The Naive Bayes Model, Maximum-Likelihood Estimation, and the EM Algorithm

Michael Collins

1 Introduction

This note covers the following topics:

- The Naive Bayes model for classification (with text classification as a specific example).
- The derivation of maximum-likelihood (ML) estimates for the Naive Bayes model, in the simple case where the underlying labels are observed in the training data.
- The EM algorithm for parameter estimation in Naive Bayes models, in the case where labels are missing from the training examples.
- The EM algorithm in general form, including a derivation of some of its convergence properties.

We will use the Naive Bayes model throughout this note, as a simple model where we can derive the EM algorithm. Later in the class we will consider EM for parameter estimation of more complex models, for example hidden Markov models and probabilistic context-free grammars.

2 The Naive Bayes Model for Classification

This section describes a model for binary classification, *Naive Bayes*. Naive Bayes is a simple but important probabilistic model. It will be used as a running example in this note. In particular, we will first consider maximum-likelihood estimation in the case where the data is “fully observed”; we will then consider the expectation maximization (EM) algorithm for the case where the data is “partially observed”, in the sense that the labels for examples are missing.

The setting is as follows. Assume we have some training set $(\underline{x}^{(i)}, y^{(i)})$ for $i = 1 \ldots n$, where each $\underline{x}^{(i)}$ is a vector, and each $y^{(i)}$ is in $\{1, 2, \ldots, k\}$. Here k is an integer specifying the number of classes in the problem. This is a *multiclass* classification problem, where the task is to map each input vector $\underline{x}$ to a label y that can take any one of k possible values. (For the special case of $k = 2$ we have a binary classification problem.)

We will assume throughout that each vector $\underline{x}$ is in the set $\{-1, +1\}^d$ for some integer d specifying the number of "features" in the model. In other words, each component x_j for $j = 1 \ldots d$ can take one of two possible values.

As one example motivating this setting, consider the problem of classifying documents into k different categories (for example $y = 1$ might correspond to a *sports* category, $y = 2$ might correspond to a *music* category, $y = 3$ might correspond to a *current affairs* category, and so on). The label $y^{(i)}$ represents the category of the i'th document in the collection. Each component $x_j^{(i)}$ for $j = 1 \ldots d$ might represent the presence or absence of a particular word. For example we might define $x_1^{(i)}$ to be $+1$ if the i'th document contains the word *Giants*, or -1 otherwise; $x_2^{(i)}$ to be $+1$ if the i'th document contains the word *Obama*, or -1 otherwise; and so on.

The Naive Bayes model is then derived as follows. We assume random variables Y and $X_1 \ldots X_d$ corresponding to the label y and the vector components $x_1, x_2, \ldots, x_d$. Our task will be to model the joint probability

$$P(Y = y, X_1 = x_1, X_2 = x_2, \ldots X_d = x_d)$$

for any label y paired with attribute values $x_1 \ldots x_d$. A key idea in the NB model is the following assumption:

$$\begin{aligned} & P(Y = y, X_1 = x_1, X_2 = x_2, \ldots X_d = x_d) \\ = \quad & P(Y = y) \prod_{j=1}^{d} P(X_j = x_j | Y = y) \end{aligned} \qquad (1)$$

This equality is derived using independence assumptions, as follows. First, by the chain rule, the following identity is exact (any joint distribution over $Y, X_1 \ldots X_d$ can be factored in this way):

$$\begin{aligned} & P(Y = y, X_1 = x_1, X_2 = x_2, \ldots X_d = x_d) \\ = \quad & P(Y = y) \times P(X_1 = x_1, X_2 = x_2, \ldots X_d = x_d | Y = y) \end{aligned}$$

Next, we deal with the second term as follows:

$$P(X_1 = x_1, X_2 = x_2, \ldots X_d = x_d | Y = y)$$

$$
\begin{aligned}
&= \prod_{j=1}^{d} P(X_j = x_j | X_1 = x_1, X_2 = x_2, \ldots X_{j-1} = x_{j-1}, Y = y) \\
&= \prod_{j=1}^{d} P(X_j = x_j | Y = y)
\end{aligned}
$$

The first equality is again exact, by the chain rule. The second equality follows by an independence assumption, namely that for all $j = 1 \ldots d$, the value for the random variable X_j is independent of all other attribute values, $X_{j'}$ for $j' \neq j$, when conditioned on the identity of the label Y. This is the *Naive Bayes* assumption. It is naive, in the sense that it is a relatively strong assumption. It is, however, a very useful assumption, in that it dramatically reduces the number of parameters in the model, while still leading to a model that can be quite effective in practice.

Following Eq. 1, the NB model has two types of parameters: $q(y)$ for $y \in \{1 \ldots k\}$, with

$$P(Y = y) = q(y)$$

and $q_j(x|y)$ for $j \in \{1 \ldots d\}$, $x \in \{-1, +1\}$, $y \in \{1 \ldots k\}$, with

$$P(X_j = x | Y = y) = q_j(x|y)$$

We then have

$$p(y, x_1 \ldots x_d) = q(y) \prod_{j=1}^{d} q_j(x_j|y)$$

To summarize, we give the following definition:

Definition 1 (Naive Bayes (NB) Model) *A NB model consists of an integer k specifying the number of possible labels, an integer d specifying the number of attributes, and in addition the following parameters:*

- *A parameter*

  $$q(y)$$

  for any $y \in \{1 \ldots k\}$. The parameter $q(y)$ can be interpreted as the probability of seeing the label y. We have the constraints $q(y) \geq 0$ and $\sum_{y=1}^{k} q(y) = 1$.

- *A parameter*

  $$q_j(x|y)$$

  for any $j \in \{1 \ldots d\}$, $x \in \{-1, +1\}$, $y \in \{1 \ldots k\}$. The value for $q_j(x|y)$ can be interpreted as the probability of attribute j taking value x, conditioned on the underlying label being y. We have the constraints that $q_j(x|y) \geq 0$, and for all y, j, $\sum_{x \in \{-1,+1\}} q_j(x|y) = 1$.

