Discriminative Multinominal Naive Bayes for Text Classification

Abstract

Multinominal Naive Bayes (MNB) algorithm has been widely used in text classification due to its computational advantage and simplicity. However, the previous research shows that MNB is not as effective as other discriminative classifiers, and ascribe the poor performance of MNB to its mismatch objective function: MNB maximizes likelihood rather than conditional likelihood or accuracy. In this paper, we propose a new text classification algorithm, called Discriminative Multinominal Naive Bayes (DMNB), which takes into account both the likelihood and the classification objectives during the frequency counting. Our empirical studies show that DMNB performs competitively with the state-of-the-art discriminative classifiers, such as Support Vector Machine (SVM) and Logistic Regression (LR) in terms of accuracy, without losing the advantages of MNB, namely its computational efficiency.

1 Introduction

Text classification (TC) is the task of assigning predefined categories or classes to text documents, and plays an important role in text mining applications. For example, content based recommenders train a TC model based on a few pages viewed by the user, and then make recommendations by classifying the text content into like (viewed) or not like categories. This application requires the TC algorithms to exhibit:

1. **Effectiveness**. Effectiveness should be achieved given a small training dataset, since the online recommendation is often based on a few user viewed pages.
2. **Efficiency**. While linear time complexity is mandatory, minimum computational cost is desired for fast user response time in online recommendation.
3. **Online learning ability**. As the user views pages one at a time, batch learning is often not suitable in this scenario.

These characteristics are similarly desirable in other applications of text mining. Multinominal Naive Bayes (MNB) has been widely used in text classification applications [6] due to its computational efficiency and simplicity. To learn a model, MNB only needs to scan each data point once, and learning can be done in an online manner. MNB has no explicit regularization parameters and thus is robust in practice.

The conventional wisdom is that the MNB may not be effective due to its mismatched objective function: MNB maximizes likelihood rather than the classification objective, e.g. conditional likelihood or accuracy [2]. As a result, the previous research shows that MNB is not as effective as other discriminative classifiers [10] that directly maximize the classification objective. However, though tremendous effort have been spent on improving the efficiency of discriminative classifiers [5, 11, 3], none of them can compete with MNB in computational efficiency.

In [9] the authors proposed a parameter learning method, called *Discriminative Frequency Estimate* (DFE), for Bayesian networks (BN), and showed DFE can be as effective as gradient descent methods but more efficient. While the simplest BN is a naive Bayes (NB), NB is not generally used in text classification. One important reason is that the training and prediction time complexity of NB is related to the number of unique words $|V|$ in the corpus T. In contrast, the time complexity of MNB is only related to the average number of unique words n appearing in documents in T. Note that we have $n << |V|$ due to the sparsity of text data.

In this paper, we propose a new text classification algorithm, called *Discriminative Multinominal Naive Bayes* (DMNB), which adapts the parameter learning method DFE of [9], to a Multinominal Naive Bayes model used for text classification. Our empirical studies show that the accuracy of DMNB is competitive with other discriminative classifiers, such as SMO and Logistic regression (LR) [7, 11, 3] in relatively large training datasets, and may outperform them in small datasets. The computational comparison shows that DMNB is significantly faster than any other known discriminative classifiers, including SVM^{perf}[5], which is currently being considered as the fastest SVM algorithm for text classification.

2 Related Work

In [6] authors conducted systematic empirical studies for comparing two different naive Bayes models: multi-variate Bernoulli and MNB, and concluded that the MNB consistently performs better than multi-variate Bernoulli in text datasets with large vocabulary size. Research presented in [10] confirmed the above conclusions, and further compared MNB with other classifiers. Their results also suggest that MNB significantly underperforms other discriminative classifiers, such as SVM with a linear kernel.

To improve the performance of MNB, [8] proposed *Complement Naive Bayes* (CNB). The authors point out that MNB may select poor weights for decision boundary when the class distribution in training data is imbalanced. They suggested while learning the conditional probability of one class, CNB uses the frequency information pertaining to all other classes,

[11] investigates a number of linear classifiers, including logistic regression, support vector machines and MNB. The authors pointed out that the regularization and numerical optimization procedures are critical for the performance of logistic regression, and then suggested to solve the numerical optimization problem by a cyclic coordinate descent algorithm which minimizes the classification error by learning parameters one after the other. [3] proposed a large scale logistic regression algorithm for text classification based on [11]. Their empirical studies showed that their approach produces compact predictive models which are competitive with SVM classifiers or ridge logistic regression combined with feature selection in terms of their efficiency.

Although recent years have seen the introduction of many text classification algorithms, text mining practitioners find the need for an effective, efficient and online learning algorithm. This paper attempts to provide one such solution.

3 Text Document Representation

In this section we provide a brief introduction to text classification. In doing so we limit our discussion to the *bag-of-words* representation and binary classification problem.

In text classification, a labeled document d is represented as $d = \{w_1, w_2, \cdots, w_i, c\}$, where w_i is the word frequency in the document d, and c is the class label of the document d. The set of unique words appearing in the whole document collection is often called dictionary V. We use the boldface lower case letters $\mathbf{w}$ for the set of word frequencies in a document d, and thus a document can also be represented as $\{\mathbf{w}, c\}$. We use T to indicate the training data and the superscript d^t for the t_{th} document in a dataset T. In general, we use a "hat" ($\hat{}$) to indicate parameter estimates.

