Scarlett Townsend

**Homework #3 - Filtering spam SMS messages using Naive Bayes**

**Introduction**

The naïve Bayes algorithm is a simple application, available as a package within R, which uses Bayes’ theorem for classification.

By obtaining the prior probability (the probability a given message is spam based on a observed data set) and likelihood (the probability a specific word is observed given the message is spam) we can obtain the joint probability of a test word and spam occurring in a given message which, when divided by the marginal probability (the probability the word is observed in general) will allow us to determine the posterior probability (the probability a message is spam given a specific word is observed).  Determining this for multiple words and establishing probabilistic thresholds for classification can allow test instance to be characterized.  Bayes’ theorem is given by:

P(spam|word) = P(word|spam)\*P(spam) / P(word)

In order to implement this model, the observed probabilities are easily computed using a frequency table which simply counts the occurrences the words in spam and non-spam , or “ham,” training data messages.  These are then applied to each message using the class-conditional independence assumption and an overall probability of spam is calculated for each message.  Imposing a threshold probability requirement allows for the final classification.

While this algorithm is simple and effective it does make a simplifying class-conditional independence assumption which means events can be considered independent as long as the events are conditioned on the same class.  This may lead to slightly less accurate probabilities however, despite this, the predicted classes are generally fairly accurate.  In order to become familiar with and evaluate the efficacy of this algorithm, it was applied to two distinct data sets.

**Step 1: Collecting the data**

Data for the spam filtering example was obtained from the Department of Telematics, School of Electrical and Computer Engineering at University of Campinas, Brazil.  It contains 5,559 messages of which 4,812 were classified as spam and 747 as ham.

**Step 2: Exploring and preparing the data**

To prepare the data, it was first transformed into a “bag of words.”  A corpus was created which contains all the text messages.  The words were then transformed into all lowercase.  Numbers, whitespace and punctuation were removed and finally stop words, common words such as “and”, “to” and “but”, were removed.  As an example, a message reading, “Am also doing in cbe only. But have to pay.” was simplified to, “also cbe pay”.

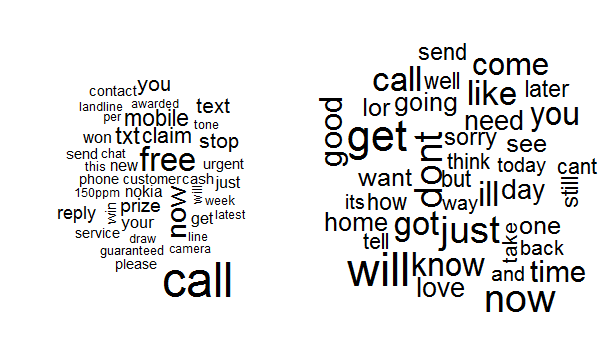
A sparse matrix was created in which the rows are for each document and the columns are the words discovered.  For this example there were over 7,000 columns.

Next the data set was divided into a training data set and a test data set.  This was done by using the first 4,169 records in the random sms dataset as a training set and the remaining 1,390 as a test set.  It was noted that each set contains approximately 13% spam.  The breakout is shown in the table below.

|  |  |  |
| --- | --- | --- |
|  | **Proportion Ham** | **Proportion Spam** |
| **Training Data Set** | 86.47% | 13.53% |
| **Test Data Set** | 86.83% | 13.17% |

The sparse matrix was simplified using the findFreqTerms command which allowed for the removal of words appearing less than 5 times in the data.  This reduced the words evaluated to approximately 1,200.

The wordclouds shown below were used to visualize the spam message contents from the training set (on the left) as compared to the ham message contents for the same set (on the right).



**Step 3: Training a model on the data**

To train the model, the Bayes algorithm was applied which estimated the probability that a given message is spam.  This was found in the e1071 package available for R from Vienna University of Technology.  It can be used to return either a predicted class or a raw probability based on the parameters selected.

**Step 4: Evaluating model performance**

The table below shows the performance of the test model classification.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Actual** | | | |
| **Predicted** | ham | spam | Total | |
| ham | 1203 | 32 | | 1235 |
| % of actual | 99.70% | 17.50% | |  |
| spam | 4 | 151 | | 155 |
| % of actual | 0.30% | 82.50% | |  |
| Total | 1207 | 183 | | 1390 |

You will notice that 4 of the 1,207 ham messages were misclassified while 32 or the 183 spam messages were, making the model very effective.  It is important, however to look more closely at the 4 incorrectly classified ham messages as falsely filtering out important emails could be far more detrimental than failing to identify spam.

**Step 5: Improving model performance**

One option in the algorithm available is to adjust the Laplace estimator.  Recall that the probabilities obtained in the training set are multiplied together when computing the likelihood of spam in the test set.  If a word never appears in a spam message in the training set then its likelihood given spam will be zero which will result in an overall zero probability of spam.  In effect, it allows a single word which appears in one type of message only to have an indisputable effect on the classification.

This can be changed so that the zero probabilities are instead substituted by a very small number which would allow the effects of all the other words in the message to impact the results.  The Laplace parameter does this in the algorithm given.  Changing this parameter results in the following results matrix:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Actual** | | |
| **Predicted** | ham | spam | Total |
| ham | 1204 | 31 | 1235 |
| % of actual | 99.80% | 16.90% |  |
| spam | 3 | 152 | 155 |
| % of actual | 0.20% | 83.10% |  |
| Total | 1207 | 183 | 1390 |

The number of false positives and false negatives were reduced. Changing the Laplace estimator further to 2, the results improve to:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Actual** | | |
| **Predicted** | ham | spam | Total |
| ham | 1205 | 38 | 1243 |
| % of actual | 99.80% | 20.80% |  |
| spam | 2 | 145 | 147 |
| % of actual | 0.20% | 79.20% |  |
| Total | 1207 | 183 | 1390 |

Looking at the three misclassifies false negatives, we can see that