Authorship Identification Using Enron Email Dataset

PID: P26

Nirav Shah nshah28@ncsu.edu

Vrushanki Patel vpatel25@ncsu.edu

Sourabh Wattamwar sswattam@ncsu.edu

Department of Computer Science, North Carolina State University
Raleigh, NC 27606

1 Background

Authorship identification is a task of recognizing the most likely author of a given document. This is one of the most frequent and pervasive issues in the modern world. Authorship identification is crucial in a variety of contexts, including attribution of original authors to old and historical documents, the detection of book and novel forgeries, and the tracking down of anonymous social media abusers. As a result, it is critical to create algorithms that can assist in determining the writers of a piece of written content. Due to emails propensity to being used for fraud and crime, identification of emails is becoming more and more important. In this project work, we investigate and try to figure out who wrote emails by using the well-known Enron Email Data set.

1.1 Problem

In our project, we begin by defining the problem as a large multi-class, multi-label classification task, where each email can be addressed to a number of different users in the user's address book. After examining the problem of multi-class authorship attribution using multiple combinations of methods for feature extraction, different classification algorithms, and testing the scalability of the trained model, we will test the accuracy of authorship identification using distinctive writing styles of an author, and word embedding techniques across multiple classification algorithms. As we scale up the number of authors, we will evaluate how various models and algorithms perform.[6]

1.2 Literature Survey

Pramod Kumar Singh et al.[4] worked on patterns using different stylometric elements in the written text can be used to reflect an author's distinctive writing style.

Sarwat Nizamani et al.[5] study implies that selecting a feature set for the goal of detecting fraudulent emails is more important than selecting a classification model. They have extracted a variety of features, in order to compare how well each category of characteristics performed in terms of the rate at which fake emails were detected.

Ahmed Abbasi et al.[1] For the purpose of comparing the writing styles of anonymous authors and those of a group of well-known authors, suggested a method dubbed the writeprints approach, which was put into practice utilizing stylometric features. In addition to the fundamental stylometric features, we also used content-specific features and classifiers like Random Forests and Multinomial Naive Bayes.

In Authorship Attribution with Limited Text on Twitter by Luke Chen et al.[3], they offered a selection of algorithms for authorship attribution that pinpoint the author of Tweets. Lexical, syntactic, and semantic information are extracted using natural language processing techniques and fed into Naive Bayes, SVM, and neural network multiclass classifiers.

Leo Breiman et al. [2] in his work provided a well defined overview of the Random Forest classifier. After analysis, it was clear that it has better accuracy, relatively robust to outliers and noise. It is faster than bagging or boosting and it is simple and easily parallelized.

2 Methods

2.1 Approach and Rationale

2.1.1 Effectiveness of TFIDF Vectorizer on multiple models

TF-IDF (Term Frequency - Inverse Document Frequency) Vectorizer is a measure of originality of a word by comparing the number of times a word appears in each document in a collection the word appears in. It was invented for document search and information retrieval. The TF-IDF Vectorizer converts raw documents into a matrix of TF-IDF features.

Mathematically TF-IDF can be represented as:

$$tfidf(t, d, D) = tf(t, d) * idf(t, D)$$

where,

$$tf(t, d) = log(1 + freq(d)$$

$$idf(t, D) = log(N/count(d \in D : t \in d))$$

TF (Term Frequency) : The frequency of a term in the vocabulary of the dataset
IDF (Inverse Document Frequency) : The measure of how much information the word provides

The word is of high importance if the TFIDF score approaches to 1

2.1.2 Effectiveness of Count Vectorizer on multiple models

Count Vectorizer tokenizes the documents to build a vocabulary of the words present and counts how often each word from the vocabulary is present in the collection of documents. It is the way of converting text to vector. It constructs a matrix in which columns and text samples by rows represent words. The vectors are of binary type, which tells us if the word is present in the text sample or not.

2.1.3 Cluster Based Classification

K-means clustering is an unsupervised learning algorithm, originally from signal processing, that aims to divide n observations into k clusters. Each observation belongs to the cluster with the smallest distance to that particular cluster's centroid. To choose the appropriate value of K for clustering, the following methods exist, the elbow method and the silhouette method. In the elbow method, we calculate the Within-Cluster-Sum of Squared Errors (WSS) for different values of K and choose the K for which the WSS first starts to diminish. The WSS score is the sum of the squares of the distances of points from its predicted cluster center. In a plot of WSS vs K, this is visible as an elbow. In the silhouette method, the value measures the similarity of a point to its own cluster compared to other clusters.

