# Estimation

## Maximum Likelihood and Smoothing

Introduction to Natural Language Processing
Computer Science 585—Fall 2009
University of Massachusetts Amherst

David Smith

# Simple Estimation

- Probability courses usually start with equiprobable events
  - Coin flips, dice, cards
- How likely to get a 6 rolling 1 die?
- How likely the sum of two dice is 6?
- How likely to see 3 heads in 10 flips?

# Binomial Distribution

For *n* trials, *k* successes, and success probability *p*:

$$P(k) = \binom{n}{k} p^k (1-p)^{n-k}$$

Prob. mass function

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Estimation problem: If we observe *n* and *k*, **what is *p*?**

# Maximum Likelihood

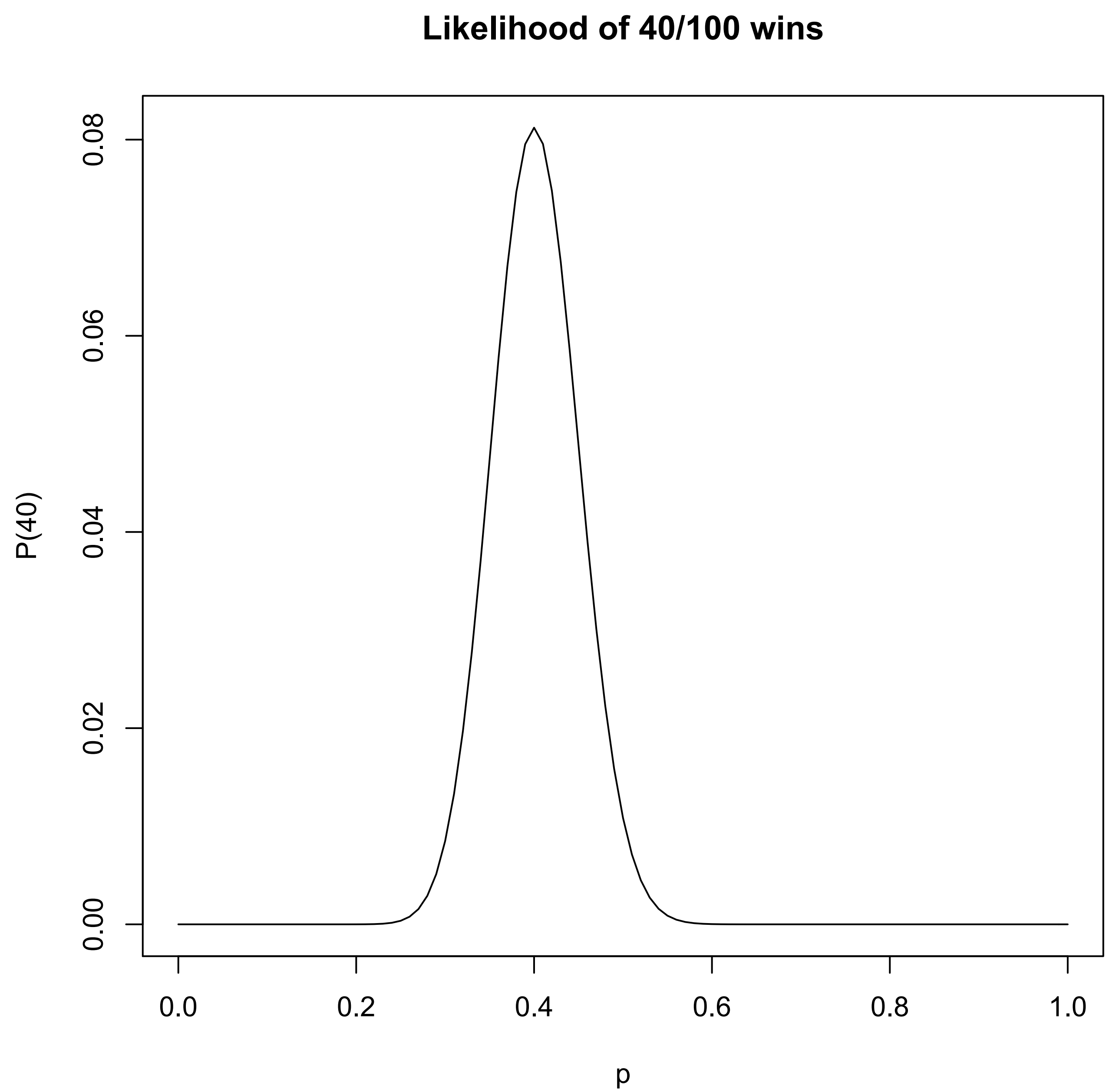
Say we win 40 games out of 100.