We then define the probability for any $y, x_1 \ldots x_d$ as

$$p(y, x_1 \ldots x_d) = q(y) \prod_{j=1}^{d} q_j(x_j|y) \qquad \square$$

The next section describes how the parameters can be estimated from training examples. Once the parameters have been estimated, given a new test example $\underline{x} = \langle x_1, x_2, \ldots, x_d \rangle$, the output of the NB classifier is

$$\arg \max_{y \in \{1 \ldots k\}} p(y, x_1 \ldots x_d) = \arg \max_{y \in \{1 \ldots k\}} \left(q(y) \prod_{j=1}^{d} q_j(x_j|y) \right)$$

3 Maximum-Likelihood estimates for the Naive Bayes Model

We now consider how the parameters $q(y)$ and $q_j(x|y)$ can be estimated from data. In particular, we will describe the *maximum-likelihood estimates*. We first state the form of the estimates, and then go into some detail about how the estimates are derived.

Our training sample consists of examples $(\underline{x}^{(i)}, y^{(i)})$ for $i = 1 \ldots n$. Recall that each $\underline{x}^{(i)}$ is a d-dimensional vector. We write $x_j^{(i)}$ for the value of the j'th component of $\underline{x}^{(i)}$; $x_j^{(i)}$ can take values -1 or $+1$.

Given these definitions, the maximum-likelihood estimates for $q(y)$ for $y \in \{1 \ldots k\}$ take the following form:

$$q(y) = \frac{\sum_{i=1}^{n} [[y^{(i)} = y]]}{n} = \frac{\text{count}(y)}{n} \qquad (2)$$

Here we define $[[y^{(i)} = y]]$ to be 1 if $y^{(i)} = y$, 0 otherwise. Hence $\sum_{i=1}^{n} [[y^{(i)} = y]] = \text{count}(y)$ is simply the number of times that the label y is seen in the training set.

Similarly, the ML estimates for the $q_j(x|y)$ parameters (for all $y \in \{1 \ldots k\}$, for all $x \in \{-1, +1\}$, for all $j \in \{1 \ldots d\}$) take the following form:

$$q_j(x|y) = \frac{\sum_{i=1}^{n} [[y^{(i)} = y \text{ and } x_j^{(i)} = x]]}{\sum_{i=1}^{n} [[y^{(i)} = y]]} = \frac{\text{count}_j(x|y)}{\text{count}(y)} \qquad (3)$$

where

$$\text{count}_j(x|y) = \sum_{i=1}^{n} [[y^{(i)} = y \text{ and } x_j^{(i)} = x]]$$

This is a very natural estimate: we simply count the number of times label y is seen in conjunction with x_j taking value x; count the number of times the label y is seen in total; then take the ratio of these two terms.

4 Deriving the Maximum-Likelihood Estimates

4.1 Definition of the ML Estimation Problem

We now describe how the ML estimates in Eqs. 2 and 3 are derived. Given the training set $(x^{(i)}, y^{(i)})$ for $i = 1 \ldots n$, the log-likelihood function is

$$\begin{aligned}
L(\underline{\theta}) &= \sum_{i=1}^{n} \log p(x^{(i)}, y^{(i)}) \\
&= \sum_{i=1}^{n} \log \left(q(y^{(i)}) \prod_{j=1}^{d} q_j(x_j^{(i)}|y^{(i)}) \right) \\
&= \sum_{i=1}^{n} \log q(y^{(i)}) + \sum_{i=1}^{n} \log \left(\prod_{j=1}^{d} q_j(x_j^{(i)}|y^{(i)}) \right) \\
&= \sum_{i=1}^{n} \log q(y^{(i)}) + \sum_{i=1}^{n} \sum_{j=1}^{d} \log q_j(x_j^{(i)}|y^{(i)}) \qquad (4)
\end{aligned}$$

Here for convenience we use $\underline{\theta}$ to refer to a parameter vector consisting of values for all parameters $q(y)$ and $q_j(x|y)$ in the model. The log-likelihood is a function of the parameter values, and the training examples. The final two equalities follow from the usual property that $\log(a \times b) = \log a + \log b$.

As usual, the log-likelihood function $L(\underline{\theta})$ can be interpreted as a measure of how well the parameter values fit the training example. In ML estimation we seek the parameter values that maximize $L(\underline{\theta})$.

The maximum-likelihood problem is the following:

Definition 2 (ML Estimates for Naive Bayes Models) *Assume a training set* $(x^{(i)}, y^{(i)})$ *for* $i \in \{1 \ldots n\}$. *The maximum-likelihood estimates are then the parameter values* $q(y)$ *for* $y \in \{1 \ldots k\}$, $q_j(x|y)$ *for* $j \in \{1 \ldots d\}$, $y \in \{1 \ldots k\}$, $x \in \{-1, +1\}$ *that maximize*

$$L(\underline{\theta}) = \sum_{i=1}^{n} \log q(y^{(i)}) + \sum_{i=1}^{n} \sum_{j=1}^{d} \log q_j(x_j^{(i)}|y^{(i)})$$

subject to the following constraints:

1. $q(y) \geq 0$ *for all* $y \in \{1 \ldots k\}$. $\sum_{y=1}^{k} q(y) = 1$.
2. *For all* y, j, x, $q_j(x|y) \geq 0$. *For all* $y \in \{1 \ldots k\}$, *for all* $j \in \{1 \ldots d\}$,

$$\sum_{x \in \{-1,+1\}} q_j(x|y) = 1$$

□

A crucial result is then the following:

Theorem 1 *The ML estimates for Naive Bayes models (see definition 2) take the form*

$$q(y) = \frac{\sum_{i=1}^{n} [[y^{(i)} = y]]}{n} = \frac{\textit{count}(y)}{n}$$

and

$$q_j(x|y) = \frac{\sum_{i=1}^{n} [[y^{(i)} = y \textit{ and } x_j^{(i)} = x]]}{\sum_{i=1}^{n} [[y^{(i)} = y]]} = \frac{\textit{count}_j(x|y)}{\textit{count}(y)}$$

I.e., they take the form given in Eqs. 2 and 3.

