Author Identification for General Addresses of the Church of Jesus Christ of Latter-Day Saints: A Multinomial Naive Bayes Implementation

Jeff Hansen
BYU / Ling 581
GitHub repo

Abstract

Multinomial Naive Bayes classifiers are useful for many classification tasks. Exploring one of these uses, this paper focuses on using it as an Author Identification tool for general address given by leader of the Church of Jesus Christ of Latter-Day Saints. Some exploration of the inclusion and exclusion of certain features like number of quotes and presence of scripture references in texts is presented with tables and some graphics. Finally, information is gathered and presented to demonstrate various overall conclusions about the usefulness of Multinomial Naive Bayes classifiers.

1 Introduction

Naive Bayes (NB) Classifiers are some of the simplest and most computationally efficient types of Machine Learning models. They are based on the Bayes Assumption that each feature inside the text is independent and unrelated to other features. Language generally demonstrates a linear flow of concepts and ideas that are directly connected, so at first glance the Bayes assumption does not seem to be applicable to language tasks. Ting et al. (2011) defend NB's ability to accurately accomplish classification tasks by comparing it to other Machine Learning models like decision trees, neural networks and support vector machines. They found that with the correct data preprocessing and removal and inclusion of specific words, NB classifiers generally outperformed these other Machine Learning models on various language classification tasks.

There are a couple of different Naive Bayes (NB) classifiers that follow different probabilistic distributions. A Bernoulli NB classifier assumes that the features only carry binary values. This is useful for problems where the presence or absence of some feature is notable. Gaussian NB models are based on continuous data that generally follows a normal distribution. These are useful for problems that have a tendency to follow a general pattern. The classifier then has an easy time of detecting things based on minute details in the distribution. A third type is the Multinomial NB model. This model uses discrete data (integers) that generally represent frequency counts or some type of rating on a certain scale. This paper utilizes a Multinomial Naive Bayes model since the frequency count of each word used in a document is a very strong indicator of author voice and style.

2 Related work

Multinomial Naive Bayes (MNB) Classifiers have been applied to many different classification tasks. Sentiment analysis is the task of assigning some emotion label to a certain text like happy, sad, positive, negative or neutral. Gamallo and Garcia (2014) implement a MNB classifier to classify twitter messages in similar categories. This is useful to gather general opinions regarding various social and political issues. Another use of Naive Bayes classifiers is spam detection. Eberhardt (2015) note that Bayesian models have been used to detect documents that are spam for years with much success. Many spam documents usually use higher frequency of words that catch readers attention, so the general MNB structure closely matches and picks up on this behavior. Sánchez-Franco et al. (2019) describe a third use for MNB classifiers. They create a MNB model that assesses the overall customer satisfaction of reviews for hotels. They hope that hotel owners can use this model's analysis to find ways to increase customer satisfaction.

As seen above, MNB can be applied to many different scenarios and classification problems. This paper specifically focuses on the Author Identification task of taking a document with an anonymous author and predicting who wrote that document.

Howedi and Mohd (2014) implement a MNB classifier to classify Arabic texts. They demonstrate that MNB classifiers are not only accurate for many different tasks, but they are also accurate when given limited amounts of data. This is a strong feature for a Machine Learning Model, since most models require large amounts of data.

3 Methods and procedure

The implementation of this project was done in a few steps. First, the original corpus was preprocessed to remove the outlying documents and reformat the text. Second, the classifier was created, trained, and tested. Third, the classifier was run a few hundred times with different settings to explore the strengths and drawbacks of a Multinomial Naive Bayes classifier.

3.1 Data preprocessing

The original corpus contains many documents that do not apply to this author identification task. Some of these outlying documents include statistical reports, funeral services, minutes for meetings, or general reports about things in the church. Searching for specific keywords in the text allowed the preprocessor to discard these documents.

Next, almost every single text inside the corpus contains miscellaneous headers that includes extra information regarding the text itself. Some of this information includes the authors name, their office in the Church, the title of their address, whether or not this talk was read by someone else, and other text that is extracted in tandem with extracting the text itself. Examples include:

> THE WORK OF REFORMATION A Discourse by Elder Orson Hyde. Reported by Unknown. Dear Brethren and Sisters ...
>
> President John Taylor Said: We are now commencing the ...
>
> Reaching Down to Lift Another See online Ensign, listen, download. Watch, listen, download. Let us ...

Regular expressions were used to remove these initial headers and information, thus leaving the relevant portion of the text for later training and testing.

Along with preprocessing the text itself, the author name also needed preprocessing. In the corpus, the author column generally does not only contain the author's name, but also contains the author's office in the church, abbreviations of their name, or different name formats. More regular expressions and other python string functions were used to extract the core name of the author.

Finally, since MNB classifiers utilize a "bag-of-words" format. The word counts are incredibly important. Since the most frequent words in each text are stop-words, these words are removed in order for the MNB classifier to focus on the more important meaning-bearing words. On top of removing stop-words, punctuation is removed, each word in the text was lemmatized, and the whole text was converted to lowercase. These last adjustments allow uniformity between each text.

3.2 Creating and training the classifier

Scikit-learn (Pedregosa et al., 2011) is a free software package that can be imported into any python project. It has many easy-to-use and powerful machine learning models that merely need training data. This paper utilized the `MulinomialNB classifier` as the core Machine Learning model. Along with this model, the `LabelEncoder`, `CountVectorizer`, `model_selection train_test_split` function, and the `metric classification_report` were also used from the scikit-learn library. These were used to format the data for training and testing and also presenting the results the of the classifier testing.

The training and test data was split up with 80% designated as training data and 20% designated as testing data. It is crucial that each author is found in both the training and test data, so instead of splitting up all of the texts 80-20 and all of the authors 80-20, each author had 80% of their texts devoted to training and 20% devoted to testing.

3.3 Testing the classifier on different hyperparameters

This paper explores the effects of changing various hyperparameters used by the MNB classifier:

1. number of authors trained in tandem
2. minimum author document count
3. presence and absence of certain referential keywords

The number of authors trained in tandem is relevant to the author identification task since there are often

many different potential authors that could have written a certain text. Ideally, MNB classifiers could be trained on many authors and still attain accurate results.

The minimum author document count is a hyperparameter that essentially just determines the minimum amount of data that the MNB classifier requires to give strong results. Many of the authors in the corpus only have a few talks attributed to them, while others have hundreds, so the MNB classifier will have a harder time selecting the less frequent authors since they will gravitate towards the authors with more documents attributed to them.

Following the pattern that frequency of words is a strong determiner of author voice, every single genre also includes other specific attributes that can help the MNB accurately identify authors. Howedi and Mohd (2014) demonstrate that including the the question mark in the MNB's count vector allowed them to get better results. Applying this general principle to religious texts from the Church of Jesus Christ of Latter-Day Saints, some speakers tend to cite more scriptures in their talks, while others opt to tell more stories with direct quotes. To help the MNB classifier recognize this information, all parenthetical citation with a scripture reference is converted to the keyword "`scripref`", and every opening quote symbol is converted to "`quoteopen`" and the close quote symbol is converted to "`quoteclose`".

Testing these three hyperparameters was performed in two groups: the author count group (testing the first hyperparameter) and the document count group (testing the second hyperparameter). Within each group there were five subgroups that tested various settings of keyword combinations included in the text. These five groups were: none of the keywords (`none`), just the open quote keyword (`quoteopen`), both open and closing quotes (`quotes`), just the scripture reference keyword (`scripref`), and all of the keywords (`ref_quotes`).

4 Evaluation and analysis

The scikit-learn (Pedregosa et al., 2011) `metric.classification_report` class function was utilized on the trained MNB classification model to print out the precision, recall, f-score, and accuracies of the classifier on each of the 36 most frequent authors. Table 1 demonstrate these results. Note that there was a 100 document-count minimum cap, and this model also included the `openquote` and `scripref` keywords to help the classifier in differentiating between authors. Later in this paper, there are other graphics and plots demonstrating varying results with different hyperparameters. The same sklearn report function was used to collect this data.

The results table (Table 1) demonstrates some interesting results. There are some very apparent correlations between Documents-Precision, Documents-Recall and Documents-F1-score. It appears that with less documents to train and test on, the MNB model tends to achieve higher precision and lower recall. It follows that with a lower document count, the MNB classifier has an easier time being precise, since there is likely less variation between these documents; on the other hand, when there are more documents, this implies that the author/speaker was held a leadership capacity for a longer period of time and likely touched on many different topics. This will cause the MNB classifier to have a harder time differentiating between this author and other authors. Figures 1, 2, and 3 better demonstrate these tendencies graphically.

