Lecture Notes III: Discrete probability in practice – Small Probabilities

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

January, 2025

The problem with estimating small probabilities

Definitions and setup

Additive methods (Laplace, Dirichlet, Bayesian, ELE)

Discounting (Ney-Essen)

Multiplicative smoothing: Estimating the next outcome (Witten-Bell, Good-Turing)

Back-off or shrinkage – mixing with simpler models

The problem with estimating small probabilities

Definitions and setup

We will look at estimating categorical distributions from samples, when the number of outcomes m is large.

- Let $S = \{1, \ldots m\}$ be the sample space, and $P = (\theta_1, \ldots \theta_m)$ a distribution over S.
- We draw n independent samples from P, obtaining the **data set** $\mathcal{D}$
- Define **the counts** $\{n_j = \#j$ appears in $\mathcal{D},\ i = 1, \ldots n\}$. The counts are also called **sufficient statistics** or **histogram**.
- Define the **fingerprint** (or **histogram of histogram**) of $\mathcal{D}$ as the counts of the counts, i.e $\{\, r_k = \#\text{counts}\, n_j = k,\ \text{for } k = 0, 1, 2 \ldots\}$

Example $m = 26$ alphabet letters

Data	**Counts** n_i	**Fingerprint** r_k
	$n_j = 0$: a,b,g,j,l,m,n,p,v,w,y,z	$r_0 = 12 = \lvert\{a,b,g,\ldots,y,z\}\rvert$
`the red fox is quick`		$r_1 = 12 = \lvert\{c,d,f,h,\ldots,u,x\}\rvert$
$n = 16$ letters	$n_j = 1$: c,d,f,h,k,o,q,r,s,t,u,x	$r_2 = 2 = \lvert\{e,i\}\rvert$
	$n_j = 2$: e,i	$r_3 = \ldots r_n = 0$
	$n_j = 0$: a,b,c...,x,z	$r_0 = 26 - 6 - 1 - 1 - 1 = 17$
	$n_j = 1$: f,i,n,r,t,w	$r_1 = 6 = \lvert\{f,i,n,r,t,w\}\rvert$
`ho ho who s on first`	$n_j = 2$: s	$r_2 = 1 = \lvert\{s\}\rvert$
$n = 15$ letters	$n_j = 3$: h	$r_3 = 1 = \lvert\{h\}\rvert$
	$n_j = 4$: o	$r_4 = 1 = \lvert\{o\}\rvert$

- It is easy to verify that $n_j \in 0 : n$, hence $r_{0:n}$ may be non-zero (but $r_{n+1,n+2,\ldots} = 0$), and that

$$m \;=\; r_0 + r_1 + \ldots r_n \quad n \;=\; 0 \times r_0 + 1 \times r_1 + \ldots k \times r_k + \ldots \tag{1}$$

Smoothing on an example

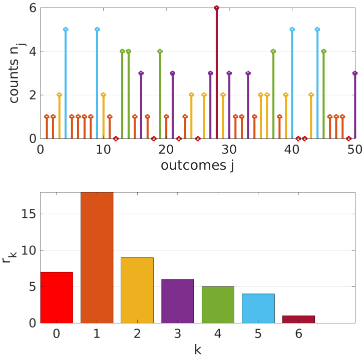
- **the counts** $\{n_j = \#j \text{ appears in } \mathcal{D},\ i = 1, \dots n\}$ (or **sufficient statistics** or **histogram**)
- **fingerprint** (or **histogram of histogram**) of $\mathcal{D}$ as the counts of the counts $\{\ r_k = \#\text{counts}\, n_j = k,\ \text{for } k = 0, 1, 2 \dots\}$, and $R_k = \{\ j,\ n_j = k, \}$

Example $m = 26$ alphabet letters

Data	**Counts** n_i	**Fingerprint** r_k
	$n_j = 0$: a,b,g,j,l,m,n,	$r_0 = 12 = \lvert\{\text{a,b,g,} \dots \text{,y,z}\}\rvert$
`the red fox is quick`	p,v,w,y,z	$r_1 = 12 = \lvert\{\text{c,d,f,h,} \dots \text{,u,x}\}\rvert$
$n = 16$ letters	$n_j = 1$: c,d,f,h,k,o,q,r,s,t,u,x	$r_2 = 2 = \lvert\{\text{e,i}\}\rvert$
	$n_j = 2$: e,i	$r_3 = \dots r_n = 0$

Examples of counts $n_{1:m}$ and fingerprints $r_{0:n}$

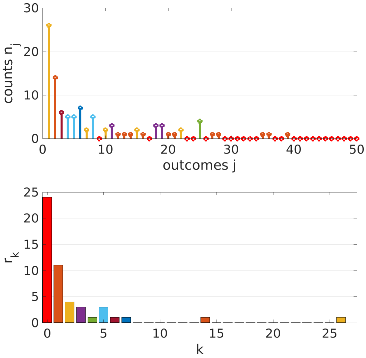
Uniform distribution



$S = \{1, \ldots m\}$, $|S| = m = 50$, $n = 100$

$\theta_j = 1/m$ for $j = 1 : m$

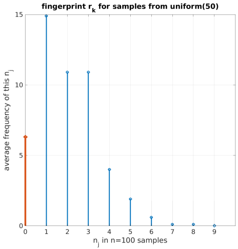
Non-uniform distribution



$S = \{1, \ldots m\}$, $|S| = m = 50$, $n = 100$

$\theta_j \propto \frac{j}{m}$ for $j = 1 : m$

The problem with small probabilities and large m



- when θ_i is small n must be very large to be able to observe i w.h.p.
- when m is large most θ_i are small

- Hence, in a sample of size n, many outcomes j may have $n_j = 0$, that is will not appear at all.

