

# STAT 238 - Bayesian Statistics

## Lecture Fourteen

Spring 2026, UC Berkeley

Aditya Guntuboyina

23 Feb 2026

We will illustrate Dirichlet-Multinomial inference on a text data example. This analysis is taken from MacKay and Peto [1].

### 1 A Text Example

We observe some text  $y_1, \dots, y_n$ . The vocabulary consists of the number of distinct words (including punctuation marks) in the text. Assume the vocabulary size is  $k$ .

#### 1.1 A simple i.i.d model

The simplest model assumes that  $y_1, \dots, y_n$  are i.i.d with an distribution  $P$  that is supported on the vocabulary. The likelihood then becomes:

$$\prod_{i=1}^n p_{y_i} = \prod_{i=1}^k p_i^{N_i} \quad (1)$$

where  $N_i$  is the observed count for the  $i$ -th word in the vocabulary. We can place a Dirichlet prior on  $(p_1, \dots, p_k)$  (e.g.,  $\text{Dirichlet}(0, \dots, 0)$ ) which would result in the  $\text{Dirichlet}(N_1, \dots, N_k)$  posterior distribution. One can then generate new text in the following way:

1. First generate  $(p_1, \dots, p_k)$  from the posterior distribution.
2. Then generate new words  $y_{n+1}, y_{n+2}, \dots$  i.i.d from the multinomial distribution  $\text{Mult}(1; p_1, \dots, p_k)$  until a specific word (e.g., the period symbol '.') is generated.

Obviously, this will not result in realistic text. This is because the i.i.d assumption collapses the text into a bag-of-words representation, discarding all sequential information. Indeed, in this model, we are effectively working with the counts  $N_1, \dots, N_k$  instead of the raw data  $y_1, \dots, y_n$ . The likelihood in terms of the counts is:

$$\frac{n!}{N_1! \dots N_k!} p_1^{N_1} \dots p_k^{N_k} \propto \prod_{i=1}^k p_i^{N_i}$$

which coincides with (1). As a result, it cannot capture even the most basic linguistic dependencies (e.g., that “the” is likely to be followed by a noun).

## 1.2 AR(1) model

A slightly better model will be obtained if we replace the i.i.d assumption with a Markovian one. This is also referred to as the first order AutoRegressive AR(1) model.

The likelihood here then becomes:

$$p_{y_1} p_{y_2|y_1} p_{y_3|y_2, y_1} \cdots p_{y_n|y_{n-1}, \dots, y_1}$$

We assume now that  $p_{y_i|y_{i-1}, \dots, y_1} = p_{y_i|y_{i-1}}$  for each  $i = 3, \dots, n$ . This gives

$$\text{Likelihood} = p_{y_1} \prod_{i=2}^n p_{y_i|y_{i-1}}.$$

We also assume that  $p_{y_1}$  is constant so we don't need to model it. This gives:

$$\text{Likelihood} \propto \prod_{i=2}^n p_{y_i|y_{i-1}}. \quad (2)$$

If you don't like the assumption that  $p_{y_1}$  is constant, you can interpret the above as the conditional likelihood of  $(y_2, \dots, y_n)$  given  $y_1$ .

Just as in the i.i.d model where we could write the likelihood in terms of counts, we can rewrite the likelihood (2) using 'bi-gram counts'. For each  $i$  and  $j$  in  $\{1, \dots, k\}$ , let  $x_{j|i}$  denote the number of times the word  $j$  follows word  $i$  in the text. Also let  $x_i = \sum_{j=1}^k x_{j|i}$  denote the number of times word  $i$  appears as the "previous word" in the text.

Then the likelihood (2) is the same as:

$$\prod_{i=1}^k \prod_{j=1}^k p_{j|i}^{x_{j|i}} \quad (3)$$

where  $p_{j|i}$  denotes the probability that word  $j$  appears as the next word given that the previous word is  $i$ .

We can also directly write the likelihood in terms of the bigram counts  $(x_{j|i}, j = 1, \dots, k)$  and  $i = 1, \dots, k$ . The model here is:

$$(x_{j|i}, j = 1, \dots, k) \stackrel{\text{i.i.d}}{\sim} \text{Multinomial}(x_i; p_{j|i}, j = 1, \dots, k).$$

so that the likelihood in terms of the bigram counts is:

$$\prod_{i=1}^k \frac{x_i!}{\prod_{j=1}^k x_{j|i}!} \prod_{j=1}^k p_{j|i}^{x_{j|i}}. \quad (4)$$

The unknown parameters in this model are the probability vectors  $(p_{j|i}, 1 \leq j \leq k)$  for each  $i = 1, \dots, k$ .

MacKay and Peto [1] perform Bayesian inference with the following prior:

$$(p_{j|i}, j = 1, \dots, k) \stackrel{\text{i.i.d}}{\sim} \text{Dirichlet}(a_1, \dots, a_k)$$

for some  $a_1, \dots, a_k > 0$ . This means that all the  $k$  probability vectors  $(p_{j|i}, 1 \leq j \leq k)$  are assumed to be independently distributed according to the same Dirichlet distribution  $\text{Dirichlet}(a_1, \dots, a_k)$ . The prior density is therefore

$$\prod_{i=1}^k \frac{\Gamma(a_1 + \dots + a_k)}{\Gamma(a_1) \dots \Gamma(a_k)} \prod_{j=1}^k p_{j|i}^{a_j - 1}.$$

