Yale University
Department of Statistics and Data Science

Statistics 365/565

Issued: 05/02/2021 **Due:** 05/12/2021

Notes: This hw covers Bayes and word2vec.

Problem 1: Bayes Suppose that we take the Dirichlet distribution over the probabilities $p \sim \mathbf{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_n)$ for $\alpha_i > 0$. Recall that the Dirichlet probability density function takes the form

$$f(p_1, \ldots, p_n) = \frac{1}{B(\alpha)} \prod_{i=1}^{n} p_i^{\alpha_i - 1}$$

You just need to know that $B(\alpha) = \int_{p_1=0}^{1} \cdots \int_{p_n=0}^{1} \prod_{i=1}^{n} p_i^{\alpha_i - 1} \mathrm{d}p_1 \cdots \mathrm{d}p_n$.

Now suppose that you observe the random vector $Y \in \mathbb{N}^n$ which is a bag of words where the j^{th} coordinate specifies the number of times that word j appears in a document. We can define Y as

$$Y = \sum_{i=1}^{M} w_i$$

where w_i is the random vector such that exactly one coordinate is 1 while all others are 0 and $(w_i)_{(j)} = 1$ if word i is j. We take all w_i to be independent and we assume that the probability that $\mathbb{P}((w_i)_{(j)} = 1) = p_j$, again recalling that at any time word i can only be *one* specific word.

Actual problem: Compute the posterior distribution of the p given $Y = y$. That is compute the probability density

$$f(p_1 = v_1, \ldots, p_n = v_n | Y = y)$$

Recall that

$$\mathbb{P}(p_1 = v_1, \ldots, p_n = v_n | Y = y) = \frac{\mathbb{P}(Y = y | p_1 = v_1, \ldots, p_n = v_n) f(p_1 = v_1, \ldots, p_n = v_n)}{\mathbb{P}(Y = y)}$$

However, note that $\mathbb{P}(Y = y)$ does not depend on the numerical values $v_1, \ldots, v_n$. Therefore,

$$\mathbb{P}(p_1 = v_1, \ldots, p_n = v_n | Y = y) \propto \mathbb{P}(Y = y | p_1 = v_1, \ldots, p_n = v_n) f(p_1 = v_1, \ldots, p_n = v_n)$$

So to compute the true distribution you just need to calculate the normalizing constant

$$Z = \int_{v_1, \ldots, v_n} \mathbb{P}(Y = y | p_1 = v_1, \ldots, p_n = v_n) f(p_1 = v_1, \ldots, p_n = v_n) \mathbf{d}v_1 \cdots \mathbf{d}v_n$$

which of course is equal to $\mathbb{P}(Y = y)$. However, by identifying an appropriate pattern, you shouldn't need to compute the integral (nor can you).

Problem 2: word2vec as PCA The word2vec class of methods consist of two different approaches to learning word embeddings. Here we will focus on what's called the skip-gram method. See the word2vec Wikipedia page for more information.

The specifics of the skip-gram method depend on the notion of a *context*. Given a sequence of words $w_1, w_2, \ldots, w_T$, the contexts of the t-th word w_t are the words in a symmetric window around w_t. For example, with a context window of size two, the contexts of the word *embeddings* in the sentence

I $\underbrace{\text{love word}}$ embeddings $\underbrace{\text{so much}}$.

are *love, word, so,* and *much*. The word-context pairs (w, c) associated with the word *embeddings* are therefore (embeddings, love), (embeddings, word), (embeddings, so), and (embeddings, much).

Under this definition of context, both words and contexts are drawn from the same vocabulary V. Let D denote the set of observed word-context pairs in a given corpus. Let $\#(w, c)$ denote the number of times the pair (w, c) appears in D, and let $\#(w) = \sum_{c' \in V} \#(w, c')$ denote the number of times w appears in D. Similarly, $\#(c)$ is the number of times the context c occurs in D.

Consider a word-context pair (w, c) and let $\mathcal{O}$ denote a random variable whose value is one if the pair (w, c) was observed in the corpus, and zero otherwise. In the skip-gram approach, this probability is modeled as:

$$\mathbb{P}(\mathcal{O} = 1 \mid w, c) = \sigma(v_w^T v_c) = \frac{1}{1 + e^{-v_w^T v_c}}$$

where σ denotes the logistic function, and $v_w, v_c \in \mathbb{R}^d$ are d-dimensional word embeddings for the word w and the context word c, respectively.

The skip-gram method aims to learn embedding vectors to maximize $\mathbb{P}(\mathcal{O} = 1 \mid w, c)$ for pairs (w, c) found in the corpus, while minimizing $\mathbb{P}(\mathcal{O} = 1 \mid w, c)$ for k randomly sampled contexts c for each word. Explicitly, the method attempts to optimize the objective function

$$\ell = \sum_{w,c} \ell(w, c)$$

where the local objective for a specific (w, c) pair is

$$\ell(w, c) = \#(w, c) \log\left(\sigma(v_w^T v_c)\right) + k \cdot \#(w) \frac{\#(c)}{|D|} \log\left(\sigma(-v_w^T v_c)\right).$$

The parameter k corresponds to the number of random contexts sampled for each word w. (Note that randomly sampling a context is likely to give an unobserved context c for w.)

Show that embeddings obtained in this way are equivalent to those obtained by factorizing a specific matrix, through the following steps:

a) Define $x = v_w^T v_c$. Show that a stationary point $\partial \ell / \partial x = 0$ of the **local** objective satisfies

$$x = \log\left(\frac{\#(w, c) \cdot |D|}{\#(w) \cdot \#(c)}\right) - \log k.$$

b) Explain how this motivates constructing embeddings $v_w \in \mathbb{R}^d$ as the rank-d SVD of a $|V| \times |V|$ matrix M. What is this matrix?

See the original paper for more details.

https://papers.nips.cc/paper/2014/file/feab05aa91085b7a8012516bc3533958-Paper.pdf