- **type** k $R_k = \{j \in S,\ n_j = k\}$ is the subset of outcomes in S that appear k times in $\mathcal{D}$
- Why are types important?
  - Because $\theta_j^{ML} = n_j/n$, all $i \in$ type k will have the same estimated value $\theta_j^{ML} = k/n$.
  - If $j, j' \in R_k$, no matter what correction method you use, there is no reason to distinguish between θ_j and $\theta_{j'}$. Hence $\theta_j = \theta_{j'}$ whenever $j, j' \in R_k$
  - Let $p_k = Pr[R_k]$. We have $p_k = r_k \theta_j$ for any $j \in R_k$.

Additive methods

- **Idea:** assume we have seen one more example of each value in S
- **Algorithm:** add 1 to each count and renormalize.

$$\theta_j^{Laplace} = \frac{n_j + 1}{n + m} \quad \text{for } j = 1 : m \tag{2}$$

- Can be used also with another value, $n_j^0 < 1$, in place of 1.
  Then, it is called **Bayesian mean smoothing** or **Dirichlet smothing** or **ELE**[1]
  Can be derived from Bayesian estimation, with the Dirichlet prior. In particular, we can take $n^0 = 1$, $n_j^0 = \frac{1}{m}$.

$$\theta_j^{Bayes} = \frac{n_j + n_j^0}{n + n_0} \quad \text{for } j = 1 : m \tag{3}$$

  The "fictitious sample size" $n^0 = \sum_{j=1}^m n_j^0$ reflects the strength of our belief about the θ_j's; if we choose all $n_j \propto \frac{1}{m}$, we say that we have an *uninformative prior*,

[1] In natural language processing.

Problems with aditive smoothing

- Reduces all estimates in the same proportion
- Does not distinguish between spread and concentrated distributions.
  - the unseen outcomes have the same probability no matter how the counts are distributed
-
- "Naive" method – DON'T USE IT

Ney-Essen discounting – tax and redistribute

- Let r = the number of distinct values observed

$$r \;=\; m - r_0$$

- **Idea**

  Tax substract 1 observation from every $n_j > 0$
  - i.e from each n_j that "can afford it"
  - total amount = r

  Red redistribute the total amount equally to all counts.

  This simple method works surprisingly well in practice.
- **Algorithm**

$$r \;=\; \sum_{j=1:m} \min(n_j, 1) \quad \text{total tax collected} \tag{4}$$

$$n_j^{NE} \;=\; \max(n_j - 1, 0) + r/m \quad \text{redistribute} \tag{5}$$

$$\theta_j^{NE} \;=\; \frac{n_j^{NE}}{n} \quad \text{estimate from new counts} \tag{6}$$

Algorithm can be generalized to any "tax amount" $\delta > 0$.

- Then, the total tax collected is $D \;=\; \sum_j \min(n_j, \delta)$
- The smoothed counts are $n_j^{NE} = \max(n_j - \delta, 0) + D/m$

Properties of NE smoothing

Flexibility

- treats outcomes with $n_j = 1$ and $n_j = 0$ the same
  Intuition: any outcome i with $n_j < \delta$ is a rare outcome and should be treated in the same way, no matter how many observations it actually has.
- For m large and r small
  - (probability mass is concentrated on a few values)
  - r small $\Rightarrow$ unobserved outcomes receive little probability
- For m large and r large
  - $r \approx m$ (large) $\Rightarrow$ unobserved outcomes get $n^{NE} \approx 1$
- For tax $\delta \neq 1$, note $D \leq \delta r$, redistributed mass $\frac{D}{m} \leq \delta \frac{r}{m}$

Witten-Bell discounting – probability of a new value

- **Idea:**
  - Look at the sequence $(x_1, \ldots x_n)$ as a binary process: either we observe a value of X that was observed before, or we observe a new one.
  - Assume that of m possible values r were observed (and $m - r$ unobserved)
  - Then the probability of observing a new value is $p_0 = \frac{r}{n}$.
  - Hence, set the probability of all unseen values of X to p_0. The other probabiliy estimates are renormalized accordingly.

$$\theta_j^{WB} = \begin{cases} \frac{n_j}{n}\frac{1}{1+p_0} = & \frac{n_j}{n+r} & n_j > 0 \\ \frac{1}{m-r}\frac{p_0}{1+p_0} = & \frac{1}{m-r}\frac{r}{n+r} & n_j = 0 \end{cases} \tag{7}$$

Witten-Bell makes sense only when some n_j counts are zero. If all $n_j > 0$ then W-B smoothing has undefined results.
WB smoothing has no parameter to choose (GOOD!)

