Machine Learning: Assignment 03 (Author Profiling)

Waheed Abbas [FA17-RCS-013]

24 November 2018

1) ABSTRACT

Author profiling is the process to identify traits such as gender, age and many other features through their writing style. In this study we identified gender of author using dataset of 426 profiles where we have 215 female and 211 male profiles. We applied stylometry and content-based method to extract features from these profiles. To select features, we applied Recursive feature elimination method to select 50 features. In final step different classification methods like Naïve Bayes, Support Vector Machine, Ada Boost Classifier and Random Forest applied to get results.

2) FEATURE EXTRACTION METHODOLOGY

In first step we applied stylometry and content-based method to extract feature from given profiles. These methods are explained in detail below.

a) STYLOMETRY BASED METHOD

Stylometry based feature selection is a statistical technique. In this study we extracted 20 different features like: number of small letters, number of capital letters, number of spaces, number of digits, number of commas, number of full stops, number of at signs, number of right brackets, number of left brackets, number of exclamation marks, number of dashes, number of question marks, number of percentage signs, number of and signs, number of hashes, number of underscores, numbers of equal signs, number of semicolons, number of colons and number of forward slashes. All mentioned features were extracted for all 426 profiles and saved into csv file (stylometry.csv). Another column also inserted to represent as author where 0 is for female and 1 is for male.

b) CONTENT BASED METHOD

We can identify author gender through content of writing like males mostly talks about news, sports and politics and females talks about cooking, fashion etc. So, get these contents from author data we used two approaches: word n-gram and char n-gram. For word n-gram we took length varying from 1 to 3 and for char n-gram we took length from 3 to 10. To extract n-gram we used TfidfVectorizer from sklearn library. We selected max 1000 most prominent features to reduce training time and overfitting. After selecting char and word n-gram features we saved these into csv files for future use.

3) FEATURE SELECTION METHODOLOGY

Most machine learning works on a simple rule – if you put garbage in, you will get garbage out. And by garbage here is noise. We don’t need every feature to train our machine learning algorithm so we select those features which are really important. Feature selection not only reduce training time but also improve accuracy and reduces overfitting.

There are many feature selection methods are given in sklearn library but for our study we used Recursive Feature Elimination method. It is a greedy optimization algorithm which aims to find best performing feature subset. It repeatedly creates models and keeps aside the best or worst performing features at each iteration. It constructs the next model with the left features until all features are exhausted. It ranks the feature based on the order of their elimination. For our study we used this method with SVM (support vector machine) to select 50 features. Why we selected 50 features? Through hit and trial, we saw that 50 features gave more accurate scores on trained model. After applying feature selection, we saved these to new csv files for future use.

4) CLASSIFIERS

There is a wide range of classification algorithms are available and each algorithm has own strength and weakness. No single algorithm works best on all learning problems. So, we used following classification algorithm on our problem:

a) GAUSSIAN NAÏVE BAYES

Gaussian Naïve Bayes is a probabilistic classifier. In which continues values associated with each feature are assumed to be distributed according to Gaussian distribution. A Gaussian distribution is also called Normal distribution. When plotted, it gives a bell-shaped curve which is symmetric about the mean of feature values.

b) LINEAR SUPPORT VECTOR MACHINE

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for classification and regression. This algorithm work by plotting each data item as a point in n-dimensional space with the value of each feature being the value of a particular coordinate. Then perform classification by finding the hyper-place that differentiate the two classes very well.

c) ADABOOST CLASSIFIER

AdaBoost is an ensemble boosting classifier. It is an iterative ensemble method. It builds strong classifier by combining by combining multiple poorly performing classifier so that you will get high accuracy strong classifier. The basic concept behind AdaBoost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate prediction of unusual observations.

d) RANDOM FOREST CLASSIFIER

Random forest is an ensemble method and fall under supervised classification algorithm category. It works by generating forest using decision trees and make it somehow random. Each decision tree gives classification for a new object and then combine the results of all trees to give final prediction.

5) EXPERIMENTAL SETUP AND RESULTS

To compare performance of different classifier on given data we used python along with different libraries to handle tasks like loading data, cleaning data, feature selection, feature extraction and model training. We applied preprocessing on word n-gram before passing to feature extraction algorithm. We used 2/3 data for training and 1/3 data for testing purpose. In first experiment we applied all four classification algorithms on stylometry features without feature selection. In second experiment we applied feature selection on both word n-gram and char n-gram features then we using all four classification algorithms to get results. Results (accuracy and root mean square error) are presented in Table 1, Table 2 and Table 3 below.

a) STYLOMETRY BASED METHOD RESULTS

Table 1: Classifier Results for stylometry without feature selection

Algorithm	**Accuracy**	**RMS Error**
Gaussian Naïve Bayes	0.631 %	0.369
Linear Support Vector Machine	0.567 %	0.433
Ada Boost Classifier	0.638 %	0.362
Random Forest	0.624 %	0.376

b) CONTENT BASED METHOD RESULTS

Table 2: Classifier Results for word n-gram with feature selection

n	**Algorithm**	**Accuracy**	**RMS Error**
unigram	Gaussian Naïve Bayes	0.766 %	0.234
	Linear Support Vector Machine	0.809 %	0.191
	Ada Boost Classifier	0.759 %	0.241
	Random Forest	0.730 %	0.270
bigram	Gaussian Naïve Bayes	0.716 %	0.284
	Linear Support Vector Machine	0.816 %	0.184
	Ada Boost Classifier	0.745 %	0.255
	Random Forest	0.709 %	0.291
trigram	Gaussian Naïve Bayes	0.624 %	0.376
	Linear Support Vector Machine	0.631 %	0.369
	Ada Boost Classifier	0.589 %	0.411
	Random Forest	0.631 %	0.369

