CSE 446: Machine Learning **Lecture**

Intro to Probabilistic Graphical Models/Latent variable models

Instructor: Sham Kakade

1 Review: Joint distributions and Bayes Rule

Suppose we have two random variables Z_1 and Z_2, with joint distribution $\Pr(Z_1, Z_2)$. Recall that:

$$\Pr(Z_1, Z_2) = \Pr(Z_1)\Pr(Z_2|Z_1)$$

This implies "Bayes rule":

$$\Pr(Z_2|Z_1) = Pr(Z_1|Z_2)Pr(Z_2)/\Pr(Z_1)$$

Also, for any T random variables, $Z_1, Z_2, \ldots Z_T$, with joint distribution $\Pr(Z_1, Z_2, \ldots Z_T)$. Then:

$$\Pr(Z_1, Z_2, \ldots, Z_T) = \Pr(Z_1)\Pr(Z_2|Z_1)\Pr(Z_3|Z_2, Z_1)\ldots\Pr(Z_t|Z_{t-1}, \ldots, Z_1)\ldots\Pr(Z_T|Z_{T-1}, \ldots, Z_1)\,.$$

The above is sometimes referred to as the chain rule of probabilities.

2 Basic idea of 'generative models'

We are now going to specify the method in which we believe our data are generated. This does not tell us how to learn the parameters of the model. However, specifying these procedures are helpful abstractions as they then give us a way to answer questions such as: what are the document groupings? How dow decided upon what is a good 'rule' to use to group our documents? The following approach allows us to address these questions in a principled (and general) approach.

For example, let us view each document as datapoint. And let us view each document as being represented by the word counts in the document. So each datapoint is just a collection of word counts (suppose we have M documents and each document is specified by a big vector of word counts). So how should we group our documents together?

Before we can answer the question, let us take a different viewpoint. For now, let us just specify a procedure for how our documents are generated; a probabilistic generative model is an underlying model of how our data are created. In what follows, we will consider a simple 'single topic' case, we assumed that each datapoint/document has a hidden topic associated with it. And that the words we observed were generated under a distribution over words implied by the topic. The learning question (next class!) is how we figure out the topics and (soft) document assignments given our data, by using our modeling assumptions.

These notes just specify a few generative models.

3 Common generative models

3.1 Mixture of Gaussians

Random variables: a "hidden" cluster $i \in \{1 \ldots k\}$ and a vector $x \in \mathbb{R}^d$.

Parameters: “mixing weights” $\pi_i = \Pr(\text{topic} = i)$, means: $\mu_1 \dots \mu_j$, noise covariance matrices $\Sigma_1, \Sigma_2, \dots \Sigma_k$

The Generative model for a datapoint:

1. sample a cluster i, which has probability π_i
2. observe x, where x is the mean μ_i corrupted with Gaussian noise:

   $$x = \mu_i + \eta$$

   where η has a multivariate normal distribution, $N(0, \Sigma_i)$.

3.2 “Bag of words” model: a (single) topic model

Suppose every document has T words.

Random variables: a “hidden” topic $i \in \{1 \dots k\}$ and a T-word outcomes $w_1, w_2, \dots w_T$ which take on some discrete values.

Parameters: the “mixing weights” $\pi_i = \Pr(\text{topic} = i)$, the “topics” $b_{wi} = \Pr(\text{word} = w | \text{topic} = i)$

The generative model for an T word “document”, where every document is only about one topic.

1. sample a “topic” i, which has probability π_i
2. gererate T words $w_1, w_2, \dots w_T$, independently. in particular, we choose word w_t as the t-th word with probability $b_{w_t i}$.

Note this generative model ignores the word order, so it is not a particularly faithful generative model.

Due to the ’graph’ (i.e. the conditional independencies implied by the generative model procedure), we can write the *joint* probability of the outcome topic i occurring with a document containing the words $w_1, w_2, \dots w_T$ as:

$$\begin{aligned}\Pr(\text{topic} = i \text{ and } w_1, w_2, \dots w_T) &= \Pr(\text{topic} = i) \Pr(w_1, w_2, \dots w_T | \text{topic} = i) \\ &= \Pr(\text{topic} = i) \Pr(w_1 | \text{topic} = i) \Pr(w_2 | \text{topic} = i) \Pr(w_T | \text{topic} = i) \\ &= \pi_i b_{w_1 i} b_{w_2 i} \dots b_{w_T i}\end{aligned}$$

where the second to last step follows due to the fact that the words are generated independently given the topic i.

3.2.1 Inference

Suppose we were given a document with $w_1, w_2, \dots w_T$. One *inference* question would be what is the probability the underlying topic is i. By Bayes rule, we have:

$$\begin{aligned}\Pr(\text{topic} = i | w_1, w_2, \dots w_T) &= \frac{1}{\Pr(w_1, w_2, \dots w_T)} \Pr(\text{topic} = i \text{ and } w_1, w_2, \dots w_T) \\ &= \frac{1}{Z} \pi_i b_{w_1 i} b_{w_2 i} \dots b_{w_T i}\end{aligned}$$

where Z is a number chosen so that the probabilities sum to 1. Critically, note that Z is not a function of i.

3.2.2 LDA: latent Dirichlet allocation

This is a popular model which allows documents to contain more than one topic.

3.3 Hidden Markov models

Random variables: a “hidden” state sequence $z_1, z_2, \ldots z_T$ (which take on some discrete values in some ’hidden state space’) and T-discrete sequential outcomes $w_1, w_2, \ldots w_T$. Suppose each z_i can take one of k outcomes and each w_i can take one of d outcomes.

Parameters: π_i, $A_{ji} = \Pr(z_{n+1} = j | z_t = i)$, $b_{mi} = \Pr(x_n = m | z_t = i)$

The Generative model for an t word “document”: for each time t,

1. sample a “hidden” state sequence z_{n+1}, using only the previous outcome z_t. The sampling is determined solely by the parameters $\{A_{ji}\}$.
2. Sample w_{n+1} using only z_{n+1}. This sampling is based only on the probabilities $\{b_{mi}\}$.

Due to the ’graph’ (i.e. the conditional independencies implied by the generative model procedure), we can write the joint probability of the hidden state sequence $z_1, z_2, \ldots z_T$ and the word sequence $w_1, w_2, \ldots w_T$ as:

$$\Pr(z_1, z_2, \ldots z_T \text{ and } w_1, w_2, \ldots w_T) = b_{w_1 z_1} A_{z_2 z_1} b_{w_2 z_2} A_{z_3 z_2} \ldots b_{w_T z_T}$$

One inference question would be to determine the probability that hidden state is $z_t = j$ given some observed sequence $w_1, w_2, \ldots w_T$, i.e.

$$\Pr(z_t = j | w_1, w_2, \ldots w_T)$$

Naively, this computation might look difficult. However, this can be done in a computationally efficiently manner using the Baum-Welch algorithm, sometimes known as the “Forward-Backward” algorithm.