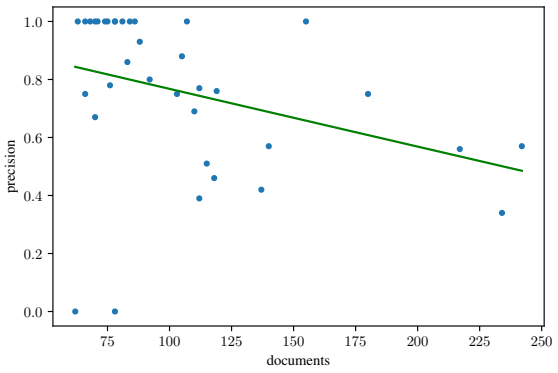

Figure 1: Correlation between number of documents and model precision for 36 authors using a second order polynomial best fit line. Correlation Coefficient: -0.336

A second order polynomial best fit line was used to find the general trend for each of these graphs (Figures 1, 2, and 3). Note that the document-recall and document-fscore tables have very apparent second order polynomial trends, but the document-precision graph is very linear even though it similarly used a second order polynomial best fit line. This is likely since the recall statistic is reliant more heavily on the author's writing and speaking style, while precision is purely dependent on the quantity of data provided. Also, note that the overall document-fscore graph demonstrates an even

Author/Label	Precision	Recall	F1-Score	Documents
Boyd K. Packer	1.0	0.71	0.83	107
Brigham Young	0.75	0.94	0.83	242
David O. McKay	0.97	0.95	0.96	180
Ezra Taft Benson	0.94	0.76	0.84	112
George Q. Cannon	0.88	0.9	0.89	140
Gordon B. Hinckley	0.7	0.98	0.82	234
Heber J. Grant	0.9	0.9	0.9	112
James E. Faust	1.0	0.5	0.67	103
John Taylor	0.93	1.0	0.96	155
Joseph F. Smith	0.75	0.68	0.71	137
Joseph Fielding Smith	1.0	0.68	0.81	110
Marion G. Romney	0.77	0.96	0.85	118
Orson Pratt	0.83	0.83	0.83	115
Spencer W. Kimball	1.0	0.55	0.71	119
Thomas S. Monson	0.86	0.97	0.92	217
Wilford Woodruff	1.0	0.74	0.85	105
accuracy			0.85	
macro avg	0.89	0.82	0.84	
weighted avg	0.87	0.85	0.85	

Table 1: The results of running the Multinomial NB Classifier on authors with more than 100 text documents attributed to them. This model was used with the openquote and scripref keywords added into the text, but it discarded the closequote keyword.

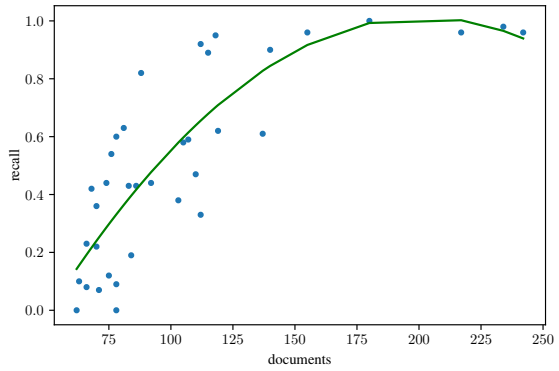

Figure 2: Correlation between number of documents and model recall for 36 authors using a second order polynomial best fit line. Correlation Coefficient: 0.753

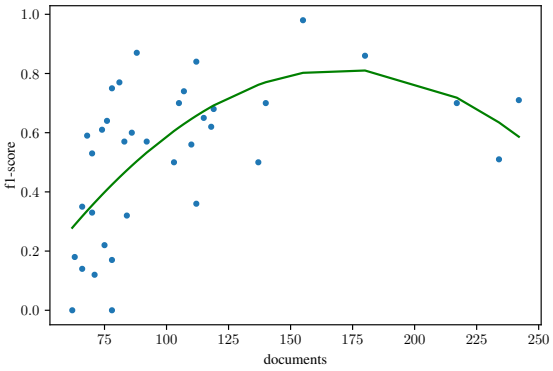

Figure 3: Correlation between number of documents and model F1-score for 36 authors using a second order polynomial best fit line. Correlation Coefficient: 0.454

stronger polynomial curve. This is because the f1-score is essentially a combination of the precision and recall, so adding these together causes the curve to be more negative for higher document counts and more positive on lower document counts.