The remainder of this section proves the result in theorem 1. We first consider a simple but crucial result, concerning ML estimation for multinomial distributions. We then see how this result leads directly to a proof of theorem 1.

4.2 Maximum-likelihood Estimation for Multinomial Distributions

Consider the following setting. We have some finite set $\mathcal{Y}$. A *distribution* over the set $\mathcal{Y}$ is a vector q with components q_y for each $y \in \mathcal{Y}$, corresponding to the probability of seeing element y. We define $\mathcal{P}_{\mathcal{Y}}$ to be the set of all distributions over the set $\mathcal{Y}$: that is,

$$\mathcal{P}_{\mathcal{Y}} = \{q \in \mathbb{R}^{|\mathcal{Y}|} : \forall y \in \mathcal{Y}, q_y \geq 0; \ \sum_{y \in \mathcal{Y}} q_y = 1\}$$

In addition, assume that we have some vector c with components c_y for each $y \in \mathcal{Y}$. We will assume that each $c_y \geq 0$. In many cases c_y will correspond to some "count" taken from data: specifically the number of times that we see element y. We also assume that there is at least one $y \in \mathcal{P}_{\mathcal{Y}}$ such that $c_y > 0$ (i.e., such that c_y is strictly positive).

We then state the following optimization problem:

Definition 3 (ML estimation problem for multinomials) *The input to the problem is a finite set $\mathcal{Y}$, and a weight $c_y \geq 0$ for each $y \in \mathcal{Y}$. The output from the problem is the distribution q^* that solves the following maximization problem:*

$$q^* = \arg\max_{q \in \mathcal{P}_{\mathcal{Y}}} \sum_{y \in \mathcal{Y}} c_y \log q_y$$

Thus the optimal vector q^* is a distribution (it is a member of the set $\mathcal{P}_\mathcal{Y}$), and in addition it maximizes the function $\sum_{y \in \mathcal{Y}} c_y \log q_y$.

We give a theorem that gives a very simple (and intuitive) form for q^*:

Theorem 2 *Consider the problem in definition 3. The vector q^* has components*

$$q^*_y = \frac{c_y}{N}$$

for all $y \in \mathcal{Y}$, where $N = \sum_{y \in \mathcal{Y}} c_y$.

Proof: To recap, our goal is to maximize the function

$$\sum_{y \in \mathcal{Y}} c_y \log q_y$$

subject to the constraints $q_y \geq 0$ and $\sum_{y \in \mathcal{Y}} q_y = 1$. For simplicity we will assume throughout that $c_y > 0$ for all y.[1]

We will introduce a single Lagrange multiplier $\lambda \in \mathbb{R}$ corresponding to the constraint that $\sum_{y \in \mathcal{Y}} q_y = 1$. The Lagrangian is then

$$g(\lambda, q) = \sum_{y \in \mathcal{Y}} c_y \log q_y - \lambda \left(\sum_{y \in \mathcal{Y}} q_y - 1 \right)$$

By the usual theory of Lagrange multipliers, the solution q^*_y to the maximization problem must satisfy the conditions

$$\frac{d}{dq_y} g(\lambda, q) = 0$$

for all y, and

$$\sum_{y \in \mathcal{Y}} q_y = 1 \tag{5}$$

Differentiating with respect to q_y gives

$$\frac{d}{dq_y} g(\lambda, q) = \frac{c_y}{q_y} - \lambda$$

Setting this derivative to zero gives

$$q_y = \frac{c_y}{\lambda} \tag{6}$$

[1] In a full proof it can be shown that for any y such that $c_y = 0$, we must have $q_y = 0$; we can then consider the problem of maximizing $\sum_{y \in \mathcal{Y}'} c_y \log q_y$ subject to $\sum_{y \in \mathcal{Y}'} q_y = 1$, where $\mathcal{Y}' = \{y \in \mathcal{Y} : c_y > 0\}$.

Combining Eqs. 6 and 5 gives

$$q_y = \frac{c_y}{\sum_{y \in \mathcal{Y}} c_y}$$

□

The proof of theorem 1 follows directly from this result. See section A for a full proof.

5 The EM Algorithm for Naive Bayes

Now consider the following setting. We have a training set consisting of vectors $x^{(i)}$ for $i = 1 \dots n$. As before, each $x^{(i)}$ is a vector with components $x_j^{(i)}$ for $j \in \{1 \dots d\}$, where each component can take either the value -1 or $+1$. In other words, our training set *does not have any labels*. Can we still estimate the parameters of the model?

As a concrete example, consider a very simple text classification problem where the vector $\underline{x}$ representing a document has the following four components (i.e., $d = 4$):

$$\begin{aligned}
x_1 &= +1 \text{ if the document contains the word } \textit{Obama}, -1 \text{ otherwise} \\
x_2 &= +1 \text{ if the document contains the word } \textit{McCain}, -1 \text{ otherwise} \\
x_3 &= +1 \text{ if the document contains the word } \textit{Giants}, -1 \text{ otherwise} \\
x_4 &= +1 \text{ if the document contains the word } \textit{Patriots}, -1 \text{ otherwise}
\end{aligned}$$

In addition, we assume that our training data consists of the following examples:

$$\begin{aligned}
\underline{x}^{(1)} &= \langle +1, +1, -1, -1 \rangle \\
\underline{x}^{(2)} &= \langle -1, -1, +1, +1 \rangle \\
\underline{x}^{(3)} &= \langle +1, +1, -1, -1 \rangle \\
\underline{x}^{(4)} &= \langle -1, -1, +1, +1 \rangle \\
\underline{x}^{(5)} &= \langle -1, -1, +1, +1 \rangle
\end{aligned}$$

Intuitively, this data might arise because documents 1 and 3 are about politics (and thus include words like *Obama* or *McCain*, which refer to politicians), and documents $2, 4$ and 5 are about sports (and thus include words like *Giants*, or *Patriots*, which refer to sports teams).

