CS 280
Programming Assignment 1
Naive Bayes Spam Filter

Helbert Paat

University of the Philippines
hapaat@up.edu.ph

October 6, 2020

I. Naive Bayes Classifier

Classifying emails according to whether they are unsolicited commercial (spam) email or non-spam email is one example of a broader set of problems called text classification. Consider representing an email via a feature vector x. The set of words encoded in the feature vector is called the vocabulary. If an email contains the j-th word of the vocabulary, then $x_j = 1$; otherwise, $x_j = 0$ for $j = 1, 2, ..., d$ where d is the size of the vocabulary.

To model $p(x|\omega)$, the Naive Bayes assumption makes a very strong assumption that the $x_j{}'s$ are conditionally independent given ω. The resulting algorithm, Naive Bayes Classifier, assigns an unknown email $x = [x_1, x_2, ..., x_d]^T$ to the class

$$\omega = arg \max_{\omega_i} \prod_{j=1}^{d} p(x_j|\omega_i) \tag{1}$$

Using Bayes' Rule, we can calculate the probability of a document being spam or ham given set of features using the following formula:

$$P(\omega|x_1,x_2,...,x_d) = \frac{P(x_1, x_2, ..., x_d|\omega)P(\omega)}{\sum_\omega \prod_{i=1}^{d} P(x_i|\omega)P(\omega)} \tag{2}$$

Using Bag of Words Feature Independence, equation (2) becomes

$$P(\omega|x_1,x_2,...,x_d) = \frac{\prod_{i=1}^{d} P(x_i|\omega)P(\omega)}{\sum_\omega \prod_{i=1}^{d} P(x_i|\omega)P(\omega)} \tag{3}$$

Consider a training set $\{(x^{(i)}, y^{(i)}; i = 1..., n\}$ of n emails. By the Maximum Likelihood Estimation, we can calculate $P(x_j|\omega = S)$ and $P(x_j|\omega = H)$ using the following equations:

$$P(x_j|\omega=S) = \frac{\sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge \omega^{(i)} = S\}}{\sum_{i=1}^{d} 1\{\omega^{(i)} = S\}} \tag{4}$$

$$P(x_j|\omega=H) = \frac{\sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge \omega^{(i)} = H\}}{\sum_{i=1}^{d} 1\{\omega^{(i)} = H\}} \tag{5}$$

Moreover, we can also calculate the prior probabilities using the following equations:

$$P(\omega=S) = \frac{\sum_{i=1}^{n} 1\{\omega^{(i)} = S\}}{n} \tag{6}$$

$$P(\omega=H) = \frac{\sum_{i=1}^{n} 1\{\omega^{(i)} = H\}}{n} \tag{7}$$

II. Methods

In this paper, a Naive Bayes classifier was designed for a subset of the TREC06 Public Spam Corpus which is a dataset for benchmarking spam algorithms. The following are the steps to implement the algorithm:

i. Create a function that splits the data into training and test set where 70% of the data goes to the training data and the remaining 30% of the data goes to the test data.

ii. Preprocess the email text files by removing punctuation marks, converting all the characters into lowercase characters to identify similar words with different text cases uniquely, and tokenizing the messages in the dataset by matching whitespaces between words as delimiter. .

iii. Create a vocabulary of words using the words that appear in the emails in the training data. Through this, features are extracted from the emails. In this paper, unigrams are the features in this task of text classification.

iv. Represent every email in the training and test set as vectors of the words in the vocabulary with the number 1 as an indicator if the word is present in the email and 0 otherwise.

v. For every word in the vocabulary, separately count how many spam and ham emails in the training data contain that word.

vi. Compute the prior probabilities using equations (6) and (7).

vii. To avoid loss of precision, instead of multiplying the likelihoods, the logarithms of the likelihoods are added and the result is exponentiated. Thus, equation (3) becomes

$$P(\omega|x_1, x_2, ..., x_d) = \frac{u}{v} \tag{8}$$

where
$u = exp(\sum_{j=1}^{d} log(P(x_j|\omega)) + log(P(\omega)))$
$v = \sum_{\omega} exp(\sum_{j=1}^{d} log(P(x_j|\omega)) + log(P(\omega)))$

viii. We calculate the conditional probabilities needed in equations (3) using equations (4) and (5). Let $a_{\Omega j} = \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge \omega^{(i)} = \Omega\}$ and $b_{\Omega j} = \sum_{i=1}^{d} 1\{\omega^{(i)} = \Omega\}$

