Author Attribution

Brooks Beckelman, Zack Bilderback, Davis Townsend

Naive - Bayes Model

The first model that we will explore for predicting the author of an article based on textual content is a Naive Bayes Model. The first step is to create a document term matrix where each row represents a document, each column represents a word, and the values represent the frequency of each word in each document.

```
## <<DocumentTermMatrix (documents: 2500, terms: 31423)>>
## Non-/sparse entries: 425955/78131545
## Sparsity           : 99%
## Maximal term length: 36
## Weighting          : term frequency (tf)

## <<DocumentTermMatrix (documents: 5, terms: 3)>>
## Non-/sparse entries: 0/15
## Sparsity           : 100%
## Maximal term length: 5
## Weighting          : term frequency (tf)
##
##                                                    Terms
## Docs                                                aaa aah aampa
##   ReutersC50/C50train/AaronPressman/106247newsML.txt   0   0     0
##   ReutersC50/C50train/AaronPressman/120600newsML.txt   0   0     0
##   ReutersC50/C50train/AaronPressman/120683newsML.txt   0   0     0
##   ReutersC50/C50train/AaronPressman/136958newsML.txt   0   0     0
##   ReutersC50/C50train/AaronPressman/137498newsML.txt   0   0     0
```

The document term matrix contains 2500 rows (documents) and 31,423 columns (words). All of the words were converted to lowercase, and all numbers, punctuation, white-space, and stop words were removed. The first 5 rows and first 3 columns are shown above to give a sense of what we are working with. The matrix is very sparse, so let's remove some terms to make it more condensed.

```
## <<DocumentTermMatrix (documents: 2500, terms: 641)>>
## Non-/sparse entries: 180911/1421589
## Sparsity           : 89%
## Maximal term length: 18
## Weighting          : term frequency (tf)
```

The matrix is still very sparse, but by removing all terms that have a count of zero in greater than 90% of the documents, we were able to reduce the number of terms to 641. We felt that it was important to get the matrix as condensed as possible, while maintaining accuracy, because there are so many documents to be considered.

The next step is to create our model from the training data. In order to do this, we split the training set based on author and calculated a multinomial probability vector for each one. The probabilities associated with Aaron Pressman for the first five words are shown below.

```
##                        X1           X2           X3           X4
## AaronPressman 0.00937434 3.987384e-06 3.987384e-06 3.987384e-06
##                         X5
## AaronPressman 0.001997679
```

Now that we have our Naive Bayes model, we must check it against the test set. To do this, we created a new document term matrix. This matrix contains the same terms (columns) as the previous matrix but has 2500 new documents (rows). For each document, each author is assigned a score. This score is calculated by first taking the product of each term's frequency in that document and the log of the mulinomial probability for that author using that word. All of these products are summed for each author, and whoever has the highest score for that document is predicted to have written it. The first ten predictions are shown below.

```
##  [1] "AaronPressman"   "AaronPressman"   "SamuelPerry"
##  [4] "AaronPressman"   "SamuelPerry"     "AlexanderSmith"
##  [7] "DarrenShuettler" "AaronPressman"   "AaronPressman"
## [10] "AaronPressman"
```

Now, let's see how accurate our predictions were overall.

```
## The model predicted 1403 correctly out of 2500 for an accuracy of 0.5612
```

This accuracy is not as high as we had hoped. Let's look at how well the model performed for each individual author.

```
## Out of the 50 authors, the model predicted less than half of the documents correctly for 18 authors.
```

```
## The model predicted less than 25% of the documents correctly for 5 authors.
```

```
## The author that the model had the hardest time predicting was David Lawder with an accuracy of 0.12
```