For this data, a good setting of the parameters of a NB model might be as follows (we will soon formalize exactly what it means for the parameter values to be a "good" fit to the data):

$$q(1) = \frac{2}{5}; \quad q(2) = \frac{3}{5}; \tag{7}$$

$$q_1(+1|1) = 1; \quad q_2(+1|1) = 1; \quad q_3(+1|1) = 0; \quad q_4(+1|1) = 0; \tag{8}$$

$$q_1(+1|2) = 0; \quad q_2(+1|2) = 0; \quad q_3(+1|2) = 1; \quad q_4(+1|2) = 1 \tag{9}$$

Thus there are two classes of documents. There is a probability of $2/5$ of seeing class 1, versus a probability of $3/5$ of seeing class 2. Given class 1, we have the vector $\underline{x} = \langle +1, +1, -1, -1 \rangle$ with probability 1; conversely, given class 2, we have the vector $\underline{x} = \langle -1, -1, +1, +1 \rangle$ with probability 1.

Remark. Note that an equally good fit to the data would be the parameter values

$$q(2) = \frac{2}{5}; \quad q(1) = \frac{3}{5};$$

$$q_1(+1|2) = 1; \quad q_2(+1|2) = 1; \quad q_3(+1|2) = 0; \quad q_4(+1|2) = 0;$$

$$q_1(+1|1) = 0; \quad q_2(+1|1) = 0; \quad q_3(+1|1) = 1; \quad q_4(+1|1) = 1$$

Here we have just switched the meaning of classes 1 and 2, and permuted all of the associated probabilities. Cases like this, where symmetries mean that multiple models give the same fit to the data, are common in the EM setting.

5.1 The Maximum-Likelihood Problem for Naive Bayes with Missing Labels

We now describe the parameter estimation method for Naive Bayes when the labels $y^{(i)}$ for $i \in \{1 \ldots n\}$ are missing. The first key insight is that for any example $\underline{x}$, the probability of that example under a NB model can be calculated by marginalizing out the labels:

$$p(\underline{x}) = \sum_{y=1}^{k} p(\underline{x}, y) = \sum_{y=1}^{k} \left(q(y) \prod_{j=1}^{d} q_j(x_j|y) \right)$$

Given this observation, we can define a log-likelihood function as follows. The log-likelihood function is again a measure of how well the parameter values fit the training examples. Given the training set $(x^{(i)})$ for $i = 1 \dots n$, the log-likelihood function (we again use $\underline{\theta}$ to refer to the full set of parameters in the model) is

$$\begin{aligned} L(\underline{\theta}) &= \sum_{i=1}^{n} \log p(x^{(i)}) \\ &= \sum_{i=1}^{n} \log \sum_{y=1}^{k} \left(q(y) \prod_{j=1}^{d} q_j(x_j^{(i)}|y) \right) \end{aligned}$$

In ML estimation we seek the parameter values that maximize $L(\underline{\theta})$. This leads to the following problem definition:

Definition 4 (ML Estimates for Naive Bayes Models with Missing Labels) *Assume a training set $(x^{(i)})$ for $i \in \{1 \dots n\}$. The maximum-likelihood estimates are then the parameter values $q(y)$ for $y \in \{1 \dots k\}$, $q_j(x|y)$ for $j\{1 \dots d\}$, $y \in \{1 \dots k\}$, $x \in \{-1, +1\}$ that maximize*

$$L(\underline{\theta}) = \sum_{i=1}^{n} \log \sum_{y=1}^{k} \left(q(y) \prod_{j=1}^{d} q_j(x_j^{(i)}|y) \right) \tag{10}$$

subject to the following constraints:

1. *$q(y) \geq 0$ for all $y \in \{1 \dots k\}$. $\sum_{y=1}^{k} q(y) = 1$.*
2. *For all y, j, x, $q_j(x|y) \geq 0$. For all $y \in \{1 \dots k\}$, for all $j \in \{1 \dots d\}$,*

$$\sum_{x \in \{-1,+1\}} q_j(x|y) = 1$$

□

Given this problem definition, we are left with the following questions:

How are the ML estimates justified? In a formal sense, the following result holds. Assume that the training examples $x^{(i)}$ for $i = 1 \dots n$ are actually i.i.d. samples from a distribution specified by a Naive Bayes model. Equivalently, we assume that the training samples are drawn from a process that first generates an $(\underline{x}, y)$ pair, then deletes the value of the label y, leaving us with only the observations $\underline{x}$. Then it can be shown that the ML estimates are *consistent*, in that as the number of training samples n increases, the parameters will converge to the true values of the underlying Naive Bayes model.[2]

From a more practical point of view, in practice the ML estimates will often uncover useful patterns in the training examples. For example, it can be verified that the parameter values in Eqs. 7, 8 and 9 do indeed maximize the log-likelihood function given the documents given in the example.

How are the ML estimates useful? Assuming that we have an algorithm that calculates the maximum-likelihood estimates, how are these estimates useful? In practice, there are several scenarios in which the maximum-likelihood estimates will be useful. The parameter estimates find useful patterns in the data: for example, in the context of text classification they can find a useful partition of documents in naturally occurring classes. In particular, once the parameters have been estimated, for any document $\underline{x}$, for any class $y \in \{1 \dots k\}$, we can calculate the conditional probability

$$p(y|\underline{x}) = \frac{p(\underline{x}, y)}{\sum_{y=1}^{k} p(\underline{x}, y)}$$

under the model. This allows us to calculate the probability of document $\underline{x}$ falling into cluster y: if we required a hard partition of the documents into k different classes, we could take the highest probability label,

$$\arg \max_{y \in \{1 \dots k\}} p(y|\underline{x})$$

There are many other uses of EM, which we will see later in the course.