4.1 Effect on accuracy of number of authors trained/tested together

Regarding the author count group, Figure 4 demonstrates that a lower number of authors trained and tested together results in very high model accuracies, while increasing the number of authors trained/test in tandem results in much lower accuracies. MNB classifiers generally have to select one of the y classifications for each text, so it is understandable that if there are less options for the classifier, it will be more accurate in predicting an author from a text.

Also, note the the lines on this graph are rather sporadic. This is likely caused by some authors being very similar to other authors. The test was run

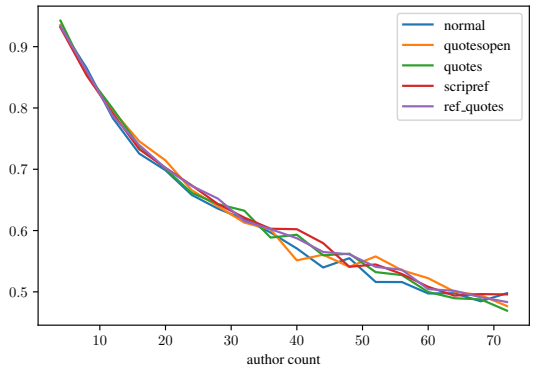

Figure 4: There is a general negative relation between count of authors trained together and the overall model accuracy

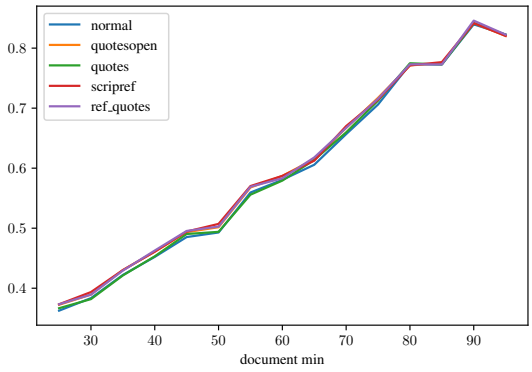

Figure 5: There is a tight positive relation between minimum document count for an author to be considered valid and the overall model accuracy

by selecting ten random groups of `n` authors, and then finding the average results of these 10 groups, so if the test happened to choose authors that were very similar, the accuracy would decrease; likewise, if the test happened to choose authors that are very different from each other (alive in different time periods, different offices in the church, different writing styles, etc.), the accuracy would improve.

4.2 Effect of document count on accuracy

As with any Machine Learning model, as more data is utilized by a MNB model, the accuracy improves. This is caused by the model being able to draw more conclusions from the training texts for each author. Ideally, there is a minimum amount of data that can be utilized to gain decent results. This would allow a model to make classifications on more authors. The following test was purposed to hopefully find some minimum amount of data that could be considered a cap for all other subsequent tests and uses of the MNB classifier. The original corpus utilized by this paper contains many different texts written by many different authors. Some authors only have 1 text document attributed to them, while other authors have over 200 documents.

Figure 5 demonstrates how accuracy improves as authors with less document counts are discarded from the training and testing sets. Note that there is a very strong positive correlation, and overall the results show a constant trend. These results help conclude that there is no ideal lower cap of document count that can prove useful for training and testing the MNB classifier.

4.3 Effect of keyword combinations on accuracy

As explained earlier, this paper aimed to improve the MNB classifier's accuracy by allowing it to also recognize how many scripture references and direct quotes a certain author used in their text. Figure 6 is a zoomed in version of the previous Figure 5 since this figure demonstrates the effect of each combination of keyword references really well.