Good-Turing – Predicting the type of the next outcome

- This method has many versions (you will see why). Powerful for large data sets.
- **First Idea**
  - Remember $r_k = \#\{j,\ n_j = k\}$ the counts of the counts. Naturally, $n = \sum_{k=1}^{\infty} k r_k$.
  - Outcome i is of **type** k if $n_j = k$. GT uses the data to estimate the probability of type k

$$p_k = \frac{k r_k}{n} \quad \text{for } k = 1 : n \tag{8}$$

- **Second Idea** is to use the probabilities $p_1, \ldots p_k \ldots$ to predict the **next** outcome
  - For example, what's the probability of seeing a new value?
    It must be equal to p_1, because this observation will have count $n_j = 1$ once it is observed.
  - Similarly, the probability of observing a type k outcome must be about p_{k+1}.
- **Third** There are r_k outcomes j in type k, hence the probability mass for each of these is $1/r_k$ of p_{k+1} which leads to (11).
- **Algorithm**

$$\text{if } n_j = k \quad \theta_j^{GT} = \frac{p_{k+1}}{r_k} = \frac{(k+1) r_{k+1}}{n r_k} \stackrel{def}{=} \frac{n_k^{GT}}{n} \qquad \text{with} \qquad n_j^{GT} = \frac{(k+1) r_{k+1}}{r_k} \tag{9}$$

In particular if $n_j = 0$

$$\theta_j^{GT} = \frac{p_1}{r_0} \tag{10}$$

- Remark GT transfers the probability mass of type $k+1$ to type k
- This implies that

$$n_j^{GT} r_k \ = \ (k+1) r_{k+1} \text{ if } n_j = k \tag{11}$$

Problems with Good-Turing

- When k is large, r_k is small and noisy.
  - Example The word "Jimmy" appears $n_{Jimmy} = 8196$ times in a corpus. But there may be no word that appears 8197 times. Then, $\theta^{GT}_{Jimmy} = 0$!
- Remedy: "smooth" the r_k values, i.e use (an estimate of) $E[r_k]$
  - Many proposals exist
  - A simple one is tois to use Good-Turing only for type 0, and to rescale the other θ^{ML} estimates down to ensure normalization.

$$\theta_j^{GT} = \begin{cases} \frac{p_1}{r_0} = \frac{r1}{nr_0} & \text{if } n_j = 0 \\ \theta_j^{ML}\left(1 - \frac{r_1}{n}\right) & \text{if } n_j > 0 \end{cases} \tag{12}$$

Comparison of the methods

Numerical values to exemplify the results: $n = 1000$, $m = 1000$, $r = 100$

Count n_j	**0**	**1**	$n_j \gg 1$
θ_j^{ML}	0	$\frac{1}{n} = \frac{1}{1000}$	$\frac{n_j}{1000}$
$\theta_j^{Laplace}$	$\frac{1}{n+m} = \frac{1}{2000}$	$\frac{2}{n+m} = \frac{1}{1000}$	$\frac{n_j+1}{n+m} = \frac{n_j+1}{2000}$
θ_j^{Bayes}, $n^0 = 1$, $n_j^0 = \frac{1}{m}$	$\frac{1}{m(n+1)} \approx \frac{1}{10^6}$	$\frac{1+1/m}{n+1} \approx \frac{1}{10^3}$	$\frac{n_j+1/m}{n+1} \approx \frac{n_j}{1000}$
θ_j^{NE}, $\delta = 1$	$\frac{r}{mn} = \frac{1}{10^4}$	$\frac{r}{mn} = \frac{1}{10^4}$	$\frac{n_j-1+r/m}{n} \approx \frac{n_j}{1000}$
θ_j^{WB}	$\frac{1}{m-r}\frac{r}{n+r} = \frac{1}{9900}$	$\frac{1}{n+r} = \frac{1}{1100}$	$\frac{n_j}{n+r} = \frac{n_j}{1100}$

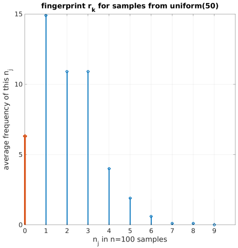
Remarks

- Laplace shrinks ML estimates of large probabilities by factor of 2. Too much! (because large θ_j^{ML} are close to their true values)
- Bayes (with uninformative prior) affects large θ_j^{ML} much less than small ones. Good
- Ney-Essen smooths more when r is larger; any n_j is affected by less than δ.
- Ney-Essen estimates of θ^{NE} for counts of 0 and 1 are equal to a fraction of $\frac{r}{m}$ (this grows with n as r grows with n).
- In Witten-Bell, the large θ_j^{ML} are shrunk depending on r, but independently of m. Proportional, bad
- ...but, if we overestimate m grossly, the overestimation will only affect the θ_j^{WB} for the 0 counts, but none of the θ_j^{WB} for the values observed. (true for NE as well).

Back-off or shrinkage – mixing with simpler models

(T B Written)

Ultimate test: which method is best?



Predict new data