Given a training set, how can we calculate the ML estimates? The final question concerns calculation of the ML estimates. To recap, the function that we would like to maximize (see Eq. 10) is

$$L(\underline{\theta}) = \sum_{i=1}^{n} \log \sum_{y=1}^{k} \left(q(y) \prod_{j=1}^{d} q_j(x_j^{(i)}|y) \right)$$

note the contrast with the regular ML problem (see Eq. 4), where we have labels $y^{(i)}$, and the function we wish to optimize is

$$L(\underline{\theta}) = \sum_{i=1}^{n} \log \left(q(y^{(i)}) \prod_{j=1}^{d} q_j(x_j^{(i)}|y^{(i)}) \right)$$

[2] Up to symmetries in the model; for example, in the text classification example given earlier with *Obama, McCain, Giants, Patriots*, either of the two parameter settings given would be recovered.

The two functions are similar, but crucially *the new definition of* $L(\underline{\theta})$ *has an additional sum over* $y = 1 \ldots k$, which appears within the $\log$. This sum makes optimization of $L(\underline{\theta})$ hard (in contrast to the definition when the labels are observed).

The next section describes the expectation-maximization (EM) algorithm for calculation of the ML estimates. Because of the difficulty of optimizing the new definition of $L(\underline{\theta})$, the algorithm will have relatively weak guarantees, in the sense that it will only be guaranteed to reach a *local optimum* of the function $L(\underline{\theta})$. The EM algorithm is, however, widely used, and can be very effective in practice.

5.2 The EM Algorithm for Naive Bayes Models

The EM algorithm for Naive Bayes models is shown in figure 1. It is an iterative algorithm, defining a series of parameter values $\underline{\theta}^0, \underline{\theta}^1, \ldots, \underline{\theta}^T$. The initial parameter values $\underline{\theta}^0$ are chosen to be random. At each iteration the new parameter values $\underline{\theta}^t$ are calculated as a function of the training set, and the previous parameter values $\underline{\theta}^{t-1}$. A first key step at each iteration is to calculate the values

$$\delta(y|i) = p(y|\underline{x}^{(i)}; \underline{\theta}^{t-1})$$

for each example $i \in \{1 \ldots n\}$, for each possible label $y \in \{1 \ldots k\}$. The value for $\delta(y|i)$ is the conditional probability for label y on the i'th example, given the parameter values $\underline{\theta}^{t-1}$. The second step at each iteration is to calculate the new parameter values, as

$$q^t(y) = \frac{1}{n} \sum_{i=1}^{n} \delta(y|i)$$

and

$$q_j^t(x|y) = \frac{\sum_{i:x_j^i = x} \delta(y|i)}{\sum_i \delta(y|i)}$$

Note that these updates are very similar in form to the ML parameter estimates in the case of fully observed data. In fact, if we have labeled examples $(x^{(i)}, y^{(i)})$ for $i \in \{1 \ldots n\}$, and define

$$\delta(y|i) = 1 \text{ if } y_i = y, 0 \text{ otherwise}$$

then it is easily verified that the estimates would be identical to those given in Eqs. 2 and 3. Thus the new algorithm can be interpreted as a method where we replaced the definition

$$\delta(y|i) = 1 \text{ if } y_i = y, 0 \text{ otherwise}$$

used for labeled data with the definition

$$\delta(y|i) = p(y|\underline{x}^{(i)}; \underline{\theta}^{t-1})$$

for unlabeled data. Thus we have essentially "hallucinated" the $\delta(y|i)$ values, based on the previous parameters, given that we do not have the actual labels $y^{(i)}$.

The next section describes why this method is justified. First, however, we need the following property of the algorithm:

Theorem 3 *The parameter estimates $q^t(y)$ and $q^t(x|y)$ for $t = 1 \ldots T$ are*

$$\underline{\theta}^t = \arg\max_{\underline{\theta}} Q(\underline{\theta}, \underline{\theta}^{t-1})$$

under the constraints $q(y) \geq 0$, $\sum_{y=1}^{k} q(y) = 1$, $q_j(x|y) \geq 0$, $\sum_{x \in \{-1,+1\}} q_j(x|y) = 1$, where

$$\begin{aligned} Q(\underline{\theta}, \underline{\theta}^{t-1}) &= \sum_{i=1}^{n} \sum_{y=1}^{k} p(y|\underline{x}^{(i)}; \underline{\theta}^{t-1}) \log p(x^{(i)}, y; \underline{\theta}) \\ &= \sum_{i=1}^{n} \sum_{y=1}^{k} p(y|\underline{x}^{(i)}; \underline{\theta}^{t-1}) \log \left(q(y) \prod_{j=1}^{d} q_j(x_j^{(i)}|y) \right) \end{aligned}$$

6 The EM Algorithm in General Form

In this section we describe a general form of the EM algorithm; the EM algorithm for Naive Bayes is a special case of this general form. We then discuss convergence properties of the general form, which in turn give convergence guarantees for the EM algorithm for Naive Bayes.

6.1 The Algorithm

The general form of the EM algorithm is shown in figure 2. We assume the following setting:

- We have sets $\mathcal{X}$ and $\mathcal{Y}$, where $\mathcal{Y}$ is a finite set (e.g., $\mathcal{Y} = \{1, 2, \ldots k\}$ for some integer k). We have a *model* $p(x, y; \underline{\theta})$ that assigns a probability to each (x, y) such that $x \in \mathcal{X}$, $y \in \mathcal{Y}$, under parameters $\underline{\theta}$. Here we use $\underline{\theta}$ to refer to a vector including all parameters in the model.

Inputs: An integer k specifying the number of classes. Training examples $(x^{(i)})$ for $i = 1 \dots n$ where each $x^{(i)} \in \{-1, +1\}^d$. A parameter T specifying the number of iterations of the algorithm.