Text representation often uses the *bag-of-words* approach. By ignoring the ordering of the words in documents, a word sequence can be transferred into a bag of words. In this way, only the frequency of a word in a document is recorded, and structural information about the document is ignored. In the *bag-of-words* approach, a document is often stored using the "sparse" format; that is: only the non-zero words are stored. This is because the "sparse" format can significantly reduce the storage space [4].

Text classification is often considered different from traditional machine learning because of its high-dimensional and sparse data charactersitics[4]. The high-dimensional data poses computational constraints, while the sparse data indicates that the number of features may be larger than the number of labeled examples. Thus, finding an algorithm which is both efficient and can generalize well is a challenge for this application domain.

4 Multinominal Naive Bayes

The task of text classification can be approached from a Bayesian learning perspective, which assumes that the word distributions in documents are generated by a specific parametric model, and the parameters can be estimated from the training data. Equation 1 shows Multinominal Naive Bayes (MNB) model [6] which is one such parametric model commonly used in text classification:

$$P(c|d) = \frac{P(c)\prod_{i=1}^{n} P(w_i|c)^{f_i}}{P(d)} \tag{1}$$

where f_i is the number of occurrences of a word w_i in a document d, $P(w_i|c)$ is the conditional probability that a word w_i may happen in a document d given the class value c, and n is the number of unique words appearing in the document d.

The parameters in Equation 1 can be estimated by a generative parameter learning approach, called maximum likelihood or *frequency estimate* (FE) , which is simply the relative frequency in data [2]. FE estimates the conditional probability $P(w_i|c)$ using the relative frequency of the word w_i in documents belonging to class c

$$\hat{P}(w_i|c) = \frac{f_{ic}}{f_c}, \tag{2}$$

where f_{ic} is the number of times that a word w_i appears in all documents with the class label c, and f_c is the total number of words in documents with class label c in T.

For convenience in implementation, the FE parameter learning method only needs to update the word frequencies f_{ic}, which can be easily converted to $P(w_i|c)$ during the prediction process. To compute the frequencies from a given training data set we go through each training document, and increase the entry for f_{ic} in a word frequency table by 1 or a constant. By scanning the training data set once we can obtain all the required frequencies and then compute the corresponding conditional probabilities.

One advantage of the Multinominal Naive Bayes model is that it can make predictions efficiently. Given a word w_i with frequency zero in a test document, the conditional probability $P(w_i|c)$ (the β_i parameter in Equation 8 as we will shortly discuss) can be ignored. Thus, to make predictions, MNB only needs to go through words with a non-zero count. This is commonly used in conjunction with the sparse representation of the text data. In contrast, other classifiers such as Bayesian networks, require one to go through all the words in the vocabulary of training data. Since in text classification the size of vocabulary is often much larger than the document length, MNB can make predictions much faster than these classifiers. In the next section we show that our training time can also benefit from the same principle.

While MNB is efficient in text classification, its parameter learning method FE is known to suffer from objective function mismatch problem [2]. The FE method is a generative learning approach because its objective function, shown in Equation 3 , is the log likelihood $logP(\mathbf{W}, C)$:

$$LL(T) = \sum_{t=1}^{|T|} log\hat{P}(c^t|\mathbf{w}^t) + \sum_{t=1}^{|T|} log\hat{P}(\mathbf{w}^t) \qquad (3)$$

The first term in this equation measures how well the classifier model estimates the probability of the class given the words. The second term measures how well the classifier model estimates the joint distribution of the words in documents.

On one hand, this mismatched objective function may mislead the learning process given a large number of words in documents. As the number of words grows larger, the value of $P(\mathbf{w})$ is decreased exponentially in n. While $P(\mathbf{w})$ is close to zero, the absolute value of $|logP(\mathbf{w})|$ will be very large. However, the first term $|logP(c|\mathbf{w})|$ will remain the same and thus the second term dominates the $LL(T)$ score. As a result, maximizing the $LL(T)$ may not always lead to an accurate classifier.

One the other hand, learning $LL(T)$ can fully utilize the information in data. If the MNB model is correct for the underlying data, the discriminative learning that directly maximize $P(c|\mathbf{w})$ ignores useful information $P(\mathbf{w})$. This would not be a problem if we have sufficient data. Unfortunately, the text data is often high-dimensional and sparse, and classifiers learned without $P(\mathbf{w})$ may not generalize well. We will verify this hypothesis in our experiments.

As a result, a parameter learning method that can integrate the advantages of generative learning and discriminative learning is desired for text classification.

5 Discriminative Multinominal Naive Bayes

We now introduce *Discriminative Multinominal Naive Bayes* (DMNB), which applies the parameter learning method proposed in [9] to a Multinominal naive Bayes model. The motivation of $DMNB$ is to maintain the frequency information while considering the discriminative nature of classification and thus is an integration of generative and discriminative learning.

The DMNB algorithm iterates through the training documents. For each document d^t, DMNB first computes the posterior probability $\hat{P}(c|d^t)$ and then updates the corresponding word frequencies using the difference between the true $P(c|d^t)$ and the posterior $\hat{P}(c|d^t)$. This difference is the prediction loss of the training document d^t and is defined in Equation 4.

$$L(d^t) = P(c|d^t) - \hat{P}(c|d^t). \qquad (4)$$

Since in practice the true value of $P(c|d^t)$ is unknown we assume $P(c|d^t) = 1$, where c is the class label of document d^t, which is known for all the documents in the training set.

