Statistical Approaches to NLP
Final PA – Latin Authorship Identification

José Andrés Molina

December 18, 2016

1 Data Prep

The corpus for this machine learning experiment was taken from the Perseus Project's XML-encoded Latin Literature corpus. These files were converted to JSON by the CLTK (Classical Language Toolkit) group and are openly available for download on their GitHub page.

The formatting for these files what not the most consistent and extracting the raw text from them to convert them into a format that would be much more readable for the classifier took several days of reading through examples and a lot of trial and error. The final result was a data prep script called `latin_authors_corpus_prep.py` that takes all of the JSON files of Latin text within `latin_text_perseus` and recursively finds the raw text, looking for specific tags (such as `"l"` for line of poetry, `"p"` for paragraph, or `"#text"` for other text, among others) and then concatenates all the raw text and stores into one giant JSON file of mappings of author names to a list of the entire text of all their works at `latin_text_perseus/latin_authors_corpus.json`. This JSON file is then read by the `Corpus` object's `load` