i) \cdot ln(1 - \pi_i))$$

- Using our parameterization of π_i the corresponding *log-likelihood function of the Logit model* becomes

$$lnL(\beta|y) = \sum_{i=1}^{n} (y_i \cdot ln(\frac{1}{1 + e^{-X_i\beta}}) + (1 - y_i) \cdot ln(1 - \frac{1}{1 + e^{-X_i\beta}}))$$

Probit Model

- Another choice for $g(\cdot)$ is the standard normal distribution

$$Pr(Y_i = 1) = \pi_i = \int_{-\infty}^{X_i\beta} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}Z^2} dZ = \Phi(X_i\beta)$$

- The above integral does not have a closed form solution and, therefore, gets evaluated numerically and is typically abbreviated as $\Phi(X_i\beta)$.
- The log-likelihood contribution $L_i(\pi|y)$ of observation i is still (as before!)

$$lnL_i(\pi|y) = y_i \cdot ln(\pi_i) + (1 - y_i) \cdot ln(1 - \pi_i)$$

- Summing-up (assuming independent realizations) and using the above parameterization of π_i, we get

$$lnL(\beta|y) = \sum \left(y_i \cdot ln(\Phi(X_i\beta)) + (1 - y_i) \cdot ln(1 - \Phi(X_i\beta))\right)$$

- With $1 - \Phi(X_i\beta) = \Phi(-X_i\beta)$ because of the symmetry, the corresponding *log-likelihood function of the Probit model* becomes

$$lnL(\beta|y) = \sum \left(y_i \cdot ln(\Phi(X_i\beta)) + (1 - y_i) \cdot ln(\Phi(-X_i\beta))\right)$$

Latent Variable Approach

2. Latent Variable Approach

- Let Y^* be a continuous unobserved variable (e.g., health, propensity to vote, ect. Also used to formulate *item-response models*)
- Define a model through its stochastic and systematic component

$$\begin{aligned} Y_i^* &\sim P(y_i^*|\mu_i) \\ \mu_i &= X_i\beta \end{aligned}$$

  with an *observation mechanism*:

$$y_i = \begin{cases} 1 & y^* \geq \tau \\ 0 & y^* < \tau \end{cases}$$

  Given that Y^* is unobserved anyway we set $\tau = 0$.
- Finally, lets assume independent realizations.

Question: What model do we get if we observe y_i^* and $P(\cdot)$ is normal?

What model do we get if $P(\cdot)$ is normal?

Let the following latent regression model be defined as

$$y_i^* = X_i\beta + \epsilon$$

where we assume that ϵ has mean 0 and fixed (not estimated!) homoskedastic variance ...

- ...$\pi^2/3$ if we assume a *standard logistic* distribution
- ...1 if we assume a *standard normal* distribution

Brief Aside on Assumptions

1. Fixed variance of ϵ.
   - Suppose we assume a different variance. Say the variance of ϵ is scaled by an unrestricted parameter σ. Then, the latent regression model would become

$$\begin{aligned} y_i^* &= X_i\beta + \sigma\epsilon \\ \frac{y_i^*}{\sigma} &= X_i\frac{\beta}{\sigma} + \epsilon \end{aligned}$$

   - This is still the same model and the same data (just rescaled, different threshold τ).
2. Fixed threshold $\tau = 0$.
   - What if $\tau \neq 0$? Then, letting α be a unknown constant term (and $\tilde{X}_i$ is X_i without a column of 1s) we get

$$Pr(y_i^* > \tau) = Pr(\alpha + \tilde{X}_i\beta + \epsilon > \tau) = Pr((\alpha - \tau) + \tilde{X}_i\beta + \epsilon > 0)$$

   Since $(\alpha - \tau)$ is unknown, setting arbitrarily $\tau = 0$ will just affect the size of the constant term.

Derivation using the Latent Variable Approach

- Given the model assumptions, we have

$$\begin{aligned} Pr(y_i = 1) &= Pr(y_i^* > 0) \\ &= Pr(X_i\beta + \epsilon > 0) \\ &= Pr(\epsilon > -X_i\beta) \\ &= 1 - Pr(\epsilon < -X_i\beta) \\ &= 1 - F(-X_i\beta) \end{aligned}$$

  where F is the cumulative distribution of ϵ.

- If F is symmetric about 0 (as it is with logistic or normal), we get

$$Pr(y_i = 1) = 1 - F(-X_i\beta) = F(X_i\beta)$$

- Now, choosing for $F(\cdot)$ a ...

  ...cumulative standard logistic yields a logit model, $Pr(y_i = 1) = \Lambda(X_i\beta)$.

  ...cumulative standard normal yields a probit model, $Pr(y_i = 1) = \Phi(X_i\beta)$.

