Statistical Modeling: A Tools Approach

Carlisle Rainey

2022-09-08

Week 1: Maximum Likelihood

Class agenda

Goal of the class Make you competent users and consumers (applied and methods papers) of methods beyond least-squares. I'm deliberately avoiding causal-inference methods (matching, DID, etc) because we have a class that covers those specifically that we're offering regularly. I want you to learn a lot about specific tools, but also develop the skills to go and learn more on your own.

We can deviate into any particular topic you'd find helpful.

Structure of the class

We have three sources of information that we'll learn from:

1. *My lectures* I have a set of tools that I want to introduce you to throughout the semester. I think of the lecture as offering "an overview" as well as "my take" on the tool. I will not supply all the details–we don't have enough time and a lecture isn't the ideal medium for deep and subtle ideas. In the past, I have supplied all of my lecture notes to students. However, the research seems clear that student note-taking boosts learning.
2. *Required readings* For each topic, I have a few readings selected to supply further details or offer a different perspective. I want you to carefully read the required readings, even if they seem familiar.
3. *Suggested and other readings* I encourage you to engage readings beyond the required set. These might be "easier" readings (e.g., FPP) or more difficult readings (e.g., Greene). In this category, I want you to use judgement. If the required readings are easy, then I recommend moving on *after* seriously engaging the required readings. If the required readings are too difficult, then seek out gentler introductions. You should NOT pursue the suggested or other readings at the expense of the required readings.

Assessments

This semester, we have a large set of tools that you must demonstrate that you (1) understand and (2) can implement.

1. Exams: We will have regular exams that require you to implement and explain particular tools. I'm open to suggestions on frequency, but I suggest a *weekly*, open-book, take-home exam with about a one hour time limit. I will grade these as pass/fail. You can re-take a (slightly modified) exam up to three times if you fail.
2. Journal: I want to you to journal throughout the semester. I want you to spend *at least* three hours (hopefully more most weeks) outside of class working on your journal. This journal should have several parts:
    a. Class Notes
    b. Review Exercises
    c. Notes from the required readings, including summaries, reactions, and (especially) questions or flags for ideas you didn't understand. This latter is very important–it will make us all better.
    d. Notes from other readings. I want to give you a bit of space to explore things on your own. You could do a deeper dive on ideas covered carefully in the lectures or readings. Or you could pursue a tangential topic (but keep it somewhat related to class). Again, summaries, reactions, and questions are appropriate. I suggest engaging with reading from substantive course with this class in mind, and record your thoughts in your journal.
    e. Connections throughout the semester.
    f. Explorations of ideas for future projects.

As I see it, "regression modeling" in political science is a several-step process:

You begin with a substantive understanding of the way the world works.

1. Choose a regression model. I introduce many.
2. Fit a regression model. Maximum likelihood and Markov chain Monte Carlo methods are powerful and general.
3. Evaluate the fit. What are the properties of the procedure? How well does the model match the data?
4. Interpret the model. I emphasize quantities of interest and confidence intervals, but also discuss hypothesis tests.

You then update your understanding of the world.

This week, I introduce our first "engine": maximum likelihood. As a starting point, we use ML to estimate the parameters of Bernoulli, Poisson, and beta distributions (without covariates). I introduce the parametric bootstrap as a tool to obtain confidence intervals. I introduce the invariance property and show how we can use the invariance property to transform the estimated parameters into other quantities of interest. To evaluate the models, we use the predictive distribution.

Maximum Likelihood

Suppose we have a random sample from a distribution $f(x;\theta)$. We find the maximum likelihood (ML) estimator $\hat{\theta}$ of θ by maximizing the likelihood of the observed data with respect to θ.

In short, we take the likelihood of the data (given the model and a particular θ) and find the parameter θ that maximizes it.

In practice, to make the math and/or computation a bit easier, we manipulate the likelihood function in two ways:

1. Relabel the likelihood function $f(x;\theta) = L(\theta)$, since it's weird to maximize with respect to a "conditioning variable"fixed" variable. (The notation $f(x;\theta)$ suggests x varies for a particular θ.)
2. Work with $\log L(\theta)$ rather than $L(\theta)$. Because log() is a monotonically increasing function, the θ that maximizes $L(\theta)$ also maximizes $\log L(\theta)$.