Initialization: Set $q^0(y)$ and $q_j^0(x|y)$ to some initial values (e.g., random values) satisfying the constraints

- $q^0(y) \geq 0$ for all $y \in \{1 \dots k\}$. $\sum_{y=1}^k q^0(y) = 1$.
- For all y, j, x, $q_j^0(x|y) \geq 0$. For all $y \in \{1 \dots k\}$, for all $j \in \{1 \dots d\}$,

$$\sum_{x \in \{-1,+1\}} q_j^0(x|y) = 1$$

Algorithm:
For $t = 1 \dots T$

1. For $i = 1 \dots n$, for $y = 1 \dots k$, calculate

$$\delta(y|i) = p(y|\underline{x}^{(i)}; \underline{\theta}^{t-1}) = \frac{q^{t-1}(y) \prod_{j=1}^d q_j^{t-1}(x_j^{(i)}|y)}{\sum_{y=1}^k q^{t-1}(y) \prod_{j=1}^d q_j^{t-1}(x_j^{(i)}|y)}$$

2. Calculate the new parameter values:

$$q^t(y) = \frac{1}{n} \sum_{i=1}^n \delta(y|i) \qquad q_j^t(x|y) = \frac{\sum_{i:x_j^{(i)}=x} \delta(y|i)}{\sum_i \delta(y|i)}$$

Output: Parameter values $q^T(y)$ and $q^T(x|y)$.

Figure 1: The EM Algorithm for Naive Bayes Models

For example, in Naive Bayes we have $\mathcal{X} = \{-1, +1\}^d$ for some integer d, and $\mathcal{Y} = \{1 \ldots k\}$ for some integer k. The parameter vector $\underline{\theta}$ contains parameters of the form $q(y)$ and $q_j(x|y)$. The model is

$$p(\underline{x}, y; \underline{\theta}) = q(y) \prod_{j=1}^{d} q_j(x|y)$$

- We use Ω to refer to the set of all valid parameter settings in the model.

  For example, in Naive Bayes Ω contains all parameter vectors such that $q(y) \geq 0$, $\sum_y q(y) = 1$, $q_j(x|y) \geq 0$, and $\sum_x q_j(x|y) = 1$ (i.e., the usual constraints on parameters in a Naive Bayes model).

- We have a training set consisting of examples $x^{(i)}$ for $i = 1 \ldots n$, where each $x^{(i)} \in \mathcal{X}$.

- The log-likelihood function is then

$$L(\underline{\theta}) = \sum_{i=1}^{n} \log p(x^{(i)}; \underline{\theta}) = \sum_{i=1}^{n} \log \sum_{y \in \mathcal{Y}} p(x^{(i)}, y; \underline{\theta})$$

- The maximum likelihood estimates are

$$\underline{\theta}^* = \arg \max_{\underline{\theta} \in \Omega} L(\underline{\theta})$$

In general, finding the maximum-likelihood estimates in this setting is intractable (the function $L(\underline{\theta})$ is a difficult function to optimize, because it contains many local optima).

The EM algorithm is an iterative algorithm that defines parameter settings $\underline{\theta}^0, \underline{\theta}^1, \ldots, \underline{\theta}^T$ (again, see figure 2). The algorithm is driven by the updates

$$\underline{\theta}^t = \arg \max_{\underline{\theta} \in \Omega} Q(\underline{\theta}, \underline{\theta}^{t-1})$$

for $t = 1 \ldots T$. The function $Q(\underline{\theta}, \underline{\theta}^{t-1})$ is defined as

$$Q(\underline{\theta}, \underline{\theta}^{t-1}) = \sum_{i=1}^{n} \sum_{y \in \mathcal{Y}} \delta(y|i) \log p(x^{(i)}, y; \underline{\theta}) \qquad (11)$$

where

$$\delta(y|i) = p(y|x^{(i)}; \underline{\theta}^{t-1}) = \frac{p(x^{(i)}, y; \underline{\theta}^{t-1})}{\sum_{y \in \mathcal{Y}} p(x^{(i)}, y; \underline{\theta}^{t-1})}$$

Thus as described before in the EM algorithm for Naive Bayes, the basic idea is to fill in the $\delta(y|i)$ values using the conditional distribution under the previous parameter values (i.e., $\delta(y|i) = p(y|x^{(i)}; \underline{\theta}^{t-1})$).

Inputs: Sets $\mathcal{X}$ and $\mathcal{Y}$, where $\mathcal{Y}$ is a finite set (e.g., $\mathcal{Y} = \{1, 2, \dots k\}$ for some integer k). A model $p(x, y; \underline{\theta})$ that assigns a probability to each (x, y) such that $x \in \mathcal{X}$, $y \in \mathcal{Y}$, under parameters $\underline{\theta}$. A set of Ω of possible parameter values in the model. A training sample $x^{(i)}$ for $i \in \{1 \dots n\}$, where each $x^{(i)} \in \mathcal{X}$. A parameter T specifying the number of iterations of the algorithm.

Initialization: Set $\underline{\theta}^0$ to some initial value in the set Ω (e.g., a random initial value under the constraint that $\underline{\theta} \in \Omega$).

Algorithm:
For $t = 1 \dots T$

$$\underline{\theta}^t = \arg\max_{\underline{\theta} \in \Omega} Q(\underline{\theta}, \underline{\theta}^{t-1})$$

where

$$Q(\underline{\theta}, \underline{\theta}^{t-1}) = \sum_{i=1}^{n} \sum_{y \in \mathcal{Y}} \delta(y|i) \log p(x^{(i)}, y; \underline{\theta})$$

and

$$\delta(y|i) = p(y|x^{(i)}; \underline{\theta}^{t-1}) = \frac{p(x^{(i)}, y; \underline{\theta}^{t-1})}{\sum_{y \in \mathcal{Y}} p(x^{(i)}, y; \underline{\theta}^{t-1})}$$

Output: Parameters $\underline{\theta}^T$.

