Machine Learning the Authors of the Federalist Papers

https://github.com/JSunde/FederalistPapers

|  |  |
| --- | --- |
| Peter Giseburt | Jakob Sunde |
| [petergg@uw.edu](mailto:petergg@uw.edu) | jsunde@uw.edu |

**Abstract**

*The Federalist Papers were 85 political essays, originally printed under a pseudonym, Publius. Since then it has been discovered that the papers were written by Hamilton, Madison, and John Jay. However, some of the papers are of disputed authorship. To predict the authors of these papers, we will perform feature selection words and then run the algorithms: k-Means, k-NN, and Naïve Bayes to determine the authorship of the disputed papers.*

**1 Project description**

We examined the text of the Federalist Papers provided by Project Gutenberg. Then, using the known authors we converted the text into features and compared the features of known authors against the papers that are unknown or in dispute to determine the author.

Individual writers have individual styles, which can be represented in terms of how frequently certain words or phrases are used. From some research on the subject, certain “function words” are more indicative of a specific author, while other words, like nouns, can be highly influenced by subject matter and thus are not a good indicator of a specific writer [1].

Despite the main authors being Hamilton or Madison, John Jay is also credited with a few of the Papers. Therefore, we chose to implement non binary classifiers, as there are three potential authors. We chose implemented k-Means, k-NN, and Naïve Bayes classifiers as they allow for multiple classifications.

**2 Data Representation**

First, we gathered our data. We downloaded a .txt file of all the Federalist Papers from Project Gutenberg that contained all the papers, and parsed all of the word frequencies for all papers.

With the author label for each paper, and the frequencies of words in each paper, we create an average word frequency vector for each author, as well as a global word frequency vector for all papers. We then calculated the distance from this overall global word frequency vector to each of the respective author’s word frequencies. The difference from each author’s frequencies to the global vector are then summed to find the words that had the greatest difference in usage from the mean frequency. These are the words that have the greatest disparity in frequency between authors. Surprisingly, these words were mostly “function words,” despite us not filtering for specific words.

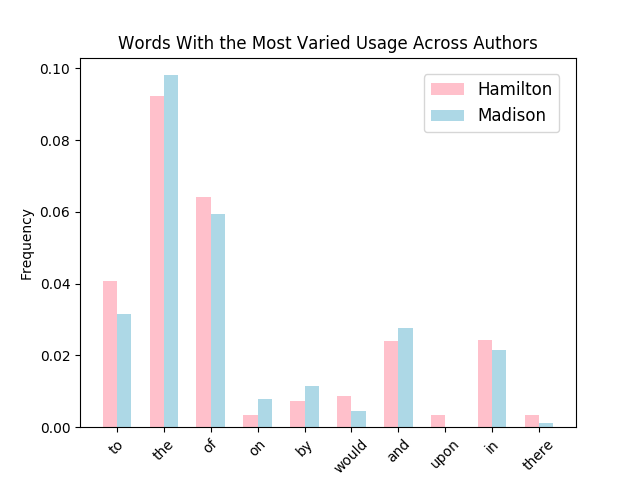
Hamilton and Madison collaborated on some papers together, and as such should have an overlapping writing style for these papers, which may skew, the word frequency we expect from each of the authors. Upon removing the papers that both Hamilton and Madison collaborated on, we received similar, slightly mutated data. It had a slight increase in errors for the training data, however for high N’s the model predicated 11 out of 12 to be Madison, which is close to the generally accepted result of all 12 being Madison’s. These joint papers were not included for the remainder of the project.

As mentioned in the abstract, John Jay also wrote some of the Federalist Papers, and initially we included him in both our feature selection and model prediction. However, we found that our algorithms never predicted Jay in our intermediate results. Furthermore, upon plotting the top words and the frequencies of these words for each author, we could see that John Jay’s data had a significance impact on which words were chosen as the biggest difference in frequency (the feature selection process is described below). Since Jay was never predicted, which is something previous research agrees on, we decided that including him was counterproductive finding words that provide a good indication if a paper was written by Hamilton or Madison, so all of our final results do not include the papers written by Jay.

**3 Feature Selection**

It is no secret that people use certain words more frequently than others. We performed feature selection with the goal of selecting words that indicated an author’s style.

We calculated the word frequencies of the words that had the greatest difference in usage between authors. This provides a way to represent the style of an author, by how frequently they use certain words in comparison to another authors writing. Nouns and other subject specific words are unlikely to appear in this set of words with the most varied usage. All the authors are writing on similar topics, so it is unlikely that one would use a specific noun much more often than another author. Other research agrees on the topic of feature words; nouns appear to be a function of the topic, while “function words” (like a, the, to, upon) seem to be a better indicator of *how* an author writes. This further solidifies the validity of this featurization of the data.



**4 k-Means**

The idea behind k-Means is to find a sample, or average paper for k clusters. If a disputed paper is closer to one of these averages than the other, then the author of that average likely wrote it.