Suppose we have samples $x_1, x_2, ..., x_N$ from $f(x;\theta)$. Then the joint density/probability is $f(x;\theta) = \prod_{n=1}^{N} f(x_n;\theta)$ and $\log L(\theta) = \sum_{n=1}^{N} \log\left[f(x_n;\theta)\right]$. The ML estimator $\hat{\theta}$ of θ is $\arg\max \log L(\theta)$.

In applied problems, we might be able to simplify $\log L$ substantially. Occasionally, we can find a nice analytical maximum. In many cases, we have a computer find the parameter that maximizes $\log L$.

Example: Bernoulli Distribution

As a running example, we use the **toothpaste cap problem**:

> We have a toothpaste cap–one with a wide bottom and a narrow top. We're going to toss the toothpaste cap. It can either end up lying on its side, its (wide) bottom, or its (narrow) top.
>
> We want to estimate the probability of the toothpaste cap landing on its top.
>
> We can model each toss as a Bernoulli trial, thinking of each toss as a random variable X where $X \sim \text{Bernoulli}(\pi)$. If the cap lands on its top, we think of the outcome as 1. If not, as 0.

Suppose we toss the cap N times and observe k tops. What is the ML estimate $\hat{\pi}$ of π?

According to the model $f(x_i;\pi) = \pi^{x_i}(1-\pi)^{(1-x_i)}$. Because the samples are iid, we can find the *joint* distribution $f(x) = f(x_1) \times ... \times f(x_N) = \prod_{i=1}^{N} f(x_i)$. We're just multiplying k πs (i.e., each of the k ones has probability π) and $(N-k)$ $(1-\pi)$s (i.e., each of the $N-k$ zeros has probability $1-\pi$), so that the $f(x;\pi) = \pi^k(1-\pi)^{(N-k)}$.

$$\text{the likelihood: } f(x;\pi) = \pi^k(1-\pi)^{(N-k)}, \text{where } k = \sum_{n=1}^{N} x_n$$

Then, we relabel.

$$\text{the likelihood: } L(\pi) = \pi^k(1-\pi)^{(N-k)}$$

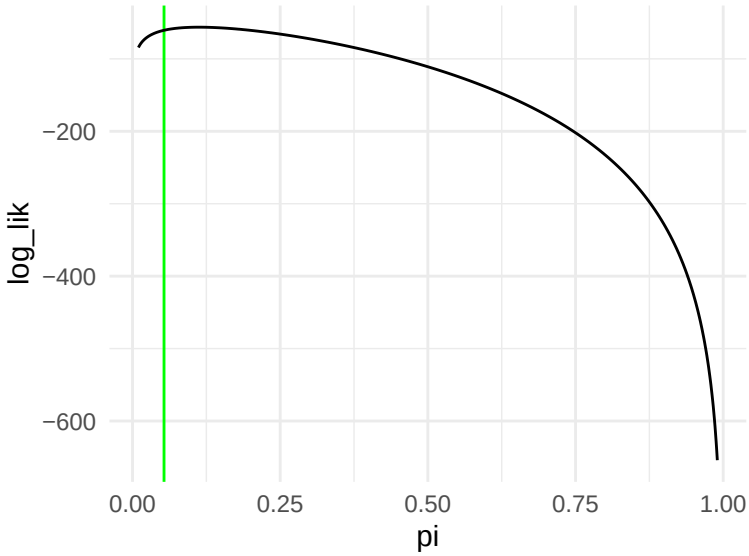
Then, we take the log and simplify.

$$\text{the log-likelihood: } \log L(\pi) = k\log(\pi) + (N-k)\log(1-\pi)$$

To find the ML estimator, we find $\hat{\pi}$ that maximizes $\log L$.

The code below plots the log-likelihood function using the 8/150 data.

```
pi <- seq(0.01, 0.99, length.out = 1000)
data <- tibble(pi = pi) %>%
  mutate(log_lik = 18*log(pi) + (150 - 8)*log(1 - pi))
ggplot(data, aes(x = pi, y = log_lik)) +
  geom_vline(xintercept = 8/150, color = "green") +
  geom_line() +
  theme_minimal()
```



In this case, the analytical optimum is easy.

$$\begin{aligned}
\frac{d\log L}{d\hat{\pi}} = k\left(\frac{1}{\hat{\pi}}\right) + (N-k)\left(\frac{1}{1-\hat{\pi}}\right)(-1) &= 0 \\
\frac{k}{\hat{\pi}} - \frac{N-y}{1-\hat{\pi}} &= 0 \\
\frac{k}{\hat{\pi}} &= \frac{N-y}{1-\hat{\pi}} \\
k(1-\hat{\pi}) &= (N-y)\hat{\pi} \\
k - y\hat{\pi} &= N\hat{\pi} - y\hat{\pi} \\
k &= N\hat{\pi} \\
\hat{\pi} &= \frac{k}{N} = \text{avg}(x)
\end{aligned}$$

The ML estimator of π is the average of the N Bernoulli trials, or, equivalently, the fraction of successes.