$$P(40) = \binom{100}{40} p^{40}(1-p)^{60}$$

The maximum likelihood estimator for *p* solves:

$$\max_p P(\text{observed data}) = \max_p \binom{100}{40} p^{40}(1-p)^{60}$$

# Maximum Likelihood



# Maximum Likelihood

How to solve $$\max_p \binom{100}{40} p^{40}(1-p)^{60}$$

# Maximum Likelihood

How to solve $\max_p \binom{100}{40} p^{40}(1-p)^{60}$

$$
\begin{aligned}
0 &= \frac{\partial}{\partial p}\binom{100}{40} p^{40}(1-p)^{60} \\
&= 40p^{39}(1-p)^{60} - 60p^{40}(1-p)^{59} \\
&= p^{39}(1-p)^{59}[40(1-p) - 60p] \\
&= p^{39}(1-p)^{59} 40 - 100p
\end{aligned}
$$

# Maximum Likelihood

How to solve $\max_p \binom{100}{40} p^{40}(1-p)^{60}$

$$
\begin{aligned}
0 &= \frac{\partial}{\partial p}\binom{100}{40} p^{40}(1-p)^{60} \\
&= 40p^{39}(1-p)^{60} - 60p^{40}(1-p)^{59} \\
&= p^{39}(1-p)^{59}[40(1-p) - 60p] \\
&= p^{39}(1-p)^{59} 40 - 100p
\end{aligned}
$$

Solutions: 0, 1, .4

# Maximum Likelihood

How to solve $\max_p \binom{100}{40} p^{40}(1-p)^{60}$

$$\begin{aligned} 0 &= \frac{\partial}{\partial p}\binom{100}{40} p^{40}(1-p)^{60} \\ &= 40p^{39}(1-p)^{60} - 60p^{40}(1-p)^{59} \\ &= p^{39}(1-p)^{59}[40(1-p) - 60p] \\ &= p^{39}(1-p)^{59} 40 - 100p \end{aligned}$$

The maximizer!

Solutions: 0, 1, .4

# Maximum Likelihood

How to solve $\max_p \binom{100}{40} p^{40}(1-p)^{60}$

$$
\begin{aligned}
0 &= \frac{\partial}{\partial p}\binom{100}{40} p^{40}(1-p)^{60} \\
&= 40p^{39}(1-p)^{60} - 60p^{40}(1-p)^{59} \\
&= p^{39}(1-p)^{59}[40(1-p) - 60p] \\
&= p^{39}(1-p)^{59} 40 - 100p
\end{aligned}
$$

The maximizer!

In general, *k*/*n*

Solutions: 0, 1, .4

# Maximum Likelihood

How to solve $\max_p \binom{100}{40} p^{40}(1-p)^{60}$

$$
\begin{aligned}
0 &= \frac{\partial}{\partial p}\binom{100}{40} p^{40}(1-p)^{60} \\
&= 40p^{39}(1-p)^{60} - 60p^{40}(1-p)^{59} \\
&= p^{39}(1-p)^{59}[40(1-p) - 60p] \\
&= p^{39}(1-p)^{59} 40 - 100p
\end{aligned}
$$

The maximizer!

In general, *k*/*n*

Solutions: 0, 1, .4

This is trivial here, but a widely useful approach.

# ML for Language Models

- Say the corpus has "in the" 100 times
- If we see "in the beginning" 5 times,

  $p_{ML}$(beginning | in the) = ?
- If we see "in the end" 8 times,

  $p_{ML}$(end | in the) = ?
- If we see "in the kitchen" 0 times,

  $p_{ML}$(kitchen | in the) = ?

# ML for Naive Bayes

- Recall: p(+ | Damon movie)

  = p(Damon | +) p(movie | +) p(+)

- If corpus of positive reviews has 1000 words, and "Damon" occurs 50 times,

  $p_{ML}$(Damon | +) = ?

- If pos. corpus has "Affleck" 0 times,

  p(+ | Affleck Damon movie) = ?

# Will the Sun Rise Tomorrow?

# Will the Sun Rise Tomorrow?

Laplace's Rule of Succession:
On day *n*+1, we've observed that the sun has risen *s* times before.

$$p_{Lap}(S_{n+1} = 1 \mid S_1 + \cdots + S_n = s) = \frac{s+1}{n+2}$$

What's the probability on day 0?
On day 1?
On day $10^6$?
Start with prior assumption of equal rise/not-rise probabilities; *update* after every observation.

# Laplace (Add One) Smoothing

- From our earlier example:

  $p_{ML}$(beginning | in the) = 5/100? reduce!

  $p_{ML}$(end | in the) = 8/100? reduce!

