

Machine Learning Based Authorship Identification

Team Members

1. Zhenzi Dong (PennKey: zhenzi; Email: zhenzi@seas.upenn.edu)
2. Anqi Gao (PennKey: anqigao; Email: anqigao@seas.upenn.edu)

Section 1: Introduction

- **Problem Domain**

The writing style of different people varies in many different ways, like the frequency usage of words, the length of sentences and paragraphs and so on. They are not easy to distinguish with naked eyes but form interesting patterns under analysis.

Nowadays, author detection recurs among different aspects and fields, for example, helping judge plagiarism in academic field, identifying the writer of harassing letters or intimidation messages in social security field, and helping reveal author and hidden stories of historical heritage.

What's more, with the results of author detection, we will get the similarity between works among different authors, thus are able to give recommendations for readers without knowing exact author name or genre. With a short paragraphs of text, we can give the authors with corpus in the style.

With inspiration of the above topics, we plan to do authorship identification among 10 authors and give a simple recommendation of most relevant corpus by giving some paragraphs of certain author.

- **Problem Description**

In this project, we will mainly focusing on identifying the anonymous works or detecting the identity for those who used a pseudonym when writing. With dataset in Project Gutenberg, we will download corpus and novels of 10 authors, remove information of authors and randomize the order of novels. After data processing, we first extract 3 different linguistic features, lexical features, punctuation features and bag of words. Then with the 3 input features, we also build 4 supervised machine learning models, including Support Vector Machine(SVM), Neural Networks(NN), Naive Bayes and K-Nearest Neighbor(KNN).

In this way, with same features, we will analyze performance of different models on detecting authors of corpus. With same model, we can compare the results of different linguistic features, thus figuring out some appropriate and most helpful features for natural language processing. What's more, among various combination of features and machine learning models, we will compare the differences between different authors and figure out some similarities between certain authors.

Second, by giving paragraphs of articles, we would provide simple recommendation of corpus with most similarity. By computing and drawing confusion matrix, we will get the probability that novels of certain author being recognized to another author, those with high possibility will be recommended. And recommendations can be given based on different features, like preference words.

Section 2: Data Preprocessing

- **Data sources and quality.**

Our dataset of this project comes from the Gutenberg Project. We picked ten most popular authors (Jane Austen, Charles Dickens, Mark Twain, Oscar Wilde, Arthur Conan Doyle, Joseph Conrad, Robert Louis Stevenson, William Shakespeare, Lewis Carroll and Fyodor Dostoyevsky) from the websites and have all their works downloaded. Formats of those works including novels, short stories, letters, plays, poems and so on. With 366 books in total, we separated their works into chapters and finally have 3374 files in total. The quality of the data we downloaded is fairly high so that there's limited works to be done for the data cleaning.

- **Data acquisition and clean**

Data acquisition is quite straightforward. It took us a while to download all the txt files from the Gutenberg Project website.

Data cleaning process mostly includes the elimination of author's name that might appear in the txt file, and the elimination of the license that every file contains. The symbol of the beginning of each chapters and the license is so obvious that there will always be a line that looks different or 'abnormal' including symbols like '*'. So we wrote a program to achieve the cleaning and the separation process. Most of the works could be done by this and after that, we re-read the files in another program that counts lines of the files and order them by total number of lines from high to low. Then we simply check the first few files manually to ensure that number of chapters that each file contains is less than two.

- **Feature extraction**

We chose three separate linguistic features for this project, the Bag of Words, Punctuation and Lexical, each in a separate file. The extraction of the 'Bag of Words' feature including the process of scanning through all the data files and collection all the vocabulary in each file and generation of word frequency for each file. We have 113,991 words in total in our dictionary, so that we have 113,991 elements to represent each vector. The extraction of punctuation feature is actually the extraction three ratios for each file, the ratio between the number of commas and the number of sentences, the ratio between the number of semi-commas and the number of sentences and the ratio between the number of colons and the number of sentences. So that we have 3 elements to represent each vector. Likewise, the result of extraction of Lexical feature is also including 3 features per vector. The average

number of words per sentence, the variation of the sentence length and the lexical diversity, that is the number of distinct words that appear in the sentence to the total number words.

