CSCI 49362 Language Technology/CSCI 7600 Computational Linguistics

Prof. Levitan

**Lab #2: Authorship Identification**

The goal of this lab is to gain experience using NLTK to preprocess text and extract features and use [scikit-learn](https://scikit-learn.org/stable/) to classify text.

**Motivation**

Psychologists and linguists have argued that there are gender differences in language. For example, Lakoff (1972) states that there is a “women’s language” where they are more likely to use hedge words, tag questions, etc.

In this lab we aim to answer the following question: *Is this accurate? Given a text, can we automatically detect the gender of the author, using a variety of linguistic cues?*

Lakoff, R. (1973). Language and woman's place. Language in society, 2(01), 45-79.

**Data**

You are provided with a subset of the BNC corpus:

570 essays, 285 written by female-identified authors, 285 by male-identified authors.

The essays used are listed in CORPUS\_TXT/BAWE\_balanced\_subset.csv. The “fname” column is the essay filename (CORPUS\_TXT/[fname].csv) and the “gender” column is the author gender (0=male, 1=female).

Disclosure: We selected a subset of essays from the BNC Corpus that are balanced by gender.

**Task**

You are provided with a script “gender\_id.py”, which loads the BNC corpus, calls a feature extractor module, and then trains a Naïve Bayes classifier using those features. Your job is to fill in the feature extraction steps in the “feature\_extractor.py” file, using NLTK.

You should extract the following features (based on the literature of authorship identification):  
1. Function words: counts for each function word (from the NLTK stopwords list)

**\*this function is already implemented for you!**

2. Syntactic: counts for the following 10 common POS tags -- ['NN', 'NNP', 'DT', 'IN', 'JJ', 'NNS','CC','PRP','VB','VBG']

3. Lexical: counts for 30 most common unigrams in entire corpus (remove stopwords and punctuation for unigram count)

4. Punctuation: Counts for the following 10 punctuation marks -- '.,:;-\'\"(!?'

5. Complexity:

* average number of characters per word
* #unique words/#total words
* average number of words per sentence
* Count of “long” words - words with >= 6 letters

You are encouraged to add your own additional features and to play around with preprocessing and normalization that might help performance. For example, in the sample function words function, we normalized the counts of each essay by dividing by number of tokens in that essay. This might improve performance.

**Running the script**

To run gender\_id.py:

>python gender\_id.py [data directory] [configuration file] [features output file]

For example:

>python gender\_id.py CORPUS\_TXT conf.txt features.csv

The script will print the accuracy of the trained classifier, using the features listed in the configuration file. To adjust which features you are extracting, simply edit the configuration file. For example, to only extract and classify the function\_words feature, the conf.txt file should only contain the line “function\_words”.

You are provided with a sample output file containing the function words features only: function\_words.csv

**Submission**

Submit:

1. feature\_extractor.py
2. all\_features.csv (containing at least all the features specified above, make sure there is a header with the name of each feature)
3. results.pdf (a write up of your results and answers to the 2 questions below)

**Group members: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**1. Results**

Report the accuracy obtained from gender\_id.py using the following individual feature sets, as well as using all features combined.

|  |  |
| --- | --- |
| Feature Set | Accuracy |
| Function words |  |
| Syntax |  |
| Lexical |  |
| Punctuation |  |
| Complexity |  |
| All features |  |

**2. Based on the features you extracted and the results, what observation(s) can you make about gender and writing styles?**

**3. What are some ideas you could try to further improve performance?**