

Language as a Stochastic Process

Lecturer: Xiaojin Zhu

jerryzhu@cs.wisc.edu

1 Basic Statistics for NLP

Pick an arbitrary letter x at random from any English text ever written. What is x ?

Random variable X : the random outcome of an experiment.

Probability distribution $P(X = x) = \theta_x$, for $x \in \{1, \dots, 27\}$ if our alphabet consists of $\{a, b, c, \dots, z, \sqcup\}$. Often use the probability vector $\theta = (\theta_1, \dots, \theta_{27})^\top$.

Basic properties: $0 \leq \theta_x \leq 1$, $\sum_x \theta_x = 1$.

Useful analogy of θ : a k -sided die. Here $k = 27$. If $k = 2$, a coin. A *fair* die/coin is when $\theta_1 = \dots = \theta_k = 1/k$, otherwise it is *biased*.

Sampling a letter x from the distribution θ ($x \sim \theta$):

1. create intervals of lengths $\theta_1, \dots, \theta_k$.
2. generate a uniform random number $r \in [0, 1]$ (most programming languages have a `random()` function for this).
3. find the interval r falls into, output the interval index.

The **Multinomial distribution** for k -sided die with probability vector θ , N throws, outcome counts n_1, \dots, n_k :

$$P(n_1, \dots, n_k | \theta) = \binom{N}{n_1 \dots n_k} \prod_{i=1}^k \theta_i^{n_i}. \quad (1)$$

Going back to language, if we ignore letter order, we can model the letters as draws from a multinomial distribution. The MLE of the parameter is the frequency of letters in the English language (not computable).

A **corpus** (*pl.* corpora) is a large, representative collection of text. See the Linguistic Data Consortium (LDC) for some examples.

The **likelihood function** is $P(\text{Data} | \theta)$. Here $\text{Data} = (n_1, \dots, n_k)$, and the likelihood takes the form of (1). Likelihood is *a function of θ* , and in general is not normalized: $\int P(\text{Data} | \theta) d\theta \neq 1$.

Conditional probability $P(X = x | H = h)$. Given the latest letter is h , what is the next letter x ?

How do we estimate conditional probabilities? Given a corpus with multiple sentences:

```
i am a student
i like this class
...
```

Let's add a special symbol "start-of-sentence" $\langle s \rangle$ to each sentence:

```
\langle s \rangle i am a student
```

$\langle s \rangle$ i like this class

$\langle s \rangle$...

and break each sentence into pairs of letters. Note we don't cross sentence boundaries, e.g., no (t $\langle s \rangle$):

$\langle \langle s \rangle$ i) (i \sqcup) (\sqcup a) (a m) ... (n t)

$\langle \langle s \rangle$ i) (i \sqcup) (\sqcup l) (l i) ... (s s)

...

Now our random variables x, h take value in $\{\langle s \rangle, a, \dots, z, \sqcup\}$. Let c_{hx} be the count of the pair (hx) in the corpus. We want to estimate the parameters $\theta = \{\theta_{hx} = P(x|h)\}$. Note this θ is much bigger! We can arrange it in a *transition matrix* with rows for h and columns for x .

Intuition: $P(x|h) = c_{hx} / \sum_{x'} c_{hx'}$. It turns out to be the MLE of θ . Since a sentence always starts with $\langle s \rangle$, the likelihood function $P(\text{Data}|\theta)$ is

$$P(i|\langle s \rangle)P(\sqcup|i) \dots P(t|n)P(i|\langle s \rangle) \dots \quad (2)$$

$$= \theta_{\langle s \rangle i} \theta_{i \sqcup} \dots \quad (3)$$

$$= \prod_{h,x} \theta_{hx}^{c_{hx}}, \quad (4)$$

with the constraints $0 \leq \theta_{hx} \leq 1, \sum_x \theta_{hx} = 1, \forall h$. One can solve the MLE with Lagrange multiplier as before.