Then we shuffled the feature files (already linked to their labels) randomly and separate them into testing and training sets in ratio of 2:8.

Section 3: Implementation and Analysis

- **Models and Algorithm**

With three features chosen and extracted in data processing process, we build 4 machine learning models. Since many factors including the genres of corpus, length of text, amount of data and chosen features will influence our detection, we need to build more than one models possible for prediction and compare their performances.

- **Neural Networks**

Neural Network has been a popular method in recent years and it is known to be complex in structure and accuracy in classification.

First, input features are sent into the first hidden layer, then the hidden units extract nonlinear features from the input vector x by first linearly combining the input features, and then applying a nonlinear transformation to this linear combination. Then, same steps are applied to the following layers. Lastly, the outputs of final layer units are then passed through a softmax squashing function to produce estimates of the probabilities of the K classes.

- **Support Vector Machine**

Given an n -dimensional feature space, an SVM classifier aims at obtaining the $(n-1)$ dimensional hyperplane which classifies samples correctly. In other words, the SVM algorithm chooses the maximum margin classifier. That is, to solve the following dual problem:

$$\max_{\alpha} -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j (x_i^T x_j) + \sum_{i=1}^m \alpha_i$$

- **K Nearest Neighbor**

Intuitively, authorship detection of novels finds similarities ('distances') between or among corpus to determine who is the writer of the book. An author's writing pattern is unlikely to change very much, so that the effects of outliers should not be concerned too much. The classifier uses tBall Tree, KD Tree and brute-force search to compute the nearest neighbors, and choose the best suitable algorithm to the data to make predictions.

- **Naive Bayes**

This algorithm is commonly seen in applications of text classifications, and is the only algorithm that directly applies generative distributions to the dataset. This model features an important assumption of conditional independence among features given labels, which

intuitively similar to theory of extracting features bag of words. In this way, we only apply Naive Bayes models with integration of bag of words.

Algorithm can be expressed as following:

$$h^*(x) \in \underset{y \in [K]}{\operatorname{argmax}} \frac{p_y \cdot p(x|y)}{\sum_{y'=1}^K p_{y'} \cdot p(x|y')}$$

- **Description of the process of training and tuning the machine learning techniques**

In the first step, with features data and labels read in, data get shuffled then split them into non-overlapping 8:2 ratio. In this way, data are now randomly splitted into 80% as training samples and 20% as testing samples.

In the second step, we are building and tuning the 4 models to have relatively high test accuracy and better confusion matrices.

- **Neural Network**

For better performance of Neural Networks, data preprocessing package StandardScaler is first applied for data normalization before training.

Then we train multi-classes neural network model by scikit-learn MLPClassifier packages. I tested hidden layers from 1 layer to 10 layers and hidden units from 5 to 200 per layer and figured out model with 5 hidden layers and 100 hidden units per layer have better accuracy and performance. And the activation function 'tanh' works better than 'relu' while solver for weight optimization 'adam' is chosen, which works pretty well on relatively large datasets in terms of both training time and validation score.

- **SVM**

We try Support Vector Machine (SVM) model which applies multiclass classification using a one-versus-rest method. In this project, we use SVM classifier with radial basis function (RBF) kernel. Penalty parameter C from 1 to 600 and cache size from 1 to 300 are tested, which prove C=60 and cache_size=200 is determined to maximize the test accuracy.

- **K Nearest Neighbors**

While training the data, we use the scikit-learn KNeighborsClassifier package. After trying different value of n of neighbors from 1 to 50, we found the highest test accuracy with number of neighbor from 5 to 10.

- **Naive Bayes (Multinomial Model)**

The multinomial event model is suitable to our problem setting, so we use the scikit-learn MultinomialNB package. Due to the theory behinds the mode, so we only trained and tested the bag of words feature vectors on this model.