The range of the silhouette value is between +1 and -1, and a high value indicates that the point has been placed in the right cluster. In the graph of Silhouette Values vs K, we would like to see a global maximum or peak which corresponds to the optimal K value.

2.1.4 Models Preferred

1) Multinomial Naive Bayes Classifier:

Multinomial Naive Bayes is a Bayesian learning algorithm, usually implemented in Natural Language Processing (NLP). Thus, naive bayes approach is a strong tool that analyzes text as an input and multiple class problem is been solved. As multinomial naive bayes is a statistical classification based on machine learning algorithms, which classifies instances into one, two, or three classes (binary classification).

2) Random Forest Classifier:

Random Forest Classifier is a meta estimator, that basically fits multiple classification trees, on various sub-samples of entire dataset, and at the final step it uses averaging in order to enhance the predictive accuracy. Using random forest classifier, the overfitting of the model can be controlled.

3) Decision Tree Classifier:

Decision Tree Classifier is a classification model that creates classification by building decision tree. As compared to random forest classification, decision tree easy to compute and implement. Further, importance of any particular variable can be simply obtained applying this model. In addition, decision tree can be implemented on mostly all types of dataset, specifically used for non-parametric data.

4) Support Vector Classifier:

A support vector machine classifier, is a deep learning based algorithm including supervised learning, for classification or regression applied on datasets in machine learning. The support vector classifier works as plotting the datapoints in high-dimensional space. Once the mapping is done the data is divided into two classes based on the optimal hyperplane obtained.

3 Plan and Experiment

3.1 Datasets

3.1.1 Folder Structure to Dataframe

The Enron Email data set provided is a directory structure with several folders like _sent_mail, _sent, and sent_items under directories named after different authors. The above folders contained all the emails sent by that particular author in separate files.

The objective was to create a data set containing folder names as the label and extract features from all the emails under the different sent folders for the author directory. Before extracting the features, we ensured that the emails were properly cleaned to remove any unwanted noise, and only the body from the emails was considered for feature extraction. We extracted all the emails for 20 authors with the most emails sent and created a rudimentary dataset which we stored as a CSV file called enron.csv. Looking at the graph in Figure 1, we realized that the distribution of sent emails amongst the 20 authors was not uniform, and random stratified sampling was required in all subsequent methods.

3.1.2 Extraction of Content-Specific Features

Content-specific features primarily describe the vocabulary of the words in the dataset that may be used to identify subjects in the email's body and specific terms that stand out because they are rarely used.

Before extracting the content-specific features, we did the following steps for text pre-processing:

1. Removal of all non-alphanumeric characters.

2. Lemmatization:
   The process of reducing the different forms of a word to one single form,
   For example, reducing "changes," "changer," or "changing" to the lemma "change".

3. Stopword Removal:
   Removing common words like a, an, the, of, etc., which are extremely common, and of no importance for classification.

After text preprocessing, we used TFIDF and Count Vectorizer for content-specific feature extraction.