Random Utility Approach

3. Random Utility Approach

- Let U_{ij} be the utility of individual i derived when choosing alternative j.
- Assume that U_{ij_0} and U_{ij_1} are independent and let

$$U_{ij} \quad \sim \quad P(U_{ij}|\mu_{ij})$$

- Let $Y^* = U_{ij_1} - U_{ij_0}$ be a difference of utilities with an *observation mechanism*:

$$y_i = \begin{cases} j_0 & y^* \leq 0 \\ j_1 & y^* > 0 \end{cases}$$

- Note, this is equivalent to what we got with the latent variable approach.
- Thus, if $P(\cdot)$ is assumed to be distributed ...

  ...extreme value (aka Gumpel), then the difference Y^* is standardized logistic and we get a logit model.

  ...standardized normal, then the difference Y^* is standardized normal as well and we get a probit model.

Other Models for Binary Data

How to generate other Models for Binary Data? Same Stochastic but different Systematic Component

- An alternative to the logit and probit CDF's consider the *complementary log-log model (cloglog)*

  $$Pr(y_i = 1) = \pi_i = 1 - exp(-exp(X_i\beta))$$

  or, alternatively:

  $$ln(-ln(1 - Pr(y_i = 1))) = X_i\beta$$

- Another alternative is the *log-log model* (without the "complementary" "-1" part)

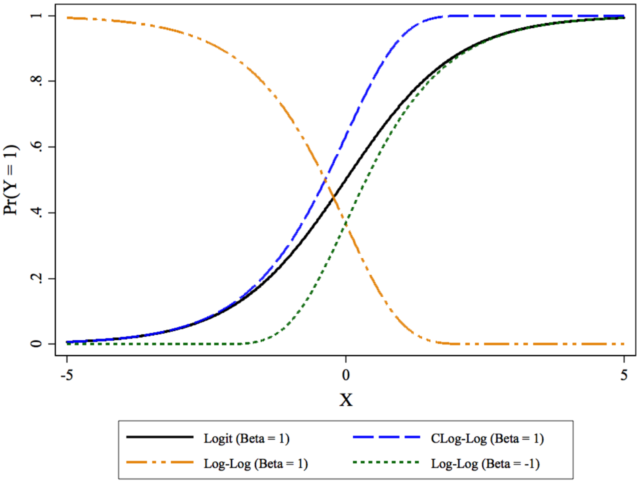
  $$Pr(y_i = 1) = \pi_i = exp(-exp(X_i\beta))$$

  or,

  $$ln(-ln(Pr(y_i = 1))) = X_i\beta$$

- Such models are used to predict duration of events (war, time to respond, ect).
- Key difference: models are not symmetrical (around 0.5).
- But why assuming that observations with a probability of .5 of choosing either of two alternatives are most sensitive to changes in independent variables?

Other Models for Binary Data



Taken from MLE handout of ???????

Scobit Model

- Taking the cumulative standard logistic function to get the logit model

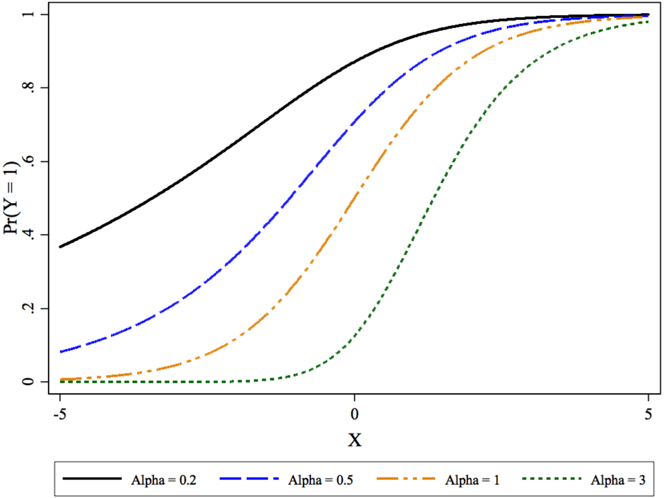
$$Pr(Y_i = 1) = \frac{1}{1 + e^{-X_i\beta}}$$

- One could generalize systematic component to get a more flexible functional form

$$Pr(Y_i = 1) = \frac{1}{(1 + e^{-X_i\beta})^{\alpha}}$$

Scobit stands for “skewed logit” and is invented by a political scientist (Nagler 1994)

Scobit CDFs with $\beta = 1$ and Varying α



Taken from MLE handout of ???????

Wanna have more? How about Neural Network Models?