The **joint probability** $P(x, h) = P(h, x)$ is the (much rarer event) that the two events both happening. The MLE is $P(x, h) = c_{hx} / \sum_{h,x} c_{hx}$.

The **marginal probability** is obtained by summing over some random variables. For example, $P(h) = \sum_x P(h, x)$. Relation:

$$P(x, h) = P(x|h)P(h). \quad (5)$$

Now, h and x do not have to be the same type of random variables. Consider this: randomly pick a word, and randomly pick a letter from the word. Let h be the length (number of letters) of the word, and x be the identity of the letter. It is perfectly fine to ask for $P(x = a|h = 2)$. What if you are told the letter is 'a', and you have to guess the length of the word?

The **Bayes rule** allows you to flip the conditional probability around:

$$P(x|h) = \frac{P(x, h)}{P(h)} = \frac{P(h|x)P(x)}{\sum_{x'} P(h|x')P(x')} = \frac{P(h|x)P(x)}{P(h)}. \quad (6)$$

2 Parameter Estimation (Statistics) = Learning a Model (Machine Learning)

Now given N draws from an unknown θ , and we observe the count histogram n_1, \dots, n_k , can we *estimate* θ ? In addition, how do we predict the next draw?

Example 1 If in $N = 10$ coin flips, we observe $n_1 = 4$ heads, and $n_2 = 6$ tails, what is θ ?

Intuition says $\theta = (0.4, 0.6)$. But in fact pretty much any θ could have generated the observed counts. How do we pick one? Should we pick one?

Parameter estimation and future prediction are two central problems of statistics, and machine learning.

2.1 The MLE Estimate

The **Maximum Likelihood Estimate (MLE)** is

$$\theta^{MLE} = \operatorname{argmax}_{\theta} P(\text{Data}|\theta). \quad (7)$$

Deriving the MLE of a multinomial (V is the same as k , c is the same as m):

$$\theta^{ML} = \arg \max_{\theta \in V\text{-simplex}} P(c_{1:V}|\theta) \quad (8)$$

$$= \arg \max_{\theta} \prod_{w=1}^V \theta_w^{c_w} \quad \text{multinomial definition} \quad (9)$$

$$= \arg \max_{\theta} \sum_{w=1}^V c_w \log \theta_w \quad \log() \text{ monotonic} \quad (10)$$

We are faced with the constrained optimization problem of finding $\theta_{1:V}$:

$$\max_{\theta_{1:V}} \sum_{w=1}^V c_w \log \theta_w \quad (11)$$

$$\text{subject to } \sum_{w=1}^V \theta_w = 1. \quad (12)$$

The general procedure to solve equality constrained optimization problems is the following: We introduce a scalar β called a *Lagrange multiplier* (one for each constraint), rewrite the equality constraint as $E(x) = 0$, and define a new Lagrangian function of the form $G(x, \beta) = F(x) - \beta E(x)$, where $F(x)$ is the original objective. Solve for the *unconstrained* optimization problem on G .

In our case, the Lagrangian is

$$\sum_{w=1}^V c_w \log \theta_w - \beta \left(\sum_{w=1}^V \theta_w - 1 \right) \quad (13)$$

After verifying that this is a concave function, we set the gradient (w.r.t. $\theta_{1:V}$ and β) to zero:

$$\frac{\partial}{\partial \theta_w} = \frac{c_w}{\theta_w} - \beta = 0 \quad (14)$$

$$\frac{\partial}{\partial \beta} = \sum_{w=1}^V \theta_w - 1 = 0, \quad (15)$$

which gives

$$\theta_w^{ML} = \frac{c_w}{\sum_{w=1}^V c_w} = \frac{c_w}{|C|}, \quad (16)$$

where $|C| = \sum_{w=1}^V c_w$ is the length of the corpus. It can be seen that the purpose of β is normalization. Therefore, in this case the MLE is simply the frequency estimate!

