Next Up Previous Contents Index
Next: Relation to multinomial unigram **Up:** Text classification and Naive **Previous:** The text classification problem **Contents** **Index**

Naive Bayes text classification

The first supervised learning method we introduce is the *multinomial Naive Bayes* or *multinomial NB* model, a probabilistic learning method. The probability of a document d being in class c is computed as

$$P(c|d) \propto P(c) \prod_{1\leq k\leq n_d} P(t_k|c) \tag{113}$$

where $P(t_k|c)$ is the conditional probability of term t_k occurring in a document of class c. We interpret $P(t_k|c)$ as a measure of how much evidence t_k contributes that c is the correct class. $P(c)$ is the prior probability of a document occurring in class c. If a document's terms do not provide clear evidence for one class versus another, we choose the one that has a higher prior probability. $\langle t_1, t_2, \ldots, t_{n_d}\rangle$ are the tokens in d that are part of the vocabulary we use for classification and n_d is the number of such tokens in d. For example, $\langle t_1, t_2, \ldots, t_{n_d}\rangle$ for the one-sentence document Beijing and Taipei join the WTO might be $\langle \text{Beijing}, \text{Taipei}, \text{join}, \text{WTO}\rangle$, with $n_d = 4$, if we treat the terms and and the as stop words.

In text classification, our goal is to find the *best* class for the document. The best class in NB classification is the most likely or *maximum a posteriori* (*MAP*) class c_{map}:

$$c_{map} = \arg\max_{c\in\mathbb{C}} \hat{P}(c|d) = \arg\max_{c\in\mathbb{C}} \ \hat{P}(c) \prod_{1\leq k\leq n_d} \hat{P}(t_k|c). \tag{114}$$

We write $\hat{P}$ for P because we do not know the true values of the parameters $P(c)$ and $P(t_k|c)$, but estimate them from the training set as we will see in a moment.

In Equation 114, many conditional probabilities are multiplied, one for each position $1\leq k\leq n_d$. This can result in a floating point underflow. It is therefore better to perform the computation by adding logarithms of probabilities instead of multiplying probabilities. The class with the highest log probability score is still the most probable; $\log(xy) = \log(x) + \log(y)$ and the logarithm function is monotonic.

Hence, the maximization that is actually done in most implementations of NB is:

$$c_{map} = \arg\max_{c \in \mathbb{C}} [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]. \qquad (115)$$

Equation 115 has a simple interpretation. Each conditional parameter $\log \hat{P}(t_k|c)$ is a weight that indicates how good an indicator t_k is for c. Similarly, the prior $\log \hat{P}(c)$ is a weight that indicates the relative frequency of c. More frequent classes are more likely to be the correct class than infrequent classes. The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class, and Equation 115 selects the class for which we have the most evidence.

We will initially work with this intuitive interpretation of the multinomial NB model and defer a formal derivation to Section 13.4 .

How do we estimate the parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$? We first try the *maximum likelihood estimate* (MLE; probtheory), which is simply the relative frequency and corresponds to the most likely value of each parameter given the training data. For the priors this estimate is:

$$\hat{P}(c) = \frac{N_c}{N}, \qquad (116)$$

where N_c is the number of documents in class c and N is the total number of documents.

We estimate the conditional probability $\hat{P}(t|c)$ as the relative frequency of term t in documents belonging to class c:

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}, \qquad (117)$$

where T_{ct} is the number of occurrences of t in training documents from class c, including multiple occurrences of a term in a document. We have made the *positional independence assumption* here, which we will discuss in more detail in the next section: T_{ct} is a count of occurrences in all positions k in the documents in the training set. Thus, we do not compute different estimates for different positions and, for example, if a word occurs twice in a document, in positions k_1 and k_2, then $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$.

The problem with the MLE estimate is that it is zero for a term-class combination that did not occur in the training data. If the term WTO in the training data only occurred in China documents, then the MLE estimates for the other classes, for example UK, will be zero:

$$\hat{P}(\mathrm{WTO}|UK) = 0. \qquad (118)$$

Now, the one-sentence document Britain is a member of the WTO will get a conditional probability of zero for UK because we are multiplying the conditional probabilities for all terms in Equation 113. Clearly, the model should assign a high probability to the UK class because the term Britain occurs. The problem is that the zero probability for WTO cannot be ``conditioned away,'' no matter how strong the evidence for the class UK from other features. The estimate is 0 because of *sparseness* : The training data are never large enough to represent the frequency of rare events adequately, for example, the frequency of WTO occurring in UK documents.

```
TRAINMULTINOMIALNB($\mathbb{C}, \mathbb{D}$)
 1  $V \leftarrow$ EXTRACTVOCABULARY($\mathbb{D}$)
 2  $N \leftarrow$ COUNTDOCS($\mathbb{D}$)
 3  for each $c \in \mathbb{C}$
 4  do $N_c \leftarrow$ COUNTDOCSINCLASS($\mathbb{D}, c$)
 5      $prior[c] \leftarrow N_c/N$
 6      $text_c \leftarrow$ CONCATENATETEXTOFALLDOCSINCLASS($\mathbb{D}, c$)
 7      for each $t \in V$
 8      do $T_{ct} \leftarrow$ COUNTTOKENSOFTERM($text_c, t$)
 9      for each $t \in V$
10      do $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
11  return $V, prior, condprob$

APPLYMULTINOMIALNB($\mathbb{C}, V, prior, condprob, d$)
1  $W \leftarrow$ EXTRACTTOKENSFROMDOC($V, d$)
2  for each $c \in \mathbb{C}$
3  do $score[c] \leftarrow \log prior[c]$
4      for each $t \in W$
5      do $score[c] \mathrel{+}= \log condprob[t][c]$
6  return $\arg\max_{c \in \mathbb{C}} score[c]$
```

Figure 13.2: Naive Bayes algorithm (multinomial model): Training and testing.