- Goal: make relationship between π and X very flexible (almost “non-parametric”).
- For the logit model we have:

$$Pr(Y_i = 1) = \pi_i = \frac{1}{1 + e^{-X_i\beta}} = logit(X_i\beta) = logit(linear(X_i))$$

- The simplest neural network model is a straight generalization of this:

$$Pr(Y_i = 1) = \pi_i = logit(linear(logit(linear(X_i))))$$

- We can calculate QoI from this as we have done all along (same machinery).
- For an application in PoliSci, see Beck, Nathaniel, Gary King, and Langche Zeng. 2000. “Improving Quantitative Studies of International Conflict: A Conjecture”. *American Political Science Review* 94(1): 21–35.
- No one keeps you from using other stochastic components than Bernoulli to model a different DGP!

Model Fit

How to check whether the assumed model does fit the data?

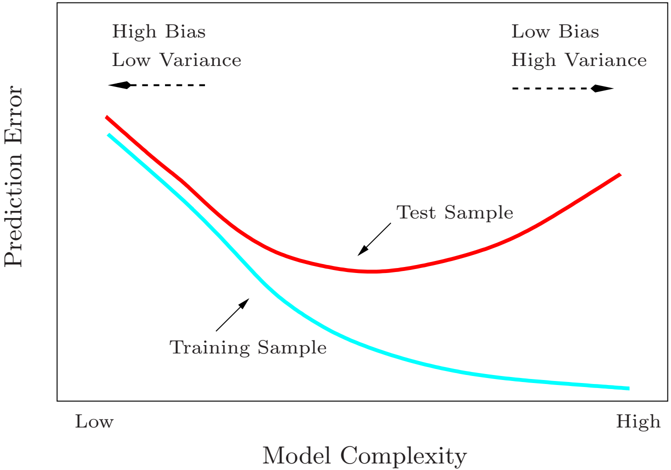
- There are many different tools to check whether the assumed model does fit your data.
- We may also find that some models do fit better than other models
- Important to evaluate the assumptions we have been making all along in setting-up a model and deriving a log-likelihood function.

- Bottom line: Do make an effort to check whether the assumed model does fit your data!

How to know which model is better?: Out-Of-Sample Forecasts

- Key requirement: Find the *systematic* rather than idiosyncratic features of any one data set (although you only have one draw, i.e., one data set).
- Set aside some (random) parts of the data (aka as *test data*) and fit your model to the rest (aka *training data*)
- Make predictions with training data and compare to the test data.
    - Compare average predictions and also full distribution
    - Say, for a given scenario you predict $Pr(y = 1) = 0.2$ using the *training* data, then 20% of such observations should actually be observed as $y = 1$ in the *test* data.
- Gold standard is test data that is really out-of-sample, i.e. not yet available.
- Of course, you need to assume that test and training data generated from the very same data-generating process. Thus, if the DGP changes between the time training and test data are observed... tough. Even a good model will fail.
- Still, out-of-sample forecast is the right test for any model.

Can We Stop Ourselves? On the Danger of Over-fitting



Taken from: Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2008. *The Elements of Statistical Learning* (2nd edition). Chapter 2: Fig 2.11, p. 38.

Fit Measures for Binary Variable Predictions

- Classify correctly predicted observations (for chosen cut-point at .5)
  - Using $\hat{\beta}$ from your model, generate predicted probabilities $\hat{\pi}_i$.
  - Generate variable of predicted values $\hat{y}_i = 1$ if $\hat{\pi}_i \geq 0.5$, 0 otherwise.
- Generate 2x2 classification table (aka *confusion matrix*).

	Predicted ($\hat{y}_i$)	
Observed (y_i)	0	1
0	n_{00}	n_{01}
1	n_{10}	n_{11}

- From this, we can construct *Percent Correctly Predicted* (PCP):

$$PCP = \frac{n_{00} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}}$$

- If, say, the DV is distributed 70 : 30, then a model (to beat) without independent variables would predict 70% of the cases correctly.
- Problems: (a) Uncertainty? (b) Precision: $\hat{\pi}_i = .51$ and $\hat{\pi}_j = .99$ are counted equally

Other Fit Measures for Binary Variable Predictions

- *Percent Reduction in Error* (PRE)
  - Classify correctly predicted observations relative to a baseline
  - Baseline is the Percent of observations in the Modal Category (PMC) of the dependent variable.

$$PRE = \frac{PCP - PMC}{1 - PMC}$$

  - *PRE* is just a function of *PCP*, thus, still the precision problems.
- *expected Percent Correctly Predicted* (ePCP)
  - Expected percentage of correct model predictions (Herron 1999 - PA article)

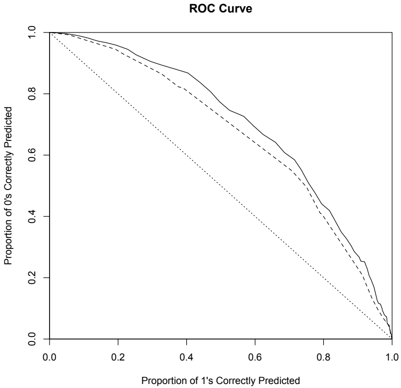
$$ePCP = \frac{1}{N}\left(\sum_{y_i=1}\hat{\pi}_i + \sum_{y_i=0}(1-\hat{\pi}_i)\right)$$

- All such classification-based measures focus on a model's ability to classify observations. No specification test, though (see Esarey and Pierce 2012)!
  - Thus, a good model fit (e.g., high PCP) *does not* imply a correct model specification.