The collected data consist of 150 trials and 8 successes, so the ML estimate of π is $\frac{8}{150} \approx 0.053$.

Example: Poisson Distribution

Suppose we collect N random samples $x = \{x_1, x_2, ..., x_N\}$ and model each draw as a random variable $X \sim \text{Poisson}(\lambda)$. Find the ML estimator of λ.

$$\begin{aligned}
\text{Poisson likelihood: } f(x;\lambda) &= \prod_{n=1}^{N} \frac{\lambda^{x_n} e^{-\lambda}}{x_n!} \\
L(\lambda) &= \prod_{n=1}^{N} \frac{\lambda^{x_n} e^{-\lambda}}{x_n!} \\
\log L(\lambda) &= \sum_{n=1}^{N} \log\left[\frac{\lambda^{x_n} e^{-\lambda}}{x_n!}\right] \\
&= \sum_{n=1}^{N} \left[x_n \log\lambda + (-\lambda)\log e - \log x_n!\right] \\
&= \log\lambda \left[\sum_{n=1}^{N} x_n\right] - N\lambda + \sum_{n=1}^{N} \log(x_n!)
\end{aligned}$$

To find the ML estimator, we find $\hat{\lambda}$ that maximizes $\log L$. In this case, the analytical optimum is easy.

$$\begin{aligned}
\frac{d\log L}{d\hat{\lambda}} = \frac{1}{\hat{\lambda}}\left[\sum_{n=1}^{N} x_n\right] - N &= 0 \\
\frac{1}{\hat{\lambda}}\left[\sum_{n=1}^{N} x_n\right] &= N \\
\left[\sum_{n=1}^{N} x_n\right] &= N\hat{\lambda} \\
\hat{\lambda} &= \frac{\sum_{n=1}^{N} x_n}{N} = \text{avg}(x)
\end{aligned}$$

The ML estimator for the Poisson distribution is just the average of the samples.

Remarks

The ML estimator is extremely common in political science because they are general, fast, and work extremely well. Lots of models that you've heard of, such as logistic regression, are estimated with ML.

We can even obtain ML estimates for the linear regression model. We assume that the observed data are samples from a normal distribution with mean $\mu_n = \alpha + \beta x_n$ and variance σ^2. For this model, the least-squares estimate that we learned earlier is also the ML estimate.

Example: Beta Distribution

Questions:

1. What is the *support* of the beta distribution? $[0, 1]$
2. Is y a discrete random variable or a continuous random variable? Continuous.
3. What is the pdf/pmf? $f(y_i;\alpha,\beta) = \dfrac{y_i^{\alpha-1}(1-y_i)^{\beta-1}}{B(\alpha,\beta)}$, where $B(\alpha,\beta) = \displaystyle\int_0^1 t^{\alpha-1}(1-t)^{\beta-1}dt$.

With the beta distribution, we add two complications that typically occur when using ML.