Table 3: Classifier Results for char n-gram with feature selection

n	**Algorithm**	**Accuracy**	**RMS Error**
3	Gaussian Naïve Bayes	0.787 %	0.213
	Linear Support Vector Machine	0.794 %	0.206
	Ada Boost Classifier	0.723 %	0.277
	Random Forest	0.660 %	0.340
4	Gaussian Naïve Bayes	0.780 %	0.220
	Linear Support Vector Machine	0.816 %	0.184
	Ada Boost Classifier	0.752 %	0.248
	Random Forest	0.752 %	0.248
5	Gaussian Naïve Bayes	0.766 %	0.234
	Linear Support Vector Machine	0.809 %	0.191

	Ada Boost Classifier	0.816 %	0.184
	Random Forest	0.745 %	0.255
6	Gaussian Naïve Bayes	0.752 %	0.248
	Linear Support Vector Machine	0.766 %	0.234
	Ada Boost Classifier	0.745 %	0.255
	Random Forest	0.738 %	0.262
7	Gaussian Naïve Bayes	0.801 %	0.199
	Linear Support Vector Machine	0.794 %	0.206
	Ada Boost Classifier	0.773 %	0.227
	Random Forest	0.674 %	0.326
8	Gaussian Naïve Bayes	0.730 %	0.270
	Linear Support Vector Machine	0.709 %	0.291
	Ada Boost Classifier	0.716 %	0.284
	Random Forest	0.681 %	0.319
9	Gaussian Naïve Bayes	0.716 %	0.284
	Linear Support Vector Machine	0.723 %	0.277
	Ada Boost Classifier	0.660 %	0.340
	Random Forest	0.674 %	0.326
10	Gaussian Naïve Bayes	0.716 %	0.284
	Linear Support Vector Machine	0.745 %	0.255
	Ada Boost Classifier	0.688 %	0.312
	Random Forest	0.645 %	0.355

6) OBSERVATION

In table 1 (stylometry method), AdaBoost classifier out performed the all other classifiers and SVM performed worse than all. In table 2 (content-based word n-gram) SVM out performed all other classifier with accuracy 0.809 %, 0.816 %, 0.631 % for unigram, bigram and trigram respectively. For content-based char n-gram for 3,4,6,9 and 10 SVM outperformed all other classifier. In 5 char n-gram AdaBoost out performed other classifiers and for 7,8 char n-gram Gaussian Naïve Bayes performed very well as compare to all other classifiers. We observe that if we applied more preprocessing on given data and especially on char n-gram along with extensive and careful feature selection we can achieve more accuracy. We can also finetune all applied algorithm to improve accuracy because here we are taking default parameters of algorithms.

7) CLUSTERING

We applied two algorithms (KMeans and Agglomerative Clustering) on stylometry and content-based datasets.

Table 4: Clustering Results for Stylometry

Algorithm	**Accuracy**	**RMS Error**
KMeans	0.652 %	0.348
Agglomerative Clustering	0.383 %	0.617

Table 5: Clustering Results for word n-gram

n	**Algorithm**	**Accuracy**	**RMS Error**
unigram	KMeans	0.433 %	0.567
	Agglomerative Clustering	0.589 %	0.411
bigram	KMeans	0.468 %	0.532
	Agglomerative Clustering	0.539 %	0.461
trigram	KMeans	0.468 %	0.532
	Agglomerative Clustering	0.532 %	0.468

Table 6: Clustering Results for char n-gram

n	**Algorithm**	**Accuracy**	**RMS Error**
3	KMeans	0.546 %	0.454
	Agglomerative Clustering	0.553 %	0.447
4	KMeans	0.477 %	0.553
	Agglomerative Clustering	0.539 %	0.461
5	KMeans	0.454 %	0.546
	Agglomerative Clustering	0.574 %	0.426
6	KMeans	0.582 %	0.418
	Agglomerative Clustering	0.468 %	0.532
7	KMeans	0.496 %	0.504
	Agglomerative Clustering	0.518 %	0.482
8	KMeans	0.489 %	0.511
	Agglomerative Clustering	0.553 %	0.447
9	KMeans	0.482 %	0.518
	Agglomerative Clustering	0.546 %	0.454
10	KMeans	0.504 %	0.496
	Agglomerative Clustering	0.539 %	0.461

8) OBSERVATION

On stylometry dataset KMeans worked pretty well as compare to Agglomerative Clustering algorithm. KMeans gave accuracy of 0.652 and Agglomerative Clustering gave 0.383. On word n-gram dataset Agglomerative Clustering performed better than KMeans and in char n-gram Agglomerative Clustering also performed well as compared to KMeans. If we compare these results with other supervise learning algorithms, we can say that both clustering algorithms performed not so good.