2.2 The MAP Estimate

The **Maximum A Posterior (MAP)** is

$$\theta^{MAP} = \operatorname{argmax}_{\theta} P(\theta|\text{Data}) = \operatorname{argmax}_{\theta} P(\theta)P(\text{Data}|\theta) \quad (17)$$

We need a **prior** distribution $P(\theta)$, which is usually taken to be the Dirichlet distribution since it is conjugate to multinomial:

$$P(\theta) = \operatorname{Dir}(\theta|\alpha_1, \dots, \alpha_k) = \frac{\Gamma(\sum_{i=1}^k \alpha_i)}{\prod_{i=1}^k \Gamma(\alpha_i)} \prod_{i=1}^k \theta_i^{\alpha_i-1} \quad (18)$$

The posterior is again a Dirichlet

$$P(\theta|\text{Data}) = \operatorname{Dir}(\theta|\alpha_1 + n_1, \dots, \alpha_k + n_k) = \frac{\Gamma(\sum_{i=1}^k \alpha_i + n_i)}{\prod_{i=1}^k \Gamma(\alpha_i + n_i)} \prod_{i=1}^k \theta_i^{\alpha_i+n_i-1} \quad (19)$$

The mode of the posterior (i.e., the MAP estimate) is

$$\theta_i = \frac{n_i + \alpha_i - 1}{N + \sum_{j=1}^k \alpha_j - k} \quad (20)$$

2.3 The Bayesian Approach

Keep the posterior distribution $P(\theta|\text{Data})$.

3 Predictive Distribution (Statistics) = Inference (Machine Learning)

For MLE and MAP, use the point estimate of θ to compute the multinomial.

For Bayesian, integrate out θ w.r.t. the posterior. This gives a multivariate Polya distribution, also known as the Dirichlet compound multinomial distribution. Let $\beta_i = \alpha_i + n_i$.

$$P(m_1 \dots m_k | \beta) = \int_{\theta} P(m_1 \dots m_k | \theta) p(\theta | \beta) d\theta \quad (21)$$

$$= \frac{m!}{\prod_i (m_i!)} \frac{\Gamma(\sum_i \beta_i)}{\Gamma(m + \sum_i \beta_i)} \prod_i \frac{\Gamma(m_i + \beta_i)}{\Gamma(\beta_i)} \quad (22)$$

where $m = \sum_i m_i$.

4 Bag of Word, tf.idf, and Cosine Similarity

A document d can be represented by a word count vector. The length of the vector is the vocabulary size. The i th element is the number of tokens for the i th word type. This is known as the Bag of word (BOW) representation, as it ignores word order. A variant is a vector of binary indicators for the presence of each word type.

A document d can also be represented by a $tf \cdot idf$ vector. tf is the *normalize term frequency*, i.e. word count vector scaled, so that the most frequent word type in this document has a value of 1:

$$tf_w = \frac{c(w, d)}{\max_v c(v, d)},$$

where $c(w, d)$ is the count of word w in d . idf is inverse document frequency. Let N be the number of documents in the collection. Let n_w be the number of documents in which word type w appears.

$$idf_w = \log \frac{N}{n_w}.$$

idf is an attempt to handle stopwords or near-stopwords: if a word (like “the”) appears in most documents, it is probably not very interesting. Since $n_w \approx N$, its idf will be close to 0. Finally

$$tf \cdot idf_w = tf_w \times idf_w.$$

Given two documents d, q in feature vector representation, one way to define how similar they are is the *cosine similarity*: q and d form an angle θ in the feature space, and

$$sim(d, q) = \cos(\theta) \quad (23)$$

$$= \frac{d^\top q}{\|d\| \cdot \|q\|} \quad (24)$$

$$= \frac{d^\top q}{\sqrt{d^\top d} \sqrt{q^\top q}}, \quad (25)$$

where the dot product

$$u^\top v = \sum_{i=1}^V u_i v_i.$$