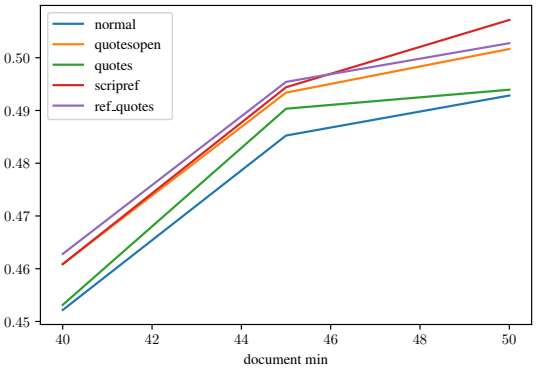

Figure 6: Zoomed in portion of Figure 5 to demonstrate the effect of each Keyword on the overall model accuracy

Though Figure 6 is a zoomed in version of the overall graph, the tendencies demonstrated in this zoomed in version are uniform throughout the whole graph. From the graphic, one can notice that the `scripref` (the red line), `ref_quotes` (the purple line), and `quotesopen` (the orange line) have a tendency to perform better than the control `normal` (blue) and the use of both `quotes` (green). Though there is a slight improvement in performance, the performace increase is only around 1%.

5 Conclusion

Overall, three properties of Multinomial Naive Bayes classifiers have been analyzed and evaluated: effect of number of authors trained together, minimum document amount per author, and the presence of keywords in the text that demonstrates the presence of direct quotes and scriptures references.

Regarding number of authors trained together, it has been concluded that MNB classifiers perform with much higher accuracy when there are fewer authors used in the training and testing process. This implies that when performing an author identification task with this type of classifier, it would be best to previously place the desired text inside a certain category of authors in order for the classifier to finish distinguishing between that select group of authors.

The minimum document amount per author tended to have the largest effect on the overall model performance. This implies that though a MNB classifier generally requires less training data than other Machine Learning Models, the data amount still has a very strong effect on model performance. With regard to the corpus used for this paper, authors with more than about 60-80 talks acted as the best training group for the classifier while including authors with fewer than 60-80 talks tended to decrease model accuracy significantly.

During the data preprocessing phase before training the MNB classifier, the open and close quote characters were changed to the `quoteopen` and `quoteclose` keywords. In similar fashion, every parenthetical citation that includes a scripture reference was changed to the keyword `scripref`. These keywords were then added into the count vectors used in the MNB classifier with the hopes to increase model performance. This supplementary information would provide the model with more information than just the normal word counts. In general the combination of the openquote and scripref keywords yielded the best results. Interestingly, including both the open and close quote keywords yielded the worst results. This implies that having both of these keywords proved to drown out the rest of the other words and keywords in the text.

6 Future Work

Since this paper has demonstrated that MNB classifiers perform best when they are trained on fewer authors, it would be beneficial to setup a multilayer network of smaller MNB classifiers. The structure would be similar to a tournament bracket. For example, let's assume that there are 64 potential authors that a certain document could be attributed to. Given 16 starting MNB classifiers, each being trained on four different authors from the set of 64, each classifier would label the document with a certain author. These 16 author results could then be divided into 4 MNB classifiers each trained on 4 authors from the set of 16. The document would be placed into each of these 4 classifiers yielding 4 results. A final MNB classifier could then be trained on these resulting four authors to give the network one final result. Since MNB models are computationally efficient this process of training new models as the document moves through the bracket will not be too costly.

Since the scripture reference keyword also proved useful, it would be interesting to divide this keyword into a few different subcategories that classify the reference to a certain book of canon like the Old Testament, New Testament, Book of Mormon, Doctrine and Covenants and Pearl of Great Price. It is likely that some speakers generally quote certain books of scripture more than others, so this added information could be informative for the MNB classifier.

References

Jeremy J Eberhardt. 2015. Bayesian spam detection. *Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal*, 2(1):2.

Pablo Gamallo and Marcos Garcia. 2014. Citius: A naivebayes strategy for sentiment analysis on english tweets. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014*. Citeseer.

Fatma Howedi and Masnizah Mohd. 2014. Text classification for authorship attribution using naive bayes classifier with limited training data. *Computer Engineering and Intelligent Systems*, 5(4):48–56.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Manuel J Sánchez-Franco, Antonio Navarro-García, and Francisco Javier Rondán-Cataluña. 2019. A naive bayes strategy for classifying customer satisfaction:

A study based on online reviews of hospitality services. *Journal of Business Research*, 101:499–506.

SL Ting, WH Ip, Albert HC Tsang, et al. 2011. Is naive bayes a good classifier for document classification. *International Journal of Software Engineering and Its Applications*, 5(3):37–46.