Algorithm 1 shows the details of this process. f_{ic} is a word frequency, where i indicates the word w_i and c is the class label of a given document d^t. This frequency is zero for all unseen words.

Algorithm 1 Discriminative Multinominal Naive Bayes

```
1. Create an empty frequency table
2. For $t$ from 1 to $|T|$ Do
     • Estimate the probability parameters using Equation 3 and the appropriate existing $f_{ic}$
     • Compute the posterior probability $\hat{P}(c|d^t)$ using Equation 1.
     • Compute the loss $L(d^t)$ using Equation 4.
     • For each non-zero word $w_i$ in the document $d^t$
         – Update $f_{ic}$=$f_{ic}$+$L(d^t)$.
```

Since in the beginning all word frequencies f_{ic} are 0, the multiplicative term in Equation 1 will be 1, resulting in a predicted $\hat{P}(c|d)$ of $\frac{1}{|C|}$ for each class , after applying probability normalization. In each step, if the current parameters cannot accurately predict $P(c|d)$ for a document d, the corresponding entries f_{ic} of the table F_{ic} storing all the word frequencies are increased significantly. If the current parameter can perfectly predict $P(c|d)$, no entries will be changed.

It is clear that DMNB has the same computational properties as MNB: it is roughly as fast as MNB and also supports online learning. Compared to MNB, the only extra cost of DMNB is the computation of $L(d)$, which requires going through each non-zero word in a document. The single pass through the training data does not increase the time complexity of DMNB. As we will show later, one iteration through the training set T will be enough for DMNB to converge in text classification tasks. Moreover, DMNB can still learn parameters in an incremental way similar to MNB. Consequently, DMNB preserves the simplicity and computational advantages of MNB.

6 An Illustrative Example

In this section, we use an example to illustrate how the prediction generated from an MNB model will be distorted due to the dependencies between words, and then show how DMNB overcomes the dependency problem.

	docID	words in document	class label
training data	1	NB classifier performance	Evaluation
	2	SVM classifier performance	Classification
	3	NB classifier	Classification
test data	4	NB	Classification

Figure 1. Data used for frequency estimation examples

Figure 1 shows a text classification task which intends to identify whether a document belongs to class "Evaluation (E)" or "Classification (C)". The first 3 documents are used for training and the last document is used for testing. There are 4 words in the training vocabulary.

Assuming a multinominal naive Bayes model, the parameters $\hat{P}(c)$ and $\hat{P}(w_i|c)$ can be learned from the training data using the frequency estimate method. The probability estimates for Figure 1 are :

$\hat{P}(c = E) = \frac{1}{3}$
$\hat{P}(c = C) = \frac{2}{3}$
$\hat{P}(w_1 = NB|c = E) = \frac{1}{3}$
$\hat{P}(w_1 = NB|c = C) = \frac{1}{5}$

To predict the class of the last document we only need to compute the posterior probability ratio:

$$\frac{\hat{P}(c = E|w_1 = NB)}{\hat{P}(c = C|w_1 = NB)} = \frac{\frac{1}{3} \times \frac{1}{3}}{\frac{2}{3} \times \frac{1}{5}} = \frac{1}{2} \times \frac{5}{3} \tag{5}$$

The posterior probability ratio is $\frac{5}{6}$, which indicates that the probability of the document belonging to $Classification$ is larger than $Evaluation$ and thus the MNB model will assign the class label "Classification" to the last document according to Equation 1. Note that as we discussed in Section 3, an MNB model only needs to take into account non-zero words to make a prediction, rather than every word in the vocabulary (4 words in this case).

Now, we introduce dependency into this example by replacing the word "NB" with "naive Bayes" for the 4 documents in Figure 1. We re-calculate the parameters and prediction for the resulting datasets.

$$\begin{aligned}\frac{\hat{P}(c = E|w_1 = naive, w_2 = Bayes)}{\hat{P}(c = C|w_1 = naive, w_2 = Bayes)} &= \frac{\frac{1}{3} \times \frac{1}{4} \times \frac{1}{4}}{\frac{2}{3} \times \frac{1}{6} \times \frac{1}{6}} \\ &= \frac{1}{2} \times (\frac{6}{4})^2\end{aligned}$$

where $\frac{1}{2}$ is still the prior probability ratio, and $\frac{6}{4}$ is the ratio of the conditional probabilities $\hat{P}(w_1 = naive|c)$ and $\hat{P}(w_2 = bayes|c)$ respectively.

Since the posterior probability ratio is $\frac{9}{8}$, which indicates that the probability of belonging to $Evaluation$ is larger than $Classification$, the MNB model will assign the class label "Evaluation" to the last document. The problem here is the dependency between words "naive" and "bayes". This dependency results in overestimating the probability that the test document belongs to the "Evaluation" class. This is clearly a problem caused by the violation of independence assumption of MNB.

To make a correct classification for the test document in Figure 1, the posterior probability $\hat{P}(c = E|w_1 = naive, w_2 = bayes)$ should be less than 0.5, and DMNB can achieves this goal. Since both MNB and DMNB support online learning, their classification model can be updated for each given training instances. Figure 2 shows how the estimated probability $\hat{P}(c = E|w_1 = naive, w_2 = bayes)$ changes for MNB and DMNB respectively, as the number of training instances used increases. For the purpose of this demonstration we have re-read the training data over multiple pass, thus effectively simulating having a larger number of instances.