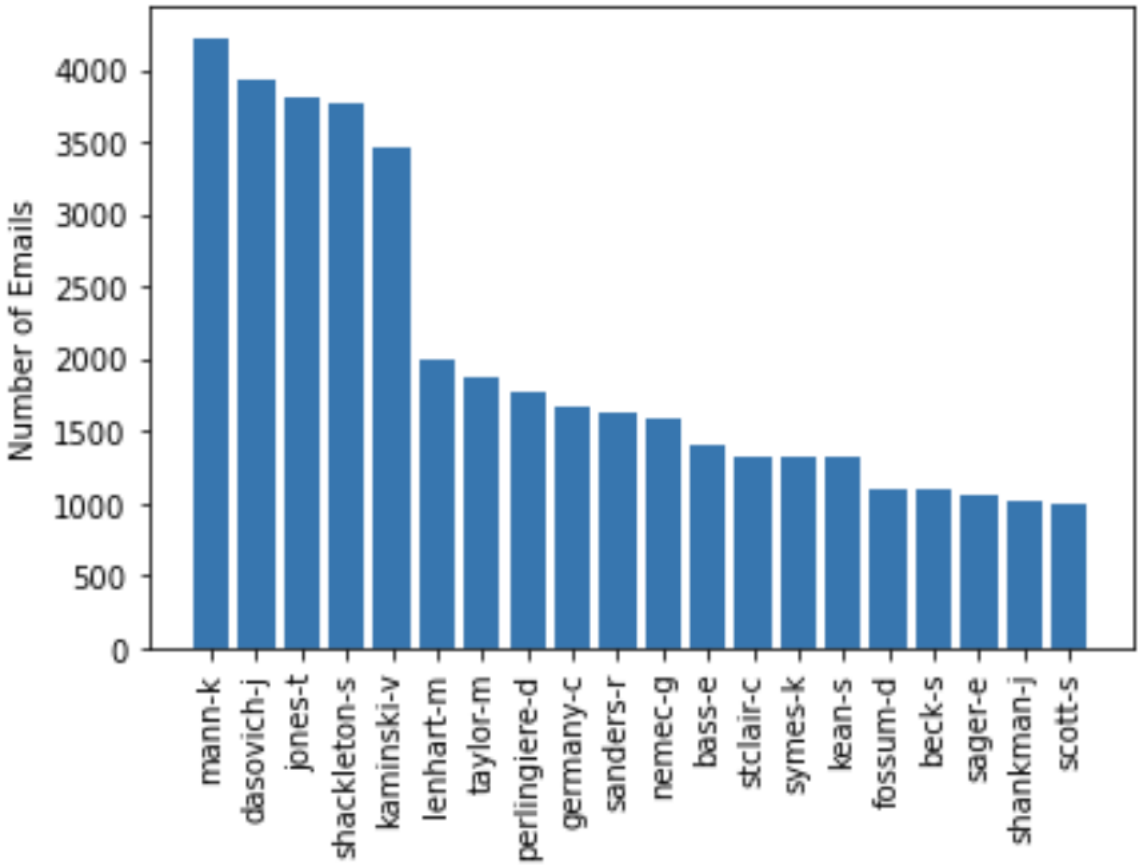

Figure 1: Caption

3.2 Hypotheses

The research objective for our project is that the TFIDF Vectorizer is better than Count Vectorizer for multiple classification models.

While the objective is simple, we believe it will help lay the foundation for future objectives. TFIDF Vectorizer should perform better than Count Vectorizer since it emphasizes the frequency of a word in the corpus, unlike Count Vectorizer.

The vectorizer TFIDF and Count both produce very high dimensional and sparse vectors. As the vectorizer creates dimensions depending on unique tokens in text, as the Email data set is quite large, it will generate high dimensionality. Also, the email content for different authors will be different, which makes the data sparse. We used Select k-best features from the dimensions to resolve the high dimensionality issue.

3.3 Experimental Design

To compare the accuracies of TFIDF and Count Vectorizer, we completed the text pre-processing. Since the content-specific features had high dimensionality, we did feature selection using Select K Best features. The optimal K value was selected by comparing the training accuracies for different values of K with Multinominal Naive Bayes as shown in Fig 2.

Later, we ran multiple classification models on both the techniques, and compared accuracies along with their F1 scores. We used Random Forest classifier, as it fits multiple number of sub divided data using various decision tree. In addition the prediction accuracy of Random Forest classifier is better than Multinomial Naïve Bayes classifier.

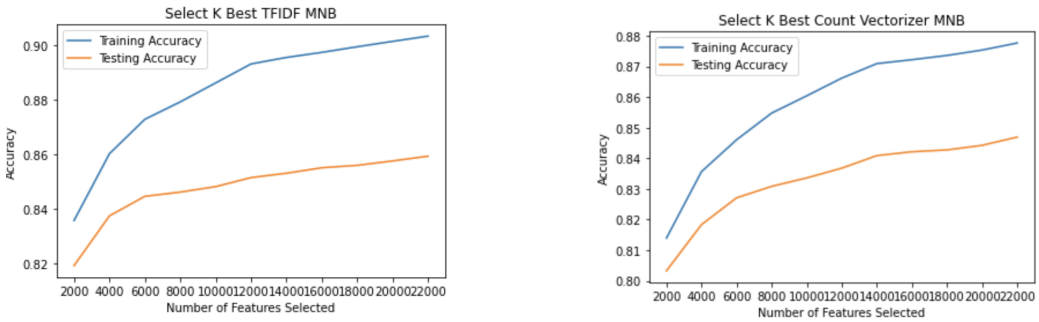

Figure 2: K-Best features for TF-IDF and Count Vectorizer Naive Bayes

Next, we implemented Decision Tree algorithm in order to visualize the level of importance of any particular variable holds, on the basis of which further analysis can be done. In addition Decision Tree classifier can handle mostly all types of data. In our case, the predicted testing accuracy is nearest to the Multinomial Naïve Bayes classifier.

We ran out dataset through Support Vector Classifier as for each category of the data, a labeled training set is generated, which can further be used to categorize new text (test dataset). Here we tested SVC using various kernels like linear, rbf, polynomial and sigmoid. As a result we found that SVC with linear kernel provides highest accuracy.