- **Authorship Detection Results**

	Bag of Words		Lexical Features		Punctuation	
	Train	Test	Train	Test	Train	Test
Neural Networks	0.9978	0.96	0.5172	0.5022	0.6162	0.5911
SVM (RBF)	0.9963	0.9185	0.4624	0.4756	0.5422	0.5421
KNN	0.8722 (k=5)	0.8044 (k=5)	0.4889 (k = 10)	0.5580 (k = 10)	0.5778 (k = 7)	0.6476 (k = 7)
Naive Bayes (Multinomial)	0.9567	0.9363	N/A	N/A	N/A	N/A

Table 1: Training and testing data accuracy of models and features

- **Models Comparison**

a) Neural Networks:

Under each feature, training and testing accuracy of Neural Networks are highest among the 4 models, with highest test accuracy with bag of words being 0.96. Neural networks is known to be complex in structure and accurate in classification, so that the model performs better by training with large raw features than simplified projected features.

b) SVM:

Results of SVM is not that good as Neural Network, which happens often in reality that deep learning method overcomes traditional machine learning methods in some problems. But SVM still achieves the best testing accuracy being 91.85% under bag of words feature.

c) K Nearest Neighbors:

As the key theory behind the KNN is the prediction by the averaging of labels over k nearest points around the new point, the result could be influenced by the number of instances in different labels. Since the number of instances in different labels are not balance in our data set, with the highest of 767 and the lowest of 107. Results are tentatively to be label 4 and that might cause some mistakes.

d) Naive Bayes:

We expected and as it turned out from the experiment that Naive Bayes (Multinomial) classifier will performed in the problem with feature being “bag of words”, because our data representation “bag of words” disregard the order of words, context of sentences and any other linguistics features, only keep the word frequencies. This characteristic is intuitively similar to the conditional independence assumption imposed in Naive Bayes. And it achieves 93.63% test accuracy.

- **Features Comparison**

- a) Bag of Words**

It can be seen from above table that models has best performance under feature of “Bag of words”, with 3 test accuracy larger than 90% and 1 at around 80%. The results prove that different authors have distinctive personal style in usage of words.

- b) Lexical Features**

It’s quite surprising that the test accuracy when applying the lexical features is not as high as the accuracy when applying the feature as bag of words. Then we draw a box plot on the elements of the feature as shown below. In order to make our result more concise, we would also calculate the T-score. T score is the difference between and within the two groups, the smaller the score, the more similar between the two groups.

As could be seen on the figure, the two labels that have biggest difference should be the label 3 and label 4 (as could be seen on the figure). When applying the T test, we could have their T value of 5.72 and that just means that the difference between the element of label 3 and element of label 4 is not that large. And we could say that the difference between our lexical feature that extract from our original data has not that much difference between each other. And I believe that might be the reason why the performance, or the accuracy that we have for the testing case it not as high as those we use the feature as bag of words.

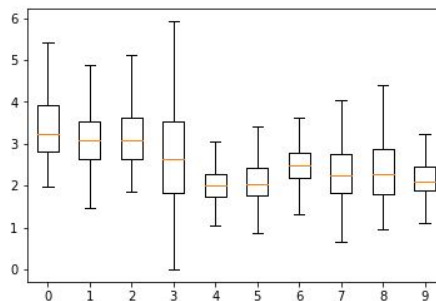


Figure 1. Box plot for one of the lexical features

- c) Punctuation Features**

The performance for the punctuation features is not that good as well. Then we also drew a box plot and that could be seen in the below figure. Actually the box plot looks quite similar as the one we drew above. In order to make our prediction looks more persuasive, we calculate the T value between label 3 and label 5 and the result is 7. The difference is still not that large.