The posterior is then:

$$\prod_{i=1}^k \frac{x_i!}{\prod_{j=1}^k x_{j|i}!} \prod_{j=1}^k p_{j|i}^{x_{j|i}} \times \prod_{i=1}^k \frac{\Gamma(a_1 + \dots + a_k)}{\Gamma(a_1) \dots \Gamma(a_k)} \prod_{j=1}^k p_{j|i}^{a_j-1} \propto \prod_{i=1}^k \prod_{j=1}^k p_{j|i}^{x_{j|i} + a_j - 1}$$

which means that:

$$(p_{j|i}, j = 1, \dots, k) \stackrel{\text{ind}}{\sim} \text{Dirichlet}(x_{j|i} + a_j, j = 1, \dots, k).$$

So the posterior mean estimate of  $p_{j|i}$  is:

$$\hat{p}_{j|i} = \frac{x_{j|i} + a_j}{x_i + a_1 + \dots + a_k}.$$

The choice of  $a_1, \dots, a_k$  controls the properties of the above posterior. Instead of plugging in some values for  $a_1, \dots, a_k$ , MacKay and Peto [1] propose estimating them by maximizing the marginal likelihood of the data  $(x_{j|i}, 1 \leq i, j \leq k)$  over all values of  $a_1, \dots, a_k$ .

The marginal distribution of the data given the Dirichlet hyperparameters  $a_1, \dots, a_k$  is:

$$\begin{aligned} & \mathbb{P} \{x_{j|i}, 1 \leq i, j \leq k \mid a_1, \dots, a_k\} \\ &= \int \prod_{i=1}^k \frac{x_i!}{\prod_{j=1}^k x_{j|i}!} \prod_{j=1}^k p_{j|i}^{x_{j|i}} \times \prod_{i=1}^k \frac{\Gamma(a_1 + \dots + a_k)}{\Gamma(a_1) \dots \Gamma(a_k)} \prod_{j=1}^k p_{j|i}^{a_j-1} d(p_{j|i} \text{ all } j, i) \\ &= \prod_{i=1}^k \frac{x_i!}{\prod_{j=1}^k x_{j|i}!} \frac{\Gamma(a_1 + \dots + a_k)}{\Gamma(a_1) \dots \Gamma(a_k)} \int \prod_{j=1}^k p_{j|i}^{x_{j|i} + a_j - 1} d(p_{j|i}, 1 \leq j \leq k) \\ &= \prod_{i=1}^k \frac{x_i!}{\prod_{j=1}^k x_{j|i}!} \left[ \frac{\Gamma(a_1 + \dots + a_k)}{\Gamma(x_i + a_1 + \dots + a_k)} \prod_{j=1}^k \frac{\Gamma(x_{j|i} + a_j)}{\Gamma(a_j)} \right]. \end{aligned}$$

In the machine learning literature, the marginal distribution of the data given the hyperparameters of the prior is referred to as the Evidence (see e.g., [https://en.wikipedia.org/wiki/Marginal\\_likelihood](https://en.wikipedia.org/wiki/Marginal_likelihood)). So:

$$\text{Evidence}(a_1, \dots, a_k) = \prod_{i=1}^k \frac{x_i!}{\prod_{j=1}^k x_{j|i}!} \left[ \frac{\Gamma(a_1 + \dots + a_k)}{\Gamma(x_i + a_1 + \dots + a_k)} \prod_{j=1}^k \frac{\Gamma(x_{j|i} + a_j)}{\Gamma(a_j)} \right].$$

The Evidence can be maximized over  $a_1, \dots, a_k$  to get a good choice of the hyperparameters. Note that the factorial terms do not involve  $a_j$ s so they can be dropped in this maximization:

$$\text{Evidence}(a_1, \dots, a_k) \propto \prod_{i=1}^k \left[ \frac{\Gamma(a_1 + \dots + a_k)}{\Gamma(x_i + a_1 + \dots + a_k)} \prod_{j=1}^k \frac{\Gamma(x_{j|i} + a_j)}{\Gamma(a_j)} \right].$$

It is always better to optimize the logarithm of the evidence (rather than the evidence directly):

$$\begin{aligned} & \log \text{Evidence}(a_1, \dots, a_k) \\ &= \text{constant} + \sum_{i=1}^k \left\{ \log \Gamma(A) - \log \Gamma(x_i + A) + \sum_{j=1}^k [\log \Gamma(x_{j|i} + a_j) - \log \Gamma(a_j)] \right\}. \end{aligned}$$

where  $A := a_1 + \dots + a_k$  and  $x_i = \sum_{j=1}^k x_{j|i}$ . One can compute the gradient with respect to  $a$  using the digamma function (see [https://en.wikipedia.org/wiki/Digamma\\_function](https://en.wikipedia.org/wiki/Digamma_function)):

$$\Psi(z) := \frac{d}{dz} \log \Gamma(z).$$

This gives:

$$\frac{\partial}{\partial a_j} \log \text{Evidence}(a_1, \dots, a_k) = \sum_{i=1}^k \{ \Psi(A) - \Psi(x_i + A) + \Psi(x_{j|i} + a_j) - \Psi(a_j) \}.$$

With these gradients (scipy has an inbuilt function for digamma calculation), some numerical optimization routine can be used for maximizing the log-Evidence over  $a_1, \dots, a_k$  (see code).

## References

- [1] D. J. C. MacKay and L. C. Peto, "A hierarchical Dirichlet language model," *Natural Language Engineering*, vol. 1, no. 3, pp. 289–308, 1995.