Figure 2: The EM Algorithm in General Form

Remark: Relationship to Maximum-Likelihood Estimation for Fully Observed Data. For completeness, figure 3 shows the algorithm for maximum-likelihood estimation in the case of fully observed data: that is, the case where labels $y^{(i)}$ are also present in the training data. In this case we simply set

$$\underline{\theta}^* = \arg\max_{\underline{\theta} \in \Omega} \sum_{i=1}^{n} \sum_{y \in \mathcal{Y}} \delta(y|i) \log p(x^{(i)}, y; \underline{\theta}) \qquad (12)$$

where $\delta(y|i) = 1$ if $y = y^{(i)}$, 0 otherwise.

Crucially, note the similarity between the optimization problems in Eq. 12 and Eq. 11. In many cases, *if the problem in Eq. 12 is easily solved (e.g., it has a closed-form solution), then the problem in Eq. 11 is also easily solved.*

Inputs: Sets $\mathcal{X}$ and $\mathcal{Y}$, where $\mathcal{Y}$ is a finite set (e.g., $\mathcal{Y} = \{1, 2, \ldots k\}$ for some integer k). A model $p(x, y; \underline{\theta})$ that assigns a probability to each (x, y) such that $x \in \mathcal{X}$, $y \in \mathcal{Y}$, under parameters $\underline{\theta}$. A set of Ω of possible parameter values in the model. A training sample $(x^{(i)}, y^{(i)})$ for $i \in \{1 \ldots n\}$, where each $x^{(i)} \in \mathcal{X}$, $y^{(i)} \in \mathcal{Y}$.

Algorithm: Set $\underline{\theta}^* = \arg\max_{\underline{\theta}\in\Omega} L(\underline{\theta})$ where

$$L(\underline{\theta}) = \sum_{i=1}^{n} \log p(x^{(i)}, y^{(i)}; \underline{\theta}) = \sum_{i=1}^{n} \sum_{y\in\mathcal{Y}} \delta(y|i) \log p(x^{(i)}, y; \underline{\theta})$$

and

$$\delta(y|i) = 1 \text{ if } y = y^{(i)}, 0 \text{ otherwise}$$

Output: Parameters $\underline{\theta}^*$.

Figure 3: Maximum-Likelihood Estimation with Fully Observed Data

6.2 Guarantees for the Algorithm

We now turn to guarantees for the algorithm in figure 2. The first important theorem (which we will prove very shortly) is as follows:

Theorem 4 *For any* $\underline{\theta}, \underline{\theta}^{t-1} \in \Omega$, $L(\underline{\theta}) - L(\underline{\theta}^{t-1}) \geq Q(\underline{\theta}, \underline{\theta}^{t-1}) - Q(\underline{\theta}^{t-1}, \underline{\theta}^{t-1})$.

The quantity $L(\underline{\theta}) - L(\underline{\theta}^{t-1})$ is the amount of progress we make when moving from parameters $\underline{\theta}^{t-1}$ to $\underline{\theta}$. The theorem states that this quantity is lower-bounded by $Q(\underline{\theta}, \underline{\theta}^{t-1}) - Q(\underline{\theta}^{t-1}, \underline{\theta}^{t-1})$.

Theorem 4 leads directly to the following theorem, which states that the likelihood is non-decreasing at each iteration:

Theorem 5 *For* $t = 1 \ldots T$, $L(\underline{\theta}^t) \geq L(\underline{\theta}^{t-1})$.

Proof: By the definitions in the algorithm, we have

$$\underline{\theta}^t = \arg\max_{\underline{\theta}\in\Omega} Q(\underline{\theta}, \underline{\theta}^{t-1})$$

It follows immediately that

$$Q(\underline{\theta}^t, \underline{\theta}^{t-1}) \geq Q(\underline{\theta}^{t-1}, \underline{\theta}^{t-1})$$

(because otherwise $\underline{\theta}^t$ would not be the $\arg\max$), and hence

$$Q(\underline{\theta}^t, \underline{\theta}^{t-1}) - Q(\underline{\theta}^{t-1}, \underline{\theta}^{t-1}) \geq 0$$

But by theorem 4 we have

$$L(\underline{\theta}^t) - L(\underline{\theta}^{t-1}) \geq Q(\underline{\theta}^t, \underline{\theta}^{t-1}) - Q(\underline{\theta}^{t-1}, \underline{\theta}^{t-1})$$

and hence $L(\underline{\theta}^t) - L(\underline{\theta}^{t-1}) \geq 0$. □

Theorem 5 states that the log-likelihood is non-decreasing: but this is a relatively weak guarantee; for example, we would have $L(\underline{\theta}^t) - L(\underline{\theta}^{t-1}) \geq 0$ for $t = 1 \dots T$ for the trivial definition $\underline{\theta}^t = \underline{\theta}^{t-1}$ for $t = 1 \dots T$. However, under relatively mild conditions, it can be shown that in the limit as T goes to ∞, the EM algorithm does actually converge to a local optimum of the log-likelihood function $L(\underline{\theta})$. The proof of this is beyond the scope of this note; one reference is Wu, 1983, *On the Convergence Properties of the EM Algorithm.*

To complete this section, we give a proof of theorem 4. The proof depends on a basic property of the $\log$ function, namely that it is concave:

Remark: Concavity of the $\log$ **function.** The $\log$ function is concave. More explicitly, for any values $x_1, x_2, \dots, x_k$ where each $x_i > 0$, and for any values $\alpha_1, \alpha_2, \dots, \alpha_k$ where $\alpha_i \geq 0$ and $\sum_i \alpha_i = 1$,

$$\log\left(\sum_i \alpha_i x_i\right) \geq \sum_i \alpha_i \log x_i$$

The proof is then as follows:

$$\begin{aligned}
L(\underline{\theta}) - L(\underline{\theta}^{t-1}) &= \sum_{i=1}^{n} \log \frac{\sum_y p(x^{(i)}, y; \underline{\theta})}{\sum_y p(x^{(i)}, y; \underline{\theta}^{t-1})} \\
&= \sum_{i=1}^{n} \log \sum_y \left(\frac{p(x^{(i)}, y; \underline{\theta})}{p(x^{(i)}; \underline{\theta}^{t-1})} \right) \\
&= \sum_{i=1}^{n} \log \sum_y \left(\frac{p(y|x^{(i)}; \underline{\theta}^{t-1}) \times p(x^{(i)}, y; \underline{\theta})}{p(y|x^{(i)}; \underline{\theta}^{t-1}) \times p(x^{(i)}; \underline{\theta}^{t-1})} \right) && (13) \\
&= \sum_{i=1}^{n} \log \sum_y \left(\frac{p(y|x^{(i)}; \underline{\theta}^{t-1}) \times p(x^{(i)}, y; \underline{\theta})}{p(x^{(i)}, y; \underline{\theta}^{t-1})} \right) \\
&\geq \sum_{i=1}^{n} \sum_y p(y|x^{(i)}; \underline{\theta}^{t-1}) \log \left(\frac{p(x^{(i)}, y; \underline{\theta})}{p(x^{(i)}, y; \underline{\theta}^{t-1})} \right) && (14)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{n}\sum_{y} p(y|x^{(i)};\underline{\theta}^{t-1}) \log p(x^{(i)}, y; \underline{\theta}) - \sum_{i=1}^{n}\sum_{y} p(y|x^{(i)};\underline{\theta}^{t-1}) \log p(x^{(i)}, y; \underline{\theta}^{t-1}) \\
&= Q(\underline{\theta}, \underline{\theta}^{t-1}) - Q(\underline{\theta}^{t-1}, \underline{\theta}^{t-1}) \qquad (15)
\end{aligned}
$$

The proof uses some simple algebraic manipulations, together with the property that the $\log$ function is concave. In Eq. 13 we multiply both numerator and denominator by $p(y|x^{(i)};\underline{\theta}^{t-1})$. To derive Eq. 14 we use the fact that the $\log$ function is concave: this allows us to pull the $p(y|x^{(i)};\underline{\theta}^{t-1})$ outside the $\log$.

A Proof of Theorem 1

We now prove the result in theorem 1. Our first step is to re-write the log-likelihood function in a way that makes direct use of "counts" taken from the training data:

$$
\begin{aligned}
L(\underline{\theta}) &= \sum_{i=1}^{n} \log q(y_i) + \sum_{i=1}^{n}\sum_{j=1}^{d} \log q_j(x_{i,j}|y_i) \\
&= \sum_{y \in \mathcal{Y}} \text{count}(y) \log q(y) \\
&\quad + \sum_{j=1}^{d} \sum_{y\in\mathcal{Y}} \sum_{x\in\{-1,+1\}} \text{count}_j(x|y) \log q_j(x|y) \qquad (16)
\end{aligned}
$$

where as before

$$
\text{count}(y) = \sum_{i=1}^{n} [[y^{(i)} = y]]
$$

$$
\text{count}_j(x|y) = \sum_{i=1}^{n} [[y_i = y \text{ and } x_j^{(i)} = x]]
$$

Eq. 16 follows intuitively because we are simply counting up the number of times each parameter of the form $q(y)$ or $q_j(x|y)$ appears in the sum

$$
\sum_{i=1}^{n} \log q(y_i) + \sum_{i=1}^{n}\sum_{j=1}^{d} \log q_j(x_{i,j}|y_i)
$$

To be more formal, consider the term

$$
\sum_{i=1}^{n} \log q(y^{(i)})
$$

We can re-write this as

$$\begin{aligned}\sum_{i=1}^{n} \log q(y^{(i)}) &= \sum_{i=1}^{n}\sum_{y=1}^{k}[[y^{(i)} = y]] \log q(y) \\ &= \sum_{y=1}^{k}\sum_{i=1}^{n}[[y^{(i)} = y]] \log q(y) \\ &= \sum_{y=1}^{k} \log q(y) \sum_{i=1}^{n}[[y^{(i)} = y]] \\ &= \sum_{y=1}^{k} (\log q(y)) \times \text{count}(y)\end{aligned}$$

The identity

$$\sum_{i=1}^{n}\sum_{j=1}^{d} \log q_j(x_{i,j}|y_i) = \sum_{j=1}^{d}\sum_{y\in\mathcal{Y}}\sum_{x\in\{-1,+1\}} \text{count}_j(x|y) \log q_j(x|y)$$

can be shown in a similar way.

Now consider again the expression in Eq. 16:

$$\sum_{y\in\mathcal{Y}} \text{count}(y) \log q(y) + \sum_{j=1}^{d}\sum_{y\in\mathcal{Y}}\sum_{x\in\{-1,+1\}} \text{count}_j(x|y) \log q_j(x|y)$$

Consider first maximization of this function with respect to the $q(y)$ parameters. It is easy to see that the term

$$\sum_{j=1}^{d}\sum_{y\in\mathcal{Y}}\sum_{x\in\{-1,+1\}} \text{count}_j(x|y) \log q_j(x|y)$$

does not depend on the $q(y)$ parameters at all. Hence to pick the optimal $q(y)$ parameters, we need to simply maximize

$$\sum_{y\in\mathcal{Y}} \text{count}(y) \log q(y)$$

subject to the constraints $q(y) \geq 0$ and $\sum_{y=1}^{k} q(y) = 1$. But by theorem 2, the values for $q(y)$ which maximize this expression under these constraints is simply

$$q(y) = \frac{\text{count}(y)}{\sum_{y=1}^{k} \text{count}(y)} = \frac{\text{count}(y)}{n}$$

By a similar argument, we can maximize each term of the form

$$\sum_{x \in \{-1,+1\}} \text{count}_j(x|y) \log q_j(x|y)$$

for a given $j \in \{1 \ldots k\}$, $y \in \{1 \ldots k\}$ separately. Applying theorem 2 gives

$$q_j(x|y) = \frac{\text{count}_j(x|y)}{\sum_{x \in \{-1,+1\}} \text{count}_j(x|y)}$$