4 Results

While comparing Count Vectorizer and TFIDF Vectorizer for various models, we can see that TFIDF Vectorizer performs better than Count Vectorizer. The highest accuracies are seen for TFIDF Vectorizer with Random Forest Classifier. TFIDF being better than Count Vectorizer follows our hypothesis and is expected behavior since TFIDF considers both term frequency and inverse document frequency. While the accuracy of Random Forests and Support Vector Classifier with TFIDF are close together, the F1 score of Random Forest Classifier is better than SVC, and hence we felt that instead of solely depending on accuracy, we must consider F1 score metrics as well. Thus, Random Forest was the best performing model.

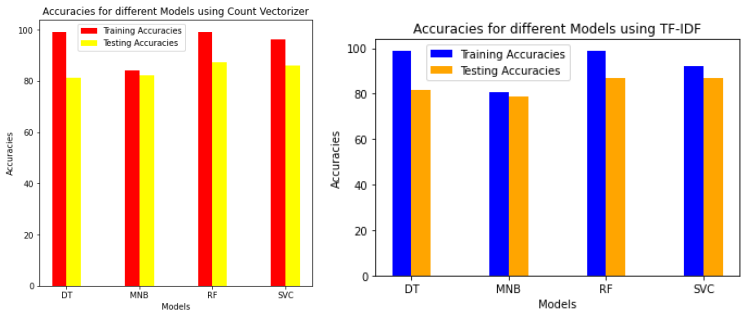

Table 1: Model Comparison

Comparison of Models using Custom Features and Count Vectorizer		
	Accuracy	F1 score
Multinominal Naïve Bayes	82.28	0.820
Random Forest Classifier	87.20	0.868
Decision Tree Classifier	81.35	0.809
Support Vector Classifier	86.14	0.861

Table 2: Model Comparison

Comparison of Models using Custom Features and TFIDF Vectorizer		
	Accuracy	F1 score
Multinominal Naïve Bayes	78.86	0.783
Random Forest Classifier	86.95	0.869
Decision Tree Classifier	81.44	0.812
Support Vector Classifier	86.83	0.868

5 Conclusion

After investigating the problem of multi-class authorship attribution using a different combination of methods for feature extraction and classification algorithm and analyzing the scope of the scalability of the trained model. We trained Multinominal Naive Bayes, Random Forest Classifier, Decision Tree Classifier, and Support Vector Classifier models using both the Count and TFIDF vectorizer, and using custom features, we have come to the following conclusions.

TFIDF Vectorizer with custom features using Random Forests provides the best F1 score and scalability of the model, the F1 score being 86.9% and Accuracy being 89.65%. While Count Vectorizer with custom Features using Random Forests performing at the following F1 score: 86.8% and accuracies: 87.20% are very similar to TFIDF with custom Features. As the accuracy are similar for both the models, we believe the TFIDF Vectorizer with Stylometric Features and by increasing the SelectKBest features is the better feature set to use for scalability reasons. We have implemented multiple models with using only 5 authors, we can scale them using more authors. Also utilizing GloVe, Word2Vec and Doc2Vec testing must be done across 5,10,15 authors to look into the scalability of these feature extraction methods.

References

[1] Ahmed Abbasi et al. "Writeprints: A Stylometric Approach to Identity-Level Identification and Similarity Detection in Cyberspace". In: *ACM Transactions on Information Systems,Volume 26,Issue 2* (2008). DOI: https://doi.org/10.1145/1344411.1344413.

[2] Leo Breiman et al. "Random Forests". In: (2001). DOI: https://doi.org/10.1023/A:1010933404324.

[3] Luke Chen et al. "Authorship Attribution with Limited Text on Twitter". In: (2017).

[4] Pramod Kumar Singh et al. "Stylometric Analysis of E-mail Content for Author Identification". In: *IML '17: Proceedings of the 1st International Conference on Internet of Things and Machine Learning* (2017). DOI: https://doi.org/10.1145/3109761.3109770.

[5] Sarwat Nizamania et al. "CCM: A Text Classification Model by Clustering". In: *2011 International Conference on Advances in Social Networks Analysis and Mining* (2011). DOI: https://doi.org/10.1109/ASONAM.2011.764.

[6] http://www.ai.sri.com/project/CALO. *Enron Email Dataset*. http://www.ferc.gov/. URL: https://www.cs.cmu.edu/~enron/.

6 GitHub Link

https://github.ncsu.edu/sswattam/CSC522-Fall22-P26