To eliminate zeros, we use *add-one* or *Laplace smoothing*, which simply adds one to each count (cf. Section 11.3.2):

$$\hat{P}(t|c) = \frac{T_{ct}+1}{\sum_{t' \in V}(T_{ct'}+1)} = \frac{T_{ct}+1}{(\sum_{t' \in V} T_{ct'}) + B'}, \qquad (119)$$

where $B = |V|$ is the number of terms in the vocabulary. Add-one smoothing can be interpreted as a uniform prior (each term occurs once for each class) that is then updated as evidence from the training data comes in. Note that this is a prior probability for the occurrence of a *term* as opposed to the prior probability of a *class* which we estimate in Equation 116 on the document level.

We have now introduced all the elements we need for training and applying an NB classifier. The complete algorithm is described in Figure 13.2 .

Table 13.1: Data for parameter estimation examples.

	docID	words in document	in c = China?
training set	1	Chinese Beijing Chinese	yes
	2	Chinese Chinese Shanghai	yes
	3	Chinese Macao	yes
	4	Tokyo Japan Chinese	no
test set	5	Chinese Chinese Chinese Tokyo Japan	?

Worked example. For the example in Table 13.1 , the multinomial parameters we need to classify the test document are the priors $\hat{P}(c) = 3/4$ and $\hat{P}(\overline{c}) = 1/4$ and the following conditional probabilities:

$$\begin{aligned} \hat{P}(\text{Chinese}|c) &= (5+1)/(8+6) = 6/14 = 3/7 \\ \hat{P}(\text{Tokyo}|c) = \hat{P}(\text{Japan}|c) &= (0+1)/(8+6) = 1/14 \\ \hat{P}(\text{Chinese}|\overline{c}) &= (1+1)/(3+6) = 2/9 \\ \hat{P}(\text{Tokyo}|\overline{c}) = \hat{P}(\text{Japan}|\overline{c}) &= (1+1)/(3+6) = 2/9 \end{aligned}$$

The denominators are $(8+6)$ and $(3+6)$ because the lengths of $text_c$ and $text_{\overline{c}}$ are 8 and 3, respectively, and because the constant B in Equation 119 is 6 as the vocabulary consists of six terms.

We then get:

$$\begin{aligned} \hat{P}(c|d_5) &\propto 3/4 \cdot (3/7)^3 \cdot 1/14 \cdot 1/14 \approx 0.0003. \\ \hat{P}(\overline{c}|d_5) &\propto 1/4 \cdot (2/9)^3 \cdot 2/9 \cdot 2/9 \approx 0.0001. \end{aligned}$$

Thus, the classifier assigns the test document to c = China. The reason for this classification decision is that the three occurrences of the positive indicator Chinese in d_5 outweigh the occurrences of the two negative indicators Japan and Tokyo. **End worked example.**

Table 13.2: Training and test times for NB.

mode	time complexity
training	$\Theta(\vert\mathbb{D}\vert L_{ave} + \vert\mathbb{C}\vert\vert V\vert)$
testing	$\Theta(L_a + \vert\mathbb{C}\vert M_a) = \Theta(\vert\mathbb{C}\vert M_a)$

What is the time complexity of NB? The complexity of computing the parameters is $\Theta(|\mathbb{C}||V|)$ because the set of parameters consists of $|\mathbb{C}||V|$ conditional probabilities and $|\mathbb{C}|$ priors. The preprocessing necessary for computing the parameters (extracting the vocabulary, counting terms, etc.) can be done in one pass through the training data. The time complexity of this component is therefore $\Theta(|\mathbb{D}|L_{ave})$, where $|\mathbb{D}|$ is the number of documents and L_{ave} is the average length of a document.

We use $\Theta(|\mathbb{D}|L_{ave})$ as a notation for $\Theta(T)$ here, where T is the length of the training collection. This is nonstandard; $\Theta(.)$ is not defined for an average. We prefer expressing the time complexity in terms of $\mathbb{D}$ and L_{ave} because these are the primary statistics used to characterize training collections.

The time complexity of APPLYMULTINOMIALNB in Figure 13.2 is $\Theta(|\mathbb{C}|L_a)$. L_a and M_a are the numbers of tokens and types, respectively, in the test document . APPLYMULTINOMIALNB can be modified to be $\Theta(L_a + |\mathbb{C}|M_a)$ (Exercise 13.6). Finally, assuming that the length of test documents is bounded, $\Theta(L_a + |\mathbb{C}|M_a) = \Theta(|\mathbb{C}|M_a)$ because $L_a < b|C|M_a$ for a fixed constant b.

Table 13.2 summarizes the time complexities. In general, we have $|\mathbb{C}||V| < |\mathbb{D}|L_{ave}$, so both training and testing complexity are linear in the time it takes to scan the data. Because we have to look at the data at least once, NB can be said to have optimal time complexity. Its efficiency is one reason why NB is a popular text classification method.

Subsections

Next Up Previous Contents Index
Next: Relation to multinomial unigram **Up:** Text classification and Naive **Previous:**

The text classification problem **Contents** **Index**