$$P(x_j = 1|\Omega) = \frac{a_{\Omega j}}{b_{\Omega j}}$$
$$P(x_j = 0|\Omega) = \frac{b_{\Omega j} - a_{\Omega j}}{b_{\Omega j}}$$

where
$a_{\Omega j}$= number of emails belonging to class Ω that contain the word x_j
$b_{\Omega j}$= total number of emails belonging to class Ω

ix. Using **Lambda Smoothing**, we have the following formulas:

$$P(x_j = 1|\Omega) = \frac{a_{\Omega j} + \lambda}{b_{\Omega j} + \lambda \cdot v}$$

where v is the vocabulary size and λ is a positive constant.

x. For each email in the test data, a tuple (m, n, o) is created where

m is the index
n is the true label
o is the probability that the email is spam given its set of features

The probability is calculated using equation (8). Note that the email is spam if $o > 0.50$.

xi. Based on this tuple, a counter is also defined where the following are calculated:

TP or True Positive is the number of emails with true label *spam* and $o > 0.50$.
TN or True Negative is the number of emails with true label *ham* and $o < 0.50$.
FP or False Positive is the number of emails with true label *ham* and $o > 0.50$.
FN or False Negative is the number of emails with true label *spam* and $o < 0.50$.

xii. Functions that compute the precision and recall measures have also been defined. The following are the formulas for precision and recall:

$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN}$$

III. Results

The experiment is conducted using the TREC06 Public Spam Corpus for training and testing. The dataset contains 37,822 email messages where 24,912 are spam messages and 12,910 are ham messanges. For all the experiments, the prior probabilities are the following:

$$P(\omega = S) = 0.6619$$
$$P(\omega = H) = 0.3381$$

i. No Smoothing

Using vocabulary of size 1000, an experiment on the Naive Bayes Algorithm without using Lambda Smoothing was conducted . The disadvantage of doing this is the following: consider a particular word in the vocabulary that appeared only in ham emails in the training set but not in any spam emails. Hence, the posterior probability that that word appears given that the email is a spam is 0 ($P(x = 1|S) = 0$). If that word appears in an email in the test set, it will automatically categorize that as a ham because the zero posterior probability zeroes out the numerator of equation (3) when ω =S. Thus, $P(S|x_1, x_2, ..., x_d) = 0$ and consequently, $P(H|x_1, x_2, ..., x_d) = 1$. On the other hand, consider a particular word in the vocabulary that appeared in all the spam emails in the training set. Hence, the posterior probability that that word appears given that the email is a spam is 1 ($P(x = 1|S = 1)$ and the posterior probability that that word did not appear given that the email is a spam is 0 ($P(x = 0|S = 0)$. If that word did not appear in an email in the test set, it will automatically categorize that as a ham because the

λ	*Recall*	*Precision*
2.0	89.65	96.90
1.0	89.88	96.99
0.5	90.17	96.96
0.1	90.57	96.83
0.005	90.77	96.80
0 (*no smoothing*)	90.77	96.80

Table 1: *Comparison of classification results using different values of λ. In the experiment, the vocabulary size is 1000.*

zero posterior probability zeroes out the numerator of equation (3) when ω =S. Thus, $P(S|x_1, x_2, ..., x_d) = 0$ and consequently, $P(H|x_1, x_2, ..., x_d) = 1$.

The last row of Table 1 shows the classification performance of Naive Bayes Algorithm without lambda smoothing. Observably, it is still able to achieve high precision and a lower recall. With the very large dataset and fixed size of the vocabulary of only 1000, the disadvantage mentioned had little impact. The vocabulary extracts words from a large dataset of emails. The 1000 words in the vocabulary are the most frequent words that appear in all the emails in the training dataset. It is rare that there are words in the vocabulary that are not in both class. Moreover, based from the experiment, there are only 4 words in the vocabulary that are present in all the spam emails and 2 words that are present in all the ham emails in the training dataset.

ii. Laplace Smoothing: Finding the Best Value of λ

In this experiment, the model used different values of λ ($\lambda = 2.0, 1.0, 0.5, 0.1, 0.005$) to see which value of λ results in the best value of precision and recall. Table 1 shows the values of *Recall* and *Precision* for each λ.

