CS534 Implementation Assignment 2: Naive Bayes

Amit Bawaskar, Michael Lam
EECS, Oregon State University

May 3, 2013

Abstract

In this assignment, we implemented the Naive Bayes classifier with the Bernoulli model and Multinomial model, and compared their performance.

1 Introduction

We implemented the Naive Bayes classifier to solve a document classification problem on the 20-newsgroup dataset. Two models were implemented and compared for performance: Bernoulli model and Multinomial model.

2 Naive Bayes

This section provides an overview of the Naive Bayes classifier and some of our implementation details.

2.1 Model

A Bayes classifier model learns the prior $p(y)$ and likelihood $p(\mathbf{x}|y)$ from the training dataset and uses Bayes rule to make inferences. In a Naive Bayes Classifier, we make the conditional independence assumption that a particular feature is unrelated to any other feature given the class label. This

means that the likelihood can be computed as $p(\mathbf{x}|y) = \prod_{i=1}^{d} p(x_i|y)$ so there is no need to estimate the joint distribution.

Despite the naive assumptions, Naive Bayes classifiers significantly reduce overfitting and complexity. They also perform reasonably well in many applications. We will evaluate its performance on the 20-newsgroup dataset.

2.2 Inference

Inference is performed by using Bayes rule with the learned likelihood and prior probabilities, and using decision theory to select the class that maximizes the posterior probability. The posterior probability is proportional to the product of the likelihood and prior by Bayes rule, i.e. $p(y|\mathbf{x}) \propto p(\mathbf{x}|y)p(y)$. Given features as input and the learned prior and likelihood probabilities, decision theory selects the class that maximizes the posterior probability.

In the Bernoulli model, the likelihood is given by

$$p(\mathbf{x}|y) = \sum_{i=1}^{|V|} p_{i|y}^{x_i}(1 - p_{i|y})^{(1-x_i)} \tag{1}$$

where $|V|$ is the vocabulary size, $x_i = 1$ if the word i is present in the test document and $x_i = 0$ otherwise.

In the Multinomial model, the likelihood is given by

$$p(\mathbf{x}|y) = \sum_{i=1}^{|V|} p_{i|y}^{x} \tag{2}$$

where $|V|$ is the vocabulary size and x_i is the number of word i appearing in the test document.

2.3 Learning

Learning the likelihood probabilities for each feature and class is done by maximum a posterior estimation. Equivalently this amounts to applying Laplace smoothing to the maximum likelihood estimator. The prior probabilities for each class is the maximum likelihood estimator for it.

In this project, the probabilities are estimated by counting training examples with some criteria. For instance, the prior in the Bernoulli model is computed by $p(y = k) = N_k/N$ where N_k is the number of documents with class k and N is the total number of documents. The likelihood for the Bernoulli model is computed by $p(x_i = 1|y = k) = N_{i|k}/N_k$, which is the fraction of documents of class k where feature (word) x_i appeared.

For the Multinomial model, the prior is computed by $p(y = k) = W_k/W$ where W_k is the total number of words in documents with class k and W is the total number of words in all documents. The likelihood for the Multinomial model is computed by $p(x_i|y = k) = W_{i|k}/W_k$, which is equal to the total number of word i in documents with class k divided by the total number of words in documents with class k.

Note that these likelihood probabilities are adjusted by using Laplace smoothing, which is described in the next section.

2.4 Implementation Details

In this project we operated with the log of probabilities in order to avoid underflow issues. That is, for every multiplication and division operation, we instead used addition and subtraction of the log of the operands. We also stored the log of probabilities. For decision theory, we simply selected the class that maximizes the log of the posterior probability since the log function is a monotonically increasing function. Thus we directly operated with the log of probabilities in every circumstance.

We also applied Laplace smoothing to the likelihood probability of each feature and class in order to assign a default prior to words that have not been encountered. For the Bernoulli model, the likelihood becomes the following: $p(x_i = 1|y = k) = (N_{i|k} + 1)/(N_k + 2)$. For the Multinomial model, the likelihood becomes the following: $p(x_i|y = k) = (W_{i|k} + 1)/(W_k + |V|)$ where $|V|$ is the vocabulary size.

3 Bernoulli Model

In the Bernoulli Model, we checked if the word is present or absent in the document. The overall test accuracy for the Bernoulli model is **62.3984%**. Figure 1 shows the confusion matrix.