1. multiple parameters
2. an intractable log-likelihood

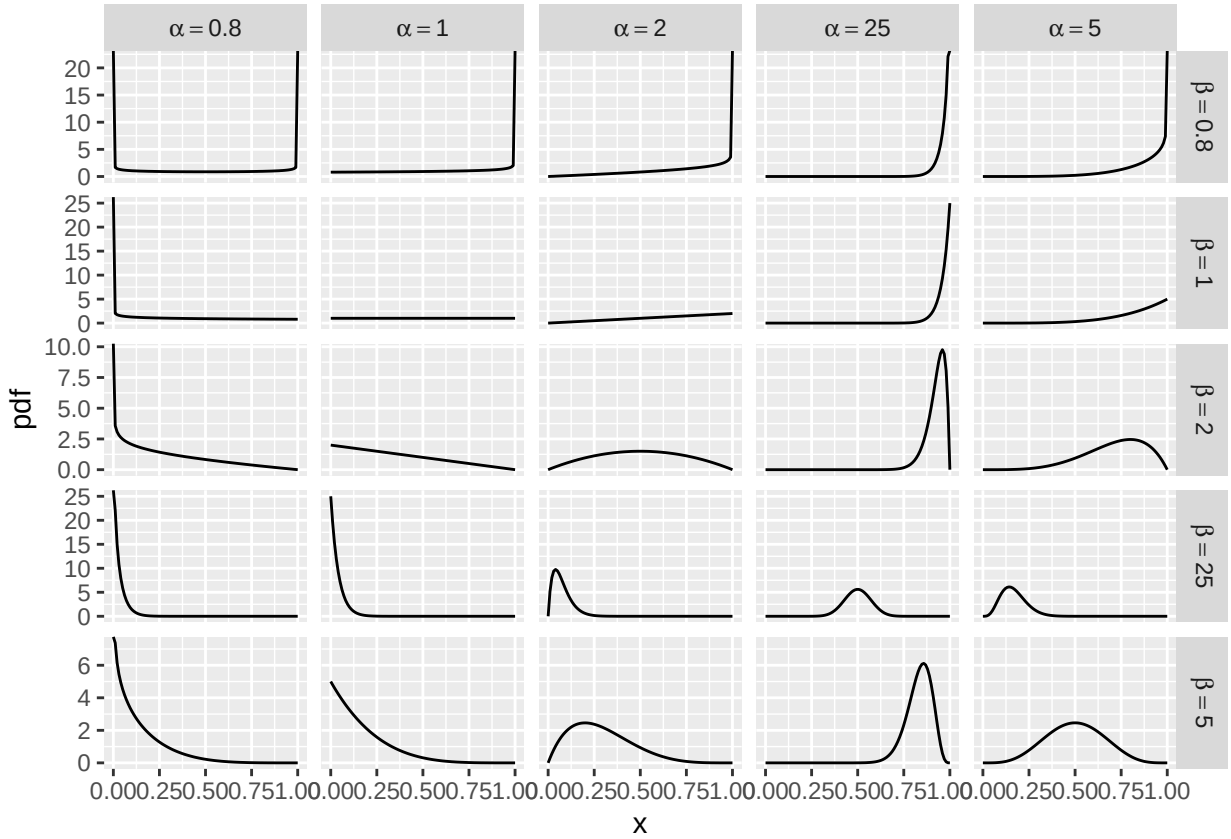
Start with the probability model $Y_i \sim f(y_i;\theta)$. In the case of the beta model, we have $Y_i \sim \text{beta}(y_i;\alpha,\beta)$. The α and β here don't have a convenient interpretation. They are "shape" parameters. You can think of α as pushing the distribution to the right and β as pushing the distribution to the left.

```
alphas <- c(0.8, 1, 2, 5, 25)
betas <- c(0.8, 1, 2, 5, 25)

x <- seq(0, 1, length.out = 100)

pdfs <- crossing(alpha = alphas,
                 beta = betas,
                 x = x) %>%
  mutate(pdf = dbeta(x, alpha, beta)) %>%
  mutate(alpha_lbl = paste0("alpha == ", alpha),
         beta_lbl = paste0("beta == ", beta))

ggplot(pdfs, aes(x = x, y = pdf)) +
  facet_grid(rows = vars(beta_lbl), cols = vars(alpha_lbl),
             labeller = "label_parsed", scales = "free") +
  geom_line()
```



We now have two parameters to estimate and we're going to assume that we have multiple observations, so that $y = [y_1, y_2, , ..., y_n]$.

In general, this is how we do ML:

Step 1 Write down the likelihood function. Recall that we can obtain the joint density of y_1 AND y_2 AND ... AND y_n by multiplying the probabilities of each (assuming independence).

$$L(\alpha, \beta) = \prod_{i=1}^{n} \overbrace{f(y_i; \alpha, \beta)}^{\text{density}} = \prod_{i=1}^{n} \frac{y_i^{\alpha-1}(1-y_i)^{\beta-1}}{B(\alpha, \beta)}$$

We see again, as will be usual, that we have this complicated product that will make our lives difficult.