First we compute an average Madison “paper” and an average Hamilton “paper”. We do this by computing the word frequencies of all of the feature words as described in Section 3 from all of the known papers of each author. Then we look at each of the disputed papers and compute the Euclidean distance between the disputed paper the two average papers. The disputed paper is labeled with the author of the average paper with the lowest distance.

|  |  |
| --- | --- |
| Predictions for Disputed Papers using Kmeans | C:\Users\Jakob\AppData\Local\Microsoft\Windows\INetCache\Content.Word\k-Means Training Error as a Function of Number of Feature Words.png |

We ran this implementation of k-Means varying the number of feature words and recorded the training error and author predictions. We then tested the known papers against average papers for different N values, finding a large amount of error starting at N = 1, which sharply dropped until N = 5 and flattened out. k-Means always predicts that a high number of the disputed papers were written by Madison, and with greater than 35 feature words, k-Means predicts that all 12 of the disputed papers were written by Madison.

In Glenn Fung’s paper, he talks about how his separating plane (with only 3 variables: to, upon, and would) predicted that all 12 disputed papers were Madison’s, and other work on the subject agrees with him [1]. However, he used a separating plane, which would incorrectly classify John Jay’s papers as either Hamilton or Madison and thus Jay’s papers were ignored. All three of their function words — “to, upon, and would” — appeared in our top feature words.

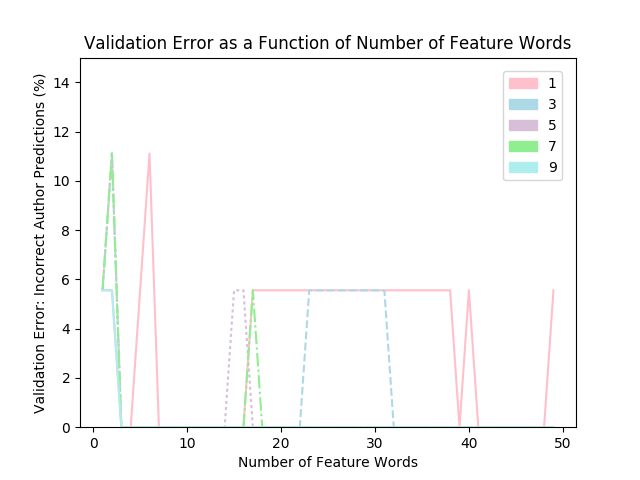
**5 k-NN**

k Nearest Neighbors looks at the k nearest neighbors and assigns the data point being examined to the class of the majority vote of those neighbors. In our case this means that for each dispute paper, we look at the closest papers, and assign the author of the disputed paper as whichever author wrote more of the k nearest neighbor papers. The intuition behind why this would correctly predict the author is that if an author tends to use certain feature words frequently, then their papers are more likely to show up as the nearest neighbors, which will result in k-NN predicting that author for the disputed paper.

In order to find the k closest papers, we calculate the sum of the difference between the frequencies of feature words from the disputed paper to all other papers. We then sort the papers by this difference and take the first k papers, resulting in the k closest papers to the disputed paper being classified. Then we classify the disputed paper with the author of the majority of the k nearby papers.

|  |  |
| --- | --- |
| C:\Users\Jakob\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Predictions for Disputed Papers using KNNKNN.PNG |  |

Again we ran this implementation while varying the number of feature words. We were surprised to find that k-NN (with k=9) is able to predict that Madison wrote all 12 papers with only one feature word. Simply looking at the frequency of the word “to”, k-NN comes to the same conclusion that previous research as agreed upon. The training error of k-NN quickly drops below 5% and even reaches 0% at some points.



In order to choose our value for k, we split Hamilton and Madison’s sets of papers in a 75% 25% split, with 25% of each authors papers forming a validation set. We than ran our implementation of k-NN with different values of k and looking at the 9 nearest neighbors resulted in the lowest validation error. Going further, we even found that despite a k value of 9 to have the lowest validation error, k values of 1 and 3 also predicted all 12 papers were written by Madison.

**6 Naïve Bayes**

Naïve Bayes computes the probability that a paper was written by a certain author. First we look at all of the papers in the training set and compute the probability that each word appears in a paper given it was written by the author of that paper. Then for each disputed paper we use the following equation to predict which author wrote it.

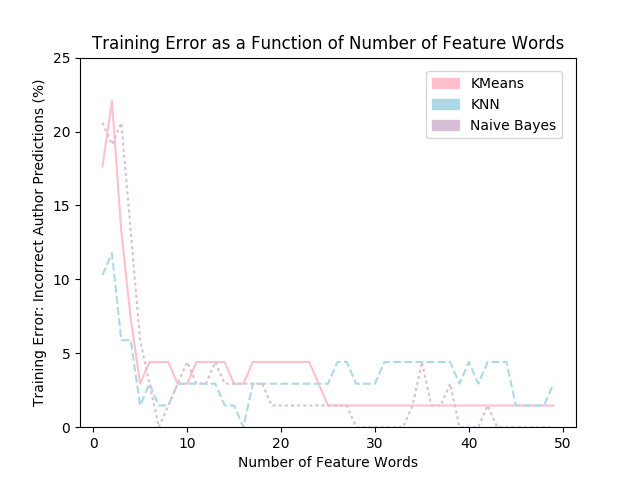
Where is the paper being examined, is the probability of any paper in the set being written by which is given by , and is the probability of a given word in existing in a paper written by .