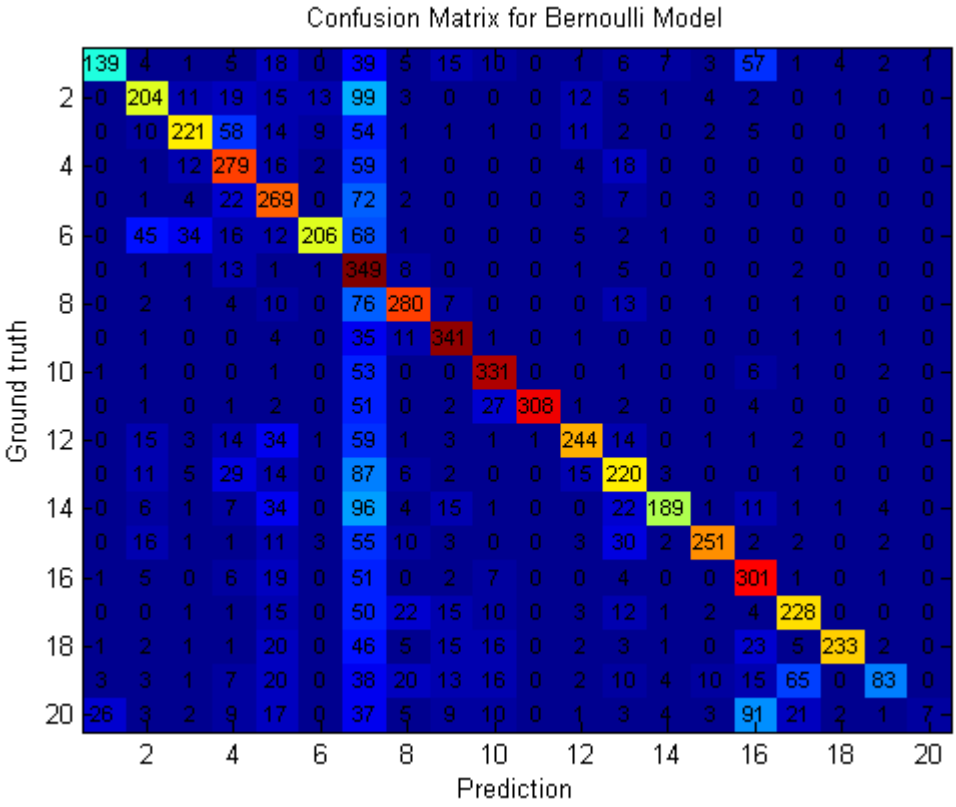

Figure 1: Confusion matrix for the Bernoulli model. The ith row and jth column represents the total number of times that a class i document is classified as class j.

4 Multinomial Model

In the Multinomial Model, instead of only looking at the presence or absence of the word, we looked at the number of times the word appeared in each document. The overall test accuracy for the Multinomial model is **77.8947%**. Figure 2 shows the confusion matrix. Overall the Multinomial model seems to perform better than the Bernoulli model in terms of classification accuracy.

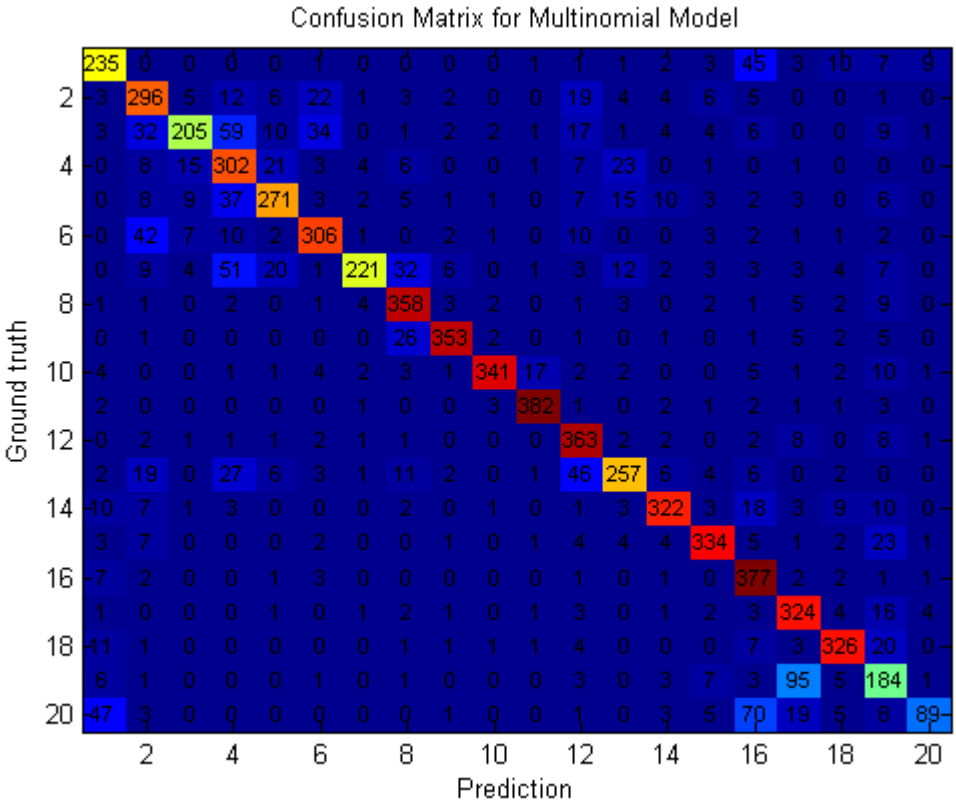

Figure 2: Confusion matrix for the Multinomial model. The ith row and jth column represents the total number of times that a class i document is classified as class j.

5 Heuristics for Reducing Vocabulary Size

We explored one heuristic for reducing the vocabulary size in hopes of improving classification accuracy. We tried eliminating words in the dictionary that have length less than or equal to a particular threshold value. We explored thresholds from 0 to 5, meaning from keeping all words to keeping only words with length greater than 5.

The results in Figures 3 & 4 indicate that the classification accuracy decreased as the threshold increases after 2 letters. Interestingly for the Multinomial model, accuracy increased negligibly when eliminating words up to 2 letters long. There is some intuition behind these results: it seems

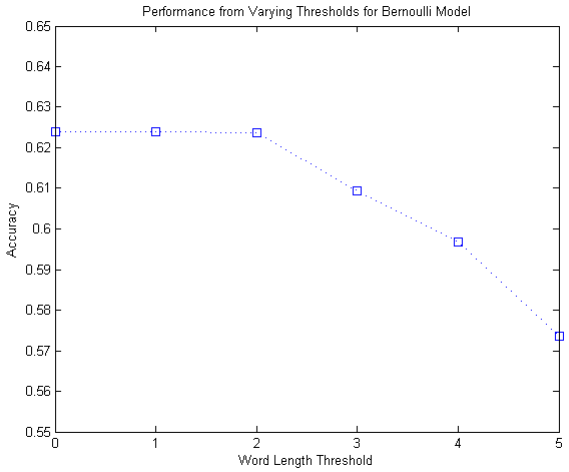

Figure 3: Classification accuracy using the Bernoulli model and varying word length thresholds. Words of length less than or equal to the threshold value are omitted in the vocabulary for learning and inference.

that words of length 3 or greater play an important role in determining the document class, while words of length 2 or fewer play a lesser role by perhaps tending to be generic English words.

5.1 Future Work

There are several directions for other possible heuristics. One thought is to manually create a blacklist of commonly used words to omit from the final vocabulary. The blacklist can include words such as articles, conjunctions, prepositions, etc. However, the drawback to this approach is having to manually construct this blacklist.

Another thought is to calculate the average count of all the words in a document and omit those words for learning in the multinomial model that occur more than the average count. This would remove words which occur "too many times" in the document. It would be interesting to see if this would help or hurt performance. Besides the average count, one could also compute various percentiles of counts and eliminate words above that percentile. However, the percentile would be another threshold/parameter value to tweak.

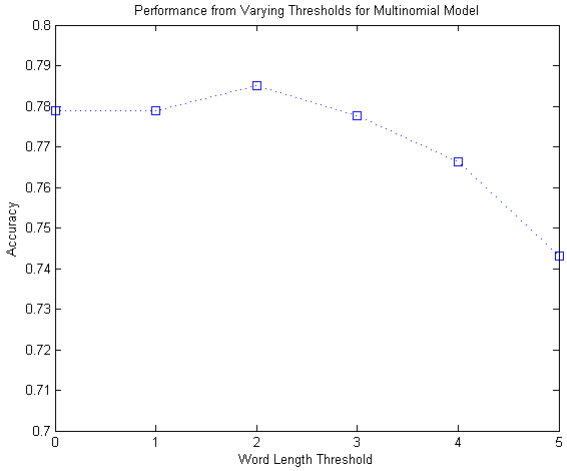

Figure 4: Classification accuracy using the Multinomial model and varying word length thresholds. Words of length less than or equal to the threshold value are omitted in the vocabulary for learning and inference.