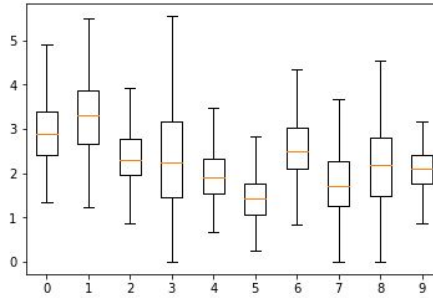


Figure 2. Box plot for one of the punctuation features

- Performance Measurement**

Here we choose confusion matrix as performance measurement, which is commonly used for visualization of the performance of an supervised problem. Each row of the matrix represents the instances in a actual class while each column represents the instances in an predicted class.

In the table below, x-axis is predicted label while y is true label. And grids with deep color are the predicted label with highest possibility of a true label.

	Bag of Words	Lexical Features	Punctuation
Neural Networks	<p>Normalized confusion matrix</p>	<p>Normalized confusion matrix</p>	<p>Normalized confusion matrix</p>
SVM (RFB)	<p>Normalized confusion matrix</p>	<p>Normalized confusion matrix</p>	<p>Normalized confusion matrix</p>

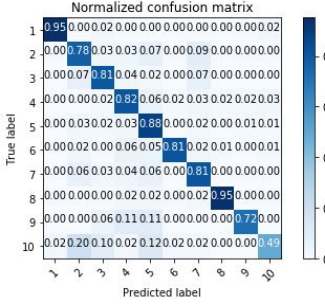
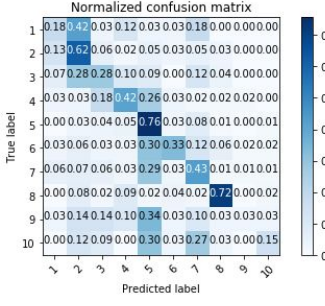
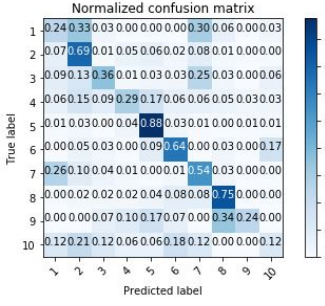
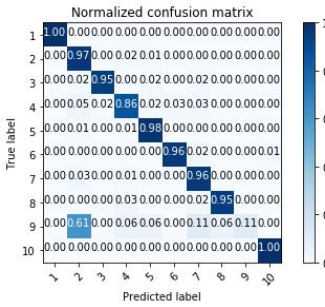
<p>KNN (k = 5,10,7)</p>			
<p>Naive Bayes</p>		<p>N/A</p>	<p>N/A</p>

Table 2: Confusion matrices of models and features

Among the four models we have applied, we found that the Neural Networks and bag of words performs best among 4 models and 3 features.

SVM and Naive Bayes have poor performance on recognizing works of the 9th author and tend to predict its authorship to the 2nd and 4th author.

Under lexical and punctuation features, models works great in detecting corpus of the 2nd (Charles Dickens), 5th (Author Conan Doyle) and 8th author (William Shakespeare) and perform badly at the 9th(Lewis Carroll) and 10th author (Fyodor Dostoyevsky).

The result is quite reasonable that most works of Author Conan Doyle are detective novel with distinctive words focusing on criminal or reasoning, while works of William Shakespeare are mainly poetry with awkward words that are not commonly used by other people. And for Fyodor Dostoyevsky, most of his work are translated from Russian to English, thus causing unavoidable error because the works will also show the linguistic style of translator.

- Recommendations

In this project, we give the recommendations based on the result of confusion matrices shown above.

It can be seen that corpus of Jane Austen (1st) has most similar lexical style with Charles Dickens (2nd). Mark Twain(3th) and Robert Louis Stevenson (7th) tend to have similar punctuation habits. The corpus of Lewis Carroll(9th) share some similarity with works of Oscar Wilde(4th) and Charles Dickens(2th). Readers like Fyodor Dostoyevsky(10th) might get recommendation to Author Conan Doyle(4th) and Robert Louis Stevenson (7th).