This calculates the probability for each author that a given paper was written by that author, by multiplying the probability than any paper in the dataset was written by that author with the product of the probabilities of each word appearing, given it was written by that author. Then whichever author had the greatest probability is assigned as the predicted author of that disputed paper.

|  |  |
| --- | --- |
| ­­Predictions for Disputed Papers using Naive BayesNaive Bayes |  |

We then predicted the authors for all 12 disputed papers using Naïve Bayes and discovered that after about 16 feature words, it consistently predicted that all 12 papers were written by Madison. It’s interesting to note the different numbers of feature words each algorithm takes in order to reach that same conclusion, with k-NN taking only one word. Similarly to k-Means and k-NN, Naïve Bayes’ training error quickly drops below 5 percent as the number of feature words increases.

**7 Discussion of Results**



As seen in the graph above, all three of the graphs start with relatively high training error, although k-NN starts around 10% more accurate than the others, but they all sharply drop immediately to below 5% training error at around 6-7 words being considered for each paper. After that, the training errors fluctuate between 0 and 4%, all three algorithms dropping to 2% training error at one point or another, although k-NN appears to be highest for most of the data points. Past 25 words, k-Means stays at 2% error, while Naïve Bayes drops all the way to zero but also spikes upwards. Based on the fact that all three algorithms stay below 5% training error past about 6 words, while the algorithms don’t share the exact same performance they all predict authors incredibly accurately.

In a comparison, it is hard to say which algorithm actually performs best. All of the algorithms predict Madison wrote all 12 disputed papers, which agrees with previous research, so in terms of test error they all perform equally past a certain number of feature words. K-NN requires fewer words to make this accurate predication, so it is certainly less computationally intensive than the other algorithms. In that sense, one could argue that k-NN is “best”. Certainly, it is possible that because k-NN requires fewer words to predict all 12 disputed papers as Madison it is most accurate. However, when all algorithms predict the same thing it is impossible to say one algorithm is more accurate than another. To do this we would either need more test data, or to split up the disputed papers into smaller chunks so that we could calculate the percentage of chunks in a specific paper that are predicted wrong.

When combining our results for both predicting the disputed papers and our calculated training error, it is interesting to see how k-NN performs. It starts at a training error of about 10% for only one word, meaning predicting 12 out of 12 as Madison’s correctly is very possible. However, after 6 words its training error is not noticeably better (it is at times worse) than either of the other two algorithms, so with the training error it is still impossible to argue k-NN is more accurate than the others, despite its better accuracy for small numbers of words. Interestingly enough, while the training error for Naïve Bayes is below 5% after 6-7 words, the algorithm needs around 15 words before its predicts all 12 papers as Madison, even getting half of the papers wrong at 13 words.

It makes sense that k-NN and k-Means perform similarly to one another. Both are clustering algorithms that generally label test points as whatever is around them, although k-NN only looks as far as the k nearest neighbors, while k-Means looks at distance between the test point and the clusters.

TODO: Talk about the validity of the algorithms and their training error, why we think one had lower error than others, and which we think might perform best on papers that don’t have to do with the topic

**7 Discussion of Jay**

TODO: Talk about Jay’s irrelevance since we ran it with his papers included, but he was never predicted as the author in any of the algorithms

Mentioned above in data representation, do we want to move it here?

**8 Conclusion**

All of our algorithms classify Madison as the author of all 12 of the papers given enough feature words. This agrees with previous work on the subject. As argued earlier, when all of our algorithms predict the same thing, it is hard to argue any of them is better than the other in terms of accuracy. K-NN does predict them all accurately from even one word, and starts with a lower test error, but past 6 words they all have a similar, incredibly low test error of 4% or less. One can only argue that the algorithm is better in the sense that it requires fewer words to predict all 12 correctly, so it is less computationally intensive.

We performed our own method for feature selection, and found that our algorithm, which we described above, chose words that gave a good indicator to a writer’s style. Previous research filtered “function words” specifically, but our feature selection chose these similar types of words purely based on difference in frequency between the authors. When applied to the both the trained data and the test data, these words proved to be a good indicator of authorship.

**9 Related Work**

Fung and Bosch’s papers used support vector machine feature selection to create separating hyperplanes of 3 dimensions. These hyperplanes correctly classify all of the training data, and predict that all 12 of the disputed papers were written by Madison. Our findings support those of these research papers, while our Naïve Bayes model might be more easily applied to papers outside of the Federalist Paper dataset, as it outputs a probability of a paper being written by a certain author, whereas the hyperplanes simply classify a paper as being written Hamilton or Madison.

**References**

[1] Fung, Glen (2003). The Disputed Federalist Papers: Svm Feature Selection via Concave Minimization. In: *Proceedings of the 2003 conference on Diversity in computing*, pp. 42–46. ACM Press, New York.

[2] Bosch, R. A. and Smith, J. A. (1998). Separating Hyperplanes and the Authorship of the Disputed Federalist Papers. In: *American Mathematical Monthly*, pp. 105, 7 (August-September), 601–608.