Step 2 Take the log and simplify.

$$
\begin{aligned}
L(\alpha, \beta) &= \prod_{i=1}^{n} \frac{y_i^{\alpha-1}(1-y_i)^{\beta-1}}{B(\alpha,\beta)} \\
\log L(\alpha, \beta) &= \sum_{i=1}^{n} \log \frac{y_i^{\alpha-1}(1-y_i)^{\beta-1}}{B(\alpha,\beta)} \\
&= \sum_{i=1}^{n} \left[\log y_i^{\alpha-1} + \log(1-y_i)^{\beta-1} - \log B(\alpha,\beta)\right] \\
&= \sum_{i=1}^{n} \left[(\alpha-1)\log y_i + (\beta-1)\log(1-y_i) - \log B(\alpha,\beta)\right] \\
&= \sum_{i=1}^{n} \left[(\alpha-1)\log y_i + (\beta-1)\log(1-y_i)\right] - n\log B(\alpha,\beta) \\
\log L(\alpha, \beta) &= (\alpha-1)\sum_{i=1}^{n} \log y_i + (\beta-1)\sum_{i=1}^{n} \log(1-y_i) - n\log B(\alpha,\beta)
\end{aligned}
$$

Step 3 Maximize

If we wanted, we could work on this one analytically.

1. Take the derivative w.r.t. α.
2. Take the derivative w.r.t. β.
3. Set both equal to zero and solve. (Two equations and two unknowns.)

But the last term $B(\alpha, \beta) = \int_0^1 t^{\alpha-1}(1-t)^{\beta-1}dt$ is tricky! So let's do it numerically.

To perform the optimization, we need a data set. For now, let's simulate a fake data set with known parameters

```
y <- rbeta(1000, shape1 = 10, shape2 = 10)
```

Let's plot the log-likelihood function to see what we're dealing with.

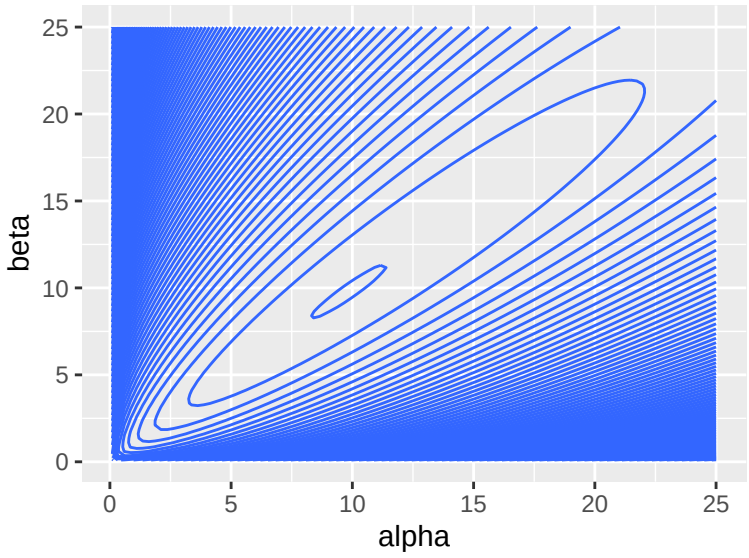
```
library(plotly)

alpha <- seq(0.1, 25, length.out = 100)
beta  <- seq(0.1, 25, length.out = 100)
data <- crossing(alpha, beta) %>%
  mutate(log_lik = alpha*sum(log(y)) + beta*sum(log(1 - y)) -
           length(y)*log(beta(alpha, beta)))

plot_ly(x = ~alpha, y = ~beta, z = ~log_lik, data = data) %>%
  add_mesh(labels = c("alpha", "beta", "log-likelihood"))
```

WebGL is not supported by your browser - visit https://get.webgl.org for more info

```
ggplot(data, aes(x = alpha, y = beta, z = log_lik)) +
  geom_contour(bins = 100)
```



Now let's program the log-likelihood function in R to handle the optimization numerically.

```
ll_fn <- function(theta, y) {
  alpha <- theta[1]  # optim() requires a single parameter vector
  beta <- theta[2]
  ll <- alpha*sum(log(y)) + beta*sum(log(1 - y)) -
          length(y)*log(beta(alpha, beta))
  return(ll)
```

```
}
```

Now let's use `optim()` to do the maximization.

```
est <- optim(par = c(1, 1), fn = ll_fn, y = y,
               control = list(fnscale = -1),
               method = "Nelder-Mead")

print(est$par, digits = 3)
```

```
## [1] 9.76 9.71
```

We can also wrap the `optim()` in a function, to make obtaining the estimates a little bit easier.

```
est_beta <- function(y) {
  est <- optim(par = c(1, 1), fn = ll_fn, y = y,
               control = list(fnscale = -1),
               method = "Nelder-Mead") # for >1d problems
  if (est$convergence != 0) print("Model did not converge!")
  res <- list(est = est$par)
  return(res)
}

ml_est <- est_beta(y)
print(ml_est, digits = 3)
```

```
## $est
## [1] 9.76 9.71
```