  $p_{ML}$(kitchen | in the) = 0/100? increase!

# Laplace (Add One) Smoothing

- Let V be the vocabulary size:

  i.e., the number of unique words that could follow "in the"

- From our earlier example:

  $p_{ML}$(beginning | in the) = (5 + 1)/(100 + V)

  $p_{ML}$(end | in the) = (8 + 1)/(100 + V)

  $p_{ML}$(kitchen | in the) = (0 + 1) / (100 + V)

# Generalized Additive Smoothing

- Laplace add-one smoothing now assigns *too much* probability to unseen words
- More common to use λ instead of 1:

$$
\begin{aligned}
p(w_3 \mid w_1, w_2) &= \frac{C(w_1, w_2, w_3) + \lambda}{C(w_1, w_2) + \lambda V} \\
&= \mu \frac{C(w_1, w_2, w_3)}{C(w_1, w_2)} + (1 - \mu) \frac{1}{V} \\
\mu &= \frac{C(w_1, w_2)}{C(w_1, w_2) + \lambda V}
\end{aligned}
$$

# Generalized Additive Smoothing

- Laplace add-one smoothing now assigns *too much* probability to unseen words
- More common to use λ instead of 1:

$$
\begin{aligned}
p(w_3 \mid w_1, w_2) &= \frac{C(w_1, w_2, w_3) + \lambda}{C(w_1, w_2) + \lambda V} \\
&= \mu \frac{C(w_1, w_2, w_3)}{C(w_1, w_2)} + (1 - \mu) \frac{1}{V} \\
\mu &= \frac{C(w_1, w_2)}{C(w_1, w_2) + \lambda V}
\end{aligned}
$$

interpolation

# Generalized Additive Smoothing

- Laplace add-one smoothing now assigns *too much* probability to unseen words
- More common to use λ instead of 1:

What's the right λ?

$$p(w_3 \mid w_1, w_2) = \frac{C(w_1, w_2, w_3) + \lambda}{C(w_1, w_2) + \lambda V}$$

interpolation

$$= \mu \frac{C(w_1, w_2, w_3)}{C(w_1, w_2)} + (1 - \mu) \frac{1}{V}$$

$$\mu = \frac{C(w_1, w_2)}{C(w_1, w_2) + \lambda V}$$

# Picking Parameters

- What happens if we optimize parameters on training data, i.e. the same corpus we use to get counts?
- Maximum likelihood estimate!
- Use *held-out data* aka *development data*

# Good-Turing Smoothing

- Intuition: Can judge rate of novel events by rate of singletons
  - Developed to estimate # of unseen species in field biology
- Let $N_r$ = # of word types with r training tokens
  - e.g., $N_0$ = number of unobserved words
  - e.g., $N_1$ = number of singletons (hapax legomena)
- Let $N = \sum r N_r$ = total # of training tokens

# Good-Turing Smoothing

- Max. likelihood estimate if w has r tokens? r/N
- Total max. likelihood probability of all words with r tokens? $N_r$ r / N
- Good-Turing estimate of this total probability:
  - Defined as: $N_{r+1}$ (r+1) / N
  - So proportion of novel words in test data is estimated by proportion of singletons in training data.
  - Proportion in test data of the $N_1$ singletons is estimated by proportion of the $N_2$ doubletons in training data. etc.
  - p(any given word w/freq. r) = $N_{r+1}$ (r+1) / (N $N_r$)
- NB: No parameters to tune on held-out data

# Backoff

- Say we have the counts:

  C(in the kitchen) = 0

  C(the kitchen) = 3

  C(kitchen) = 4

  C(arboretum) = 0

- ML estimates seem counterintuitive:

  p(kitchen | in the) = p(arboretum | in the) = 0

# Backoff

- Clearly we shouldn't treat "kitchen" the same as "arboretum"
- Basic add-λ (and other) smoothing methods assign the same prob. to *all* unseen events
- **Backoff** divides up prob. of unseen unevenly in proportion to, e.g., lower-order n-grams
- If $p(z \mid x,y) = 0$, use $p(z \mid y)$, etc.

# Deleted Interpolation

- Simplest form of backoff
- Form a *mixture* of different order n-gram models; learn weights on held-out data

$$\begin{aligned} p_{del}(z \mid x, y) &= \alpha_3 p(z \mid x, y) + \alpha_2 p(z \mid y) + \alpha_1 p(z) \\ \sum \alpha_i &= 1 \end{aligned}$$

- How else could we back off?

# Readings, etc.

- For more information on basic probability, read M&S 2.1
- For more information on language model estimation, read M&S 6
- Next, time Hidden Markov Models