In each step t, both algorithms take a document, and update the corresponding word frequencies. Note, both MNB and DMNB allow many pass through the training data, and thus t can be more than 3 by scanning the training data many times. With the increased number of documents used, the estimated probability $\hat{P}(c = E|w_1 = naive, w_2 = bayes)$ in DMNB converges to approximately 0.35 , which leads to a correct classification. However, MNB converges to 0.52, and thus makes a wrong prediction.

This example illustrates that while both DMNB and MNB tend to converge with increased training effort, computing the frequencies in a discriminative fashion tends to result in a more accurate classification in an efficient manner

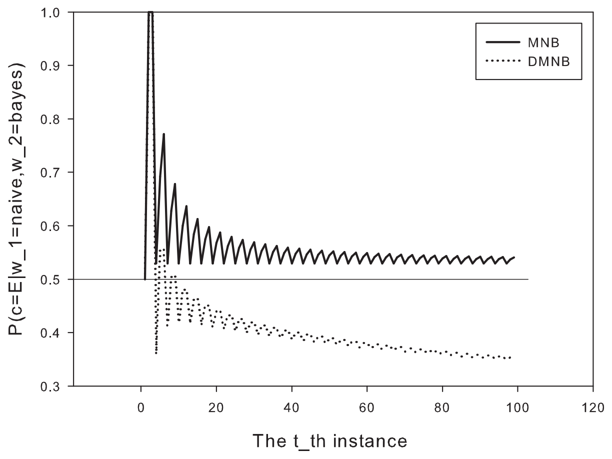

Figure 2. The y-axis is the predicted probability. The x-axis is the t_{th} document fed into the algorithms.

7 Relation with Linear Classifier

The MNB model can be written as a linear classifier using the following transformation:

$$\frac{P(+|d)}{P(-|d)} = \frac{P(+)}{P(-)} \prod_{i=1}^{n} (\frac{P(w_i|+)}{P(w_i|-)})^{f_i} \qquad (6)$$

Taking the logarithm of both sides of Equation 6, we get the following:

$$log\frac{P(+|d)}{P(-|d)} = log\frac{P(+)}{P(-)} + \sum_{i=1}^{n} f_i log\frac{P(w_i|+)}{P(w_i|-)} \qquad (7)$$

Rewriting Equation 7 using $y = log\frac{P(+|d)}{P(-|d)}$, $x_i = f_i$, $\beta_0 = log\frac{P(+)}{P(-)}$ and $\beta_i = log\frac{P(w_i|+)}{P(w_i|-)}$, we have the formula for well known linear classifier:

$$\hat{y} = \beta_0 + \sum_{i=1}^{n} \beta_i x_i \qquad (8)$$

Here, the prediction will be positive if $\hat{y} > 0$, and negative otherwise.

We note that the DFE parameter learning method in $DMNB$ is different from the stochastic gradient descent (SGD). Figure 3 illustrates the difference of three parameter learning methods: Frequency Estimate, Discriminative Frequency Estimate , and Stochastic Gradient Descent. The FE method updates the frequency table for each feature using constant 1 and thus parameters are determined by the likelihood information $P(w_i|c)$. The DFE method updates the frequency table for each feature using the current error and thus the final parameters are determined by the likelihood information and the prediction error. Note that updating each feature with the same weight can preserve the likelihood information $P(w_i|c)$. In contrast, the SGD method updates the parameters for each feature with different weights. Because of the constraint of probability axioms, the SGD method needs a probability normalization step, and thus ignores $P(w_i|c)$.

FE

	x_1	x_2
e^1	1	1
e^2	1	1
e^3	1	1

DFE

	x_1	x_2
e^1	err^1	err^1
e^2	err^2	err^2
e^3	err^3	err^3

SGD

	x_1	x_2
e^1	err^1*x_1	err^1*x_2
e^2	err^2*x_1	err^2*x_2
e^3	err^3*x_1	err^3*x_2

Figure 3. An illustration for different parameter learning methods. e^t is the t_{th} instance, x_i is the i_{th} word, and err^t is the prediction error for e^t based on the current parameters.

Thus, DFE can be viewed as a combination of FE and SGD: it considers both the likelihood information and prediction error, while FE and SGD only consider one of the two. If we view the parameter learning as a hypothesis searching procedure, Figure 3 shows that FE only searches a small hypothesis space, DFE searches a larger space, and SGD searches the largest space. More precisely, the search space of FE is a subset of DFE, and the search space of DFE is a subset of SGD. However, the larger search space requires more learning time. Thus, the questions is: whether the best hypothesis that can generalize well lies in the space that a parameter learning method can possibly search. Section 8.4 investigates this point.

8 Experiments

This section provides empirical comparisons of $DMNB$ and its competitors in terms of accuracy given different number of training data, and computational cost.

8.1 Dataset and Classifiers

We use 19 text classification benchmark data sets from the WEKA collection. These data sets are mainly from Reuters, TREC and OHSUMED, and have been widely used in text classification research [1]. All multi-class datasets are transformed into binary by keeping the largest two classes, but all conclusions in this paper were also verified in case of multi-class problems. All experiments were performed on an Intel Pentium(R) 3.40GHz computer with 4G of RAM.