Model Selection using ROC

- *Problem:* Classifications require a normative decision.
  - Let C be the number of times it is more costly classifying a 1 than a 0.
  - C must be chosen independently of the data; from review of literature, (survey of) policy makers
  - $C = 1$ often chosen, but without justification
- *Decision Theory:* Choose $Y = 1$ when $\hat{\pi} > 1/(1 + C)$ and 0 otherwise.
  - If $C = 1$, predict $y = 1$ when $\hat{\pi} > 0.5$ (as for PCP, PRE, ePCP)
  - If $C = 2$, predict $y = 1$ when $\hat{\pi} > 1/3$
  - Increasing C reduces chances of type I error ("false alarm")
  - If $C \to 0$ then $\hat{\pi} \to 1$, and if $C \to \infty$ then $\hat{\pi} \to 0$
- Only with chosen C it makes sense to compute (a) % of 1s and 0s correctly predicted, and (b) error patterns in different subsets of the data (or forecast)
- If you cannot justify *a priori* a value for C, use all of them! Plot ROC (receiver-operator characteristics) curves

ROC Curves



Taken from the `demo(roc)` in the `library(Zelig)` (see `help.zelig(logit)`)

- Compute % 1s and % 0s correctly predicted for every possible value of *C*.
- Plot % 1s by % 0s
- Overlay curve for several model specifications on the same graph.
- Normative decision about *C* does not matter if one curve is above another. We then say that one model *dominates* the other.
- Otherwise, one model (specification) is better than another in specified ranges of *C*.
- In R use e.g, `library(Zelig)` or `library(epicalc)`

Further Model Fit, Specification and Robustness Checks

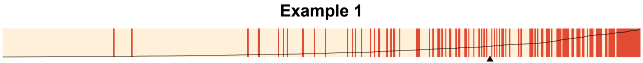
- *Cross-Validation* (for all types of models)
  - Randomly divide the data set into K equally sized folds (each fold will contain about $\frac{N}{K}$ observations)
  - Train model K-times on all but the k-th fold ($k \in \{1, \ldots, K\}$), then use k-th fold to estimate model on unseen data. Average across the K results.
  - Useful for smaller data sets where one cannot set aside test data
  - What does "average results" imply? Point estimates are the mean of the estimated point estimates of the subsets.
  - Standard errors should account for *within* as well as *across* variance (see King et al. 2001. "Analyzing Incomplete Political Science Data: An Alternative Algorithm for Multiple Imputation". *American Political Science Review* 95: 49 – 69, equation (3))
- *Repeated random sub-sampling validation* (unobs. heterogeneity)
  - Sample 2/3 of data, run model and collect results. Repeat several (about $m = 20$) times for different samples and combine results per King et al 2001 (aka "Rubin Rule", see above).
- Confront (all) *observable implications* with your observations.

Likelihood-Based Approaches

- Evaluation of model fit through any test statistic that is based on a transformation of the log-likelihood will be a *relative* measure of model fit (e.g., LRT)
- Akaike Information Criterion: $AIC = -2 \cdot lnL + 2p$
  - where p is the number of parameters in the statistical model, and L is maximum of the likelihood function for given model.
  - Pick the model among the possible ones with minimum AIC value. There is no statistical test of difference in AIC.
  - The penalty term ($2p$) does discourage overfitting while rewarding goodness of fit (because of LL).
- Bayesian Information Criterion: $BIC = -2 \cdot lnL + p \cdot ln(N)$
  - where N is the number of observations.
  - Larger penalty term ($p \cdot ln(N)$).
- AIC and BIC work even for non-nested models. Further examples are Vuong test, Bayes factors,....

Assessing Model Fit graphically - Separation Plot

Brian Greenhill, Michael D. Ward, Audrey Sacks. 2011. “The Separation Plot: A New Visual Method for Evaluating the Fit of Binary Models” *American Journal of Political Science*, 55(4): 991-1002.



- Graph fitted values with different colors for each observed outcome.
- Line indicates the predicted values of the observations
- Helpful for identifying clusters of false negatives and false positives (systematic or coding errors)
- Can be used for models with more than two categorical outcomes!
- In R use e.g, `library(separationplot)`