Observably, the values of precision and recall scores for each of these λ's are very close, but no single value of λ yields the best value of precision and recall. However, $\lambda = 1$ is chosen to be the best among these values of λ because it resulted in the best value of precision score. A better precision at the expense of recall is considered because precision is a measure of safety. Higher precision means fewer ham messages are misclassified [1]. It is more desirable to have spams in the email box that to incorrectly classify hams as spams.

f	*Recall*	*Precision*
100	81.44	95.19
200	88.36	93.69
300	95.05	91.77
500	93.21	89.42

Table 2: *Comparison of classification results using different values of f, the number of most frequent words removed. In the experiment, the vocabulary size is 200 and words occuring less than 3 times have been removed.*

iii. Removing Frequent and Infrequent Words in the Vocabulary

The vocabulary used in the previous experiment consists of 1000 words, and common English words like "the, a, of" which are found in both spams and hams dominate the dictionary. In the paper of Havold [1], the most informative words for spam filtering have been identified using attribute selection strategies. Words in the vocabulary have been chosen by removing the top 200 frequent words and removing words occurring less than 3 times. Hence, different values for the number of most frequent words (f) to be removed from the vocabulary have been tested while removing words occurring less than 3 times. In this experiment, $\lambda = 1$ and 200 was chosen to be the size of the dictionary. Table 2 shows the results of this experiment.

The result of this experiment confirmed Havold's choice of removing 200 most frequent words resulted in a considerable value of precision while retaining a higher recall level. However, removing words occurring less than 3 times had no impact on the results because the vocabulary size has been fixed to 200 and the least frequent word in this vocabulary has 1650 occurrences. Additionally, it is observable that the majority of the classification scores, particularly precision scores, of this experiment are lower than the first experiment because we have only considered a much lower vocabulary size of 200 compared to 1000 of the first experiment.

Using a vocabulary size of 1000, removing the 200 most frequent words resulted in a precision score of 97.09 and recall of 94.78.

iv. Mutual Information

Another attribute selection strategy mentioned in Havold's paper is to rank attributes using Mutual Information [1]. In this experiment, the 200 highest scoring attributes have been considered as the vocabulary. The Mutual Information score is defined as

$$\sum_{x\in\{0,1\}}\sum_{c\in\{\omega\}} P(x,c)log\frac{P(x,c)}{P(x)P(c)} \tag{9}$$

Lambda smoothing where $\lambda = 1$ has also been applied to equation (9) to avoid errors when $P(x,c) = 0$. Hence, equation (9) becomes

$$\sum_{x\in\{0,1\}}\sum_{c\in\{\omega\}} P(x,c)log\frac{P(x,c)+1}{P(x)P(c)+v} \tag{10}$$

where v is the vocabulary size.

In this experiment, precision score of 96.11 and recall score of 83.20 have been achieved. Comparing this to the previous result when the same vocabulary size of 200 has been considered and the 200 most frequent words have been removed, it is evident that the precision score is better but the recall score is worse.

IV. Conclusion and Recommendation

In this paper, it has been shown that Naive Bayes Algorithm is able to do well in classifying whether an email is a spam or a ham. With large dataset such as the TREC06 Public Spam Corpus, the difference in classification performance of Naive Bayes algorithm with or without Lambda Smoothing is small. Moreover, the precision and recall scores can be boosted by attribute selection strategies of removing the 200 most frequent words in the vocabulary. It has also been shown that choosing few number of words with the highest mutual information scores to be the vocabulary resulted in a high precision score but undesirable recall score.

For future studies, the attribute set can be extended with n-grams ($n = 1, 2, 3$) as done by Harold [1]. A simple weighting scheme may also be applied to the posterior probabilities to improve precision. Additionally, more words with the highest Mutual Information scores can also be considered in the vocabulary. On the implementation, texts can be lemmatized as part of the preprocessing stage. The model can also be trained on equal number of spam and hams in the training set. Different proportions of the train and test split can also be considered with a much larger proportion for training set since the dataset is huge. Moreover, larger vocabulary size can also be tested to see its impact on classification scores and the impact of removing words occurring less than n times ($n = 1, 2, ..., 15$) on classification performance and speed.

References

[1] Johan Havold, Naive Bayes Spam Filtering using Word Position Attributes, In *Conference on Email and Anti-Spam*,, Stanford University, 2005.