Table 1. Dataset Descriptions

Dataset	NumIns	Vocabulary	DocLength	Positive
Fbis	893	2001	194	0.43
La1	1681	13196	151	0.56
La2	1664	12433	148	0.54
New3	1264	26833	278	0.44
Oh0	375	3183	56	0.48
Oh10	330	3239	56	0.5
Oh15	311	3101	60	0.49
Oh5	293	3013	58	0.49
Ohscal	3071	11466	56	0.47
Re0	927	2887	56	0.34
Re1	701	3759	54	0.47
Tr11	206	6430	288	0.35
Tr12	147	5805	349	0.36
Tr21	272	7903	502	0.84
Tr23	136	5833	463	0.33
Tr31	579	10129	261	0.60
Tr41	417	7455	191	0.41
Tr45	288	8262	335	0.55
Wap	537	8461	128	0.36

NumIns: the number of instances
Vocabulary: the vocabulary size
DocLength: the average length of documents in words
Positive: the percentage of the positive instances

The following summarizes the learning algorithms used in our experiments. All implementations, including our $DMNB$ algorithm, are currently available in WEKA.

1. **DMNB**: Discriminative Multinominal Naive Bayes proposed in this paper. One iteration, which means scanning the training data once, is used in our experiments.

2. **MNB**: Multinominal Naive Bayes [6] as implemented in WEKA.

3. **CNB**: Complement Naive Bayes [8] as implemented in WEKA.

4. **LR**: A large scale logistic regression classifier for text classification [11, 3]. The prior for each parameter is Gaussian and the number of iteration is 100, which is recommended in [11].

5. **SMO**: SMO in WEKA [7]. We use linear kernel for SVM due to its popularity and high performance in text classification [10]. The normalization option is turned off to improve the training time. Setting parameter $C = 1$ resulted in overall best performance in case of our datasets.

8.2 Performance Measurements

While DMNB performs overall best given different performance measures, including accuracy, F-measure and AUC, our discussions in this paper mainly focuses on accuracy due to the limited space.However, we also show the results of comparisons using F-measure and AUC as a reference.

Tables 2, 3 and 4 show the results of the two-tailed t-test with a 95% confidence interval for accuracy, F-measure and AUC. Each entry $l/t/w$ means that the algorithm in a given row when compared to the algorithm in the corresponding column, loses in l, ties in t, and wins in w data sets. Tables 5, 6 and 7 give the detailed experimental results of each algorithm on each data set for these three performance measures. All statistics and their standard deviations are based on 5 runs of 10-fold stratified cross validation.

8.3 Accuracy

We summarize the highlights briefly bellow:

1. DMNB is competitive with other state-of-the-art discriminative text classifiers, SMO and LR. Discriminative classifiers generally performs better than generative classifiers. Based on the average scores and the summary shown in Table 2, we can rank these classifier types as follows:

$$\{DMNB, SMO, LR\} > \{MNB, CNB\} \quad (9)$$

   where $>$ indicates better performance. When the performance difference is insignificant we group the classifier types together. Our results are consistent with [10]. For the sake of completeness, we also ran these experiment with SVM^{perf} [5] , and observed that SVM^{perf} performs similarly with $DMNB$ and SMO.

2. The poor performance of MNB is related to the number of words that appear in the documents. For example, in two datasets “Tr21” and “Tr23”, the accuracy achieved by the generative classifiers (MNB and CNB) are 20-30% lower than the discriminative classifiers. A quick examination of Table 1 reveals that “Tr21” and “Tr23” have the largest document length among all datasets. This observation supports our previous analysis in Section 5; the maximum likelihood estimate may not perform well given a large number of words in a document. We note that the multinominal model is not influenced by words with zero count in a document, and thus provide the average document length as a better estimation for the number of features for a dataset.

Table 2. Summary of Accuracy Comparisons.

	MNB	CNB	LR	SMO
DMNB	0/14/5	0/14/5	0/17/2	0/18/1
MNB		0/19/0	4/13/2	4/15/0
CNB			4/13/2	4/15/0
LR				2/17/0

Table 3. Summary of F-measure Comparisons.

	MNB	CNB	LR	SMO
DMNB	0/14/5	0/14/5	0/18/1	0/17/2
MNB		0/19/0	4/13/2	4/15/0
CNB			4/13/2	4/15/0
LR				1/18/0

8.4 Accuracy in Small Datasets

We conducted experiments in 19 datasets, and only present the results for 12 representative datasets due to the limited space. Figures 4 shows the relationship between the number of training instances and the average testing accuracy, which is obtained on the remaining of the data that is not used for training. The total number of runs is 1000. For training, we sampled datasets with 16, 32, 64, and 128 instances.

Figure 4 shows that DMNB performs consistently best in all training data size range, while SMO and LR can perform worse in small training dataset. Additionally we observed that:

1. If the data does not follow the underlying assumption in Equation 1, MNB achieves low accuracy in training data. In datasets "Tr21", "Tr23", and "Tr45", given 128 training instances, MNB achieves training accuracy of 74%, 78%, and 93% respectively, while $DMNB$ and other discriminative classifiers obtain 100% training accuracy. As a result, in these three datasets, MNB underperforms the other classifiers in testing accuracy as well, and this shows that $DMNB$ overcomes the deficiency caused by the naive assumption in the MNB model.

2. In the remaining 16 datasets, all methods obtain 100% training accuracy. However, MNB and $DMNB$ consistently achieve better testing accuracy compared to SMO and LR, when the number of training instances is small , i.e. 16 - 32. This results show that when the data approximately follows the underlying assumption of the MNB model, both MNB and $DMNB$ generalize better than the other methods that directly optimize the classification objectives.

Table 4. Summary of AUC Comparisons.

	MNB	CNB	LR	SMO
DMNB	0/12/7	0/9/10	0/7/12	0/11/8
MNB		0/9/10	0/11/8	0/12/7
CNB			3/14/2	4/15/0
LR				3/16/0

8.5 Computational Cost

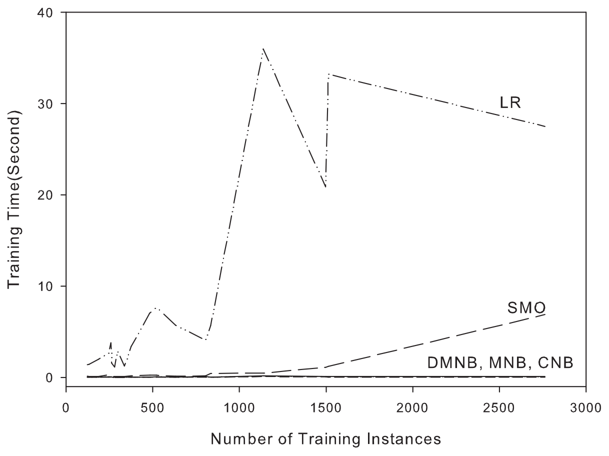

Figure 5. Training time comparisons in 19 datasets. X axis is the number of instances in training data, and Y axis is the training time required by an algorithm.

We measured the training time for each of the above mentioned algorithms. Figure 5 shows the average training time for 5 runs of 10-fold stratified cross validation. From this figure it is clear that the three naive Bayes algorithms are faster than LR and SMO, while $DMNB$ has similar training time cost with MNB. For example, in the largest dataset "Ohscal" with 2763 training instances, $DMNB$ is only two times slower than MNB, but 300 times faster than LR and approximately 100 times faster than SMO. For the sake of completeness, we also compare $DMNB$ to the latest linear SVM classifiers SVM^{perf} [5], and observed that the C implementation of SVM^{perf} is 5-30 times slower than the Java implementation of DMNB. However, we only present detailed SMO results in the paper due to its popularity.

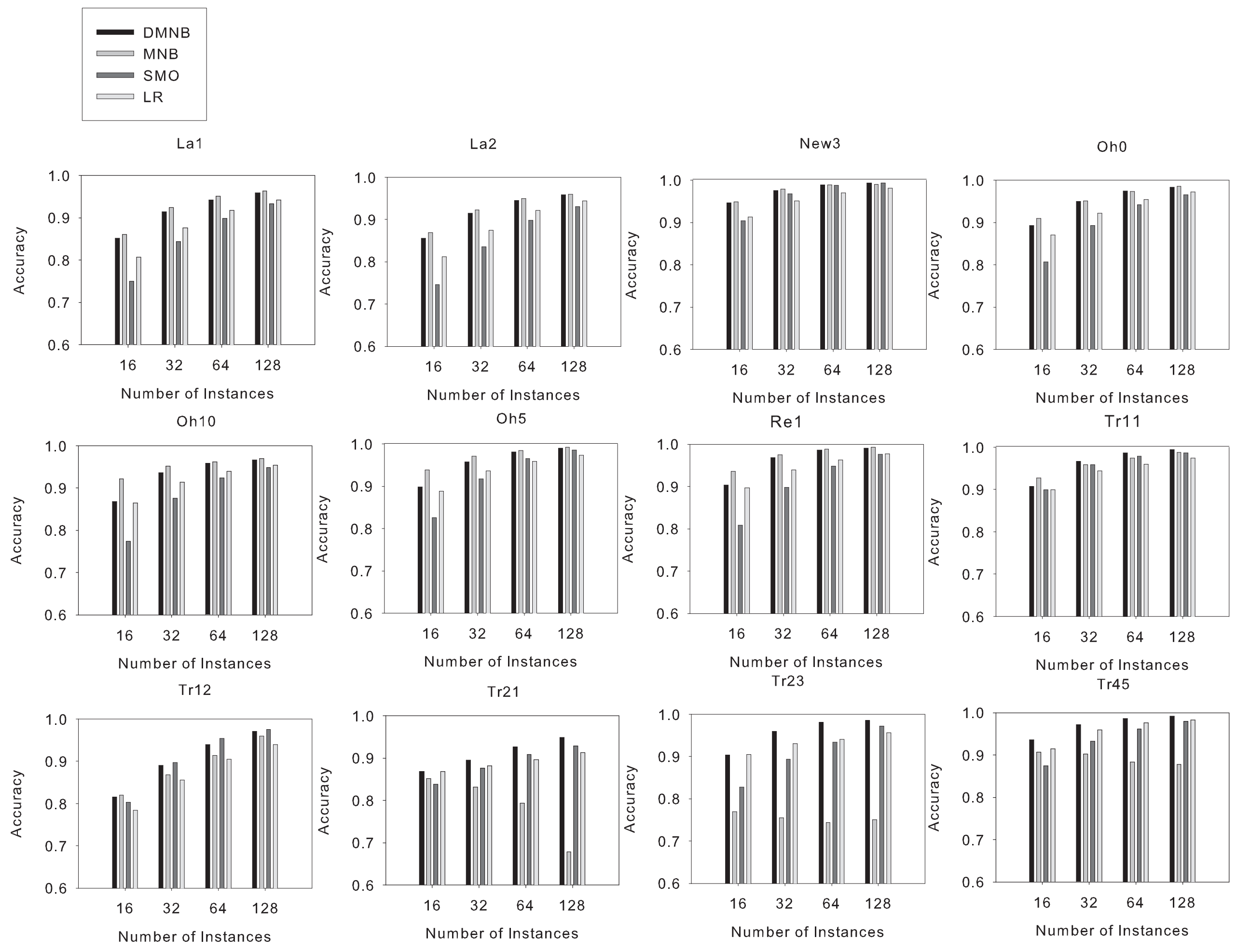

Figure 4. The Y axis is the testing accuracy, and the X axis is the number of training instances. In each group, the first bar is DMNB, the second bar is MNB, the third bar is SMO, and the fourth bar is LR

As our analysis in the last paragraph of Section 5, both $DMNB$ and MNB have the same time complexity $O(mn)$, where m is the number of instances and n is the average number of words appearing in documents. In contrast, SMO is $O(m^{1.8})$, and as Figure 5 shows the training time of SMO does not increases linearly with the number of training instances

9 Conclusions

In this paper, we have analyzed a widely used text classification model, Multinominal Naive Bayes, and its parameter learning method. The main contribution of our work is the new DMNB text classification algorithm, which maintains the computational efficiency, online learning, and simplicity of MNB while improving its accuracy. Our experiments show that the DMNB algorithm perform competitively with other state-of-the-art text classification algorithms, but often achieves better testing accuracy with small training data.

Finally, the implementation of $DMNB$ is now part of WEKA machine learning tools, and can be downloaded from the following link: http://www.cs.waikato.ac.nz/ml/weka/.

References

[1] G. Forman and I. Cohen. Learning from little: Comparison of classifiers given little training. In *Proceeding of PKDD2004*, pages 161–172. 2004.

[2] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

[3] A. Genkin, D. D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. 2004.

[4] T. Joachims. *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms*, volume 668 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, April 2002. Dissertation.

Table 5. Comparisons of Accuracy

Dataset	DMNB	MNB	CNB	LR	SMO
Fbis	99.64±0.57	98.95± 1.09	98.95± 1.09	99.91±0.31	99.89±0.34
La1	97.94±1.02	97.87± 0.76	97.87± 0.79	97.78±1.24	97.64±1.17
La2	97.73±1.32	97.36± 1.26	97.38± 1.26	97.67±1.39	97.58±1.32
New3	99.54±0.60	99.04± 0.77	99.04± 0.77	99.45±0.60	99.62±0.58
Oh0	98.61±1.97	99.63± 0.93	99.63± 0.93	97.70±2.47	98.56±1.89
Oh10	97.15±2.48	97.45± 2.69	97.45± 2.69	96.67±3.31	96.24±3.33
Oh15	99.42±1.25	99.04± 1.48	99.10± 1.45	98.72±1.84	97.69±2.76
Oh5	99.59±1.13	99.66± 1.04	99.66± 1.04	99.52±1.20	99.52±1.20
Ohscal	94.80±1.05	93.14± 1.33 ●	93.10± 1.35 ●	89.36±1.95 ●	93.19±1.51 ●
Re0	96.33±2.01	94.58± 2.00 ●	94.67± 1.98 ●	95.75±2.15	96.01±2.00
Re1	99.54±0.67	99.69± 0.60	99.69± 0.60	99.32±1.23	99.71±0.58
Tr11	99.32±1.69	99.03± 1.95	99.03± 1.95	98.16±2.76	99.32±1.69
Tr12	94.65±6.38	95.86± 5.05	95.99± 5.07	94.46±5.01	97.69±3.53
Tr21	96.47±3.49	60.81± 8.91 ●	60.66± 8.74 ●	93.32±3.42 ●	94.19±3.64
Tr23	97.35±4.66	75.20±10.83 ●	74.46±11.31 ●	94.97±6.26	96.84±5.45
Tr31	98.58±1.83	99.24± 1.11	99.24± 1.11	99.41±1.08	99.52±0.93
Tr41	99.47±1.00	99.28± 1.22	99.28± 1.22	98.71±1.69	99.23±1.13
Tr45	99.37±1.35	90.57± 5.59 ●	90.57± 5.59 ●	98.33±2.24	99.31±1.56
Wap	100.00±0.00	100.00± 0.00	100.00± 0.00	100.00±0.00	99.81±0.56
Average	98.19	94.55	94.51	97.33	97.98

● significantly worse, and ○ better, comparing to DMNB.

[5] T. Joachims. Training linear svms in linear time. In *KDD*, pages 217–226, 2006.

[6] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. AAAI-98 Workshop on Learning for Text Categorization., 1998.

[7] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods–Support Vector Learning*. MIT Press, 1998.

[8] J. D. Rennie, L. Shih, J. Teevan, and D. R. Karger. Tackling the poor assumptions of naive bayes text classifiers. In *ICML*, pages 616–623, 2003.

[9] J. Su, H. Zhang, C. X. Ling, and S. Matwin. Discriminative parameter learning for bayesian networks. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1016–1023. Morgan Kaufmann, 2008.

[10] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR*, pages 42–49, 1999.

[11] T. Zhang and F. J. Oles. Text categorization based on regularized linear classification methods. *Inf. Retr.*, 4(1):5–31, 2001.

Table 6. Comparisons of F-measure

Dataset	DMNB	MNB	CNB	LR	SMO
Fbis	1.00±0.01	0.99±0.01	0.99±0.01	1.00±0.00	1.00±0.00
La1	0.98±0.01	0.98±0.01	0.98±0.01	0.98±0.01	0.98±0.01
La2	0.98±0.01	0.98±0.01	0.98±0.01	0.98±0.01	0.98±0.01
New3	0.99±0.01	0.99±0.01	0.99±0.01	0.99±0.01	1.00±0.01
Oh0	0.99±0.02	0.99±0.01	1.00±0.01	0.97±0.03	0.99±0.02
Oh10	0.97±0.03	0.97±0.03	0.97±0.03	0.97±0.03	0.96±0.03
Oh15	0.99±0.01	0.99±0.02	0.99±0.02	0.99±0.02	0.98±0.03 •
Oh5	1.00±0.01	1.00±0.01	1.00±0.01	0.99±0.01	0.99±0.01
Ohscal	0.95±0.01	0.93±0.01 •	0.93±0.01 •	0.89±0.02 •	0.93±0.02 •
Re0	0.95±0.03	0.92±0.03 •	0.92±0.03 •	0.94±0.03	0.94±0.03
Re1	1.00±0.01	1.00±0.01	1.00±0.01	0.99±0.01	1.00±0.01
Tr11	0.99±0.03	0.99±0.03	0.99±0.03	0.97±0.04	0.99±0.02
Tr12	0.92±0.12	0.95±0.08	0.95±0.08	0.91±0.09	0.96±0.06
Tr21	0.98±0.06	0.70±0.09 •	0.70±0.09 •	0.96±0.02	0.97±0.02
Tr23	0.96±0.07	0.74±0.10 •	0.73±0.10 •	0.91±0.11	0.95±0.08
Tr31	0.99±0.02	0.99±0.01	0.99±0.01	1.00±0.01	1.00±0.01
Tr41	0.99±0.01	0.99±0.02	0.99±0.02	0.98±0.02	0.99±0.02
Tr45	0.99±0.01	0.91±0.05 •	0.91±0.05 •	0.98±0.02	0.99±0.01
Wap	1.00±0.00	1.00±0.00	1.00±0.00	1.00±0.00	1.00±0.01
Average	0.98	0.95	0.95	0.97	0.98

• significantly worse, and ○ better, comparing to DMNB.

Table 7. Comparisons of AUC

Dataset	DMNB	MNB	CNB	LR	SMO
Fbis	1.00±0.00	1.00±0.01 •	0.99±0.01 •	1.00±0.00	1.00±0.00
La1	1.00±0.00	0.99±0.01 •	0.98±0.01 •	0.98±0.01 •	0.98±0.01 •
La2	1.00±0.00	0.99±0.01 •	0.97±0.01 •	0.98±0.01 •	0.98±0.01 •
New3	1.00±0.00	1.00±0.00 •	0.99±0.01 •	0.99±0.01 •	1.00±0.01
Oh0	1.00±0.00	1.00±0.00	1.00±0.01	0.97±0.03 •	0.99±0.02 •
Oh10	1.00±0.01	0.99±0.02	0.97±0.03 •	0.97±0.03 •	0.96±0.03 •
Oh15	1.00±0.00	1.00±0.00	0.99±0.02	0.99±0.02 •	0.98±0.03 •
Oh5	1.00±0.00	1.00±0.00	1.00±0.01	0.99±0.01	1.00±0.01
Ohscal	0.99±0.00	0.98±0.01 •	0.93±0.01 •	0.90±0.02 •	0.93±0.01 •
Re0	0.99±0.01	0.98±0.01 •	0.94±0.03 •	0.95±0.03 •	0.95±0.02 •
Re1	1.00±0.00	1.00±0.00	1.00±0.01	0.99±0.01	1.00±0.01
Tr11	1.00±0.01	0.99±0.01	0.99±0.01	0.98±0.04	0.99±0.02
Tr12	0.97±0.07	0.96±0.07	0.96±0.06	0.92±0.07	0.97±0.05
Tr21	0.97±0.06	0.92±0.08 •	0.74±0.07 •	0.78±0.11 •	0.89±0.10 •
Tr23	0.99±0.03	0.94±0.08	0.81±0.09 •	0.93±0.08 •	0.96±0.06
Tr31	0.99±0.02	1.00±0.01	0.99±0.01	0.99±0.01	1.00±0.01
Tr41	1.00±0.00	1.00±0.01	0.99±0.01	0.99±0.02 •	0.99±0.01
Tr45	1.00±0.00	1.00±0.01	0.92±0.04 •	0.98±0.02 •	0.99±0.01
Wap	1.00±0.00	1.00±0.00	1.00±0.00	1.00±0.00	1.00±0.00
Average	0.99	0.99	0.96	0.96	0.98

• significantly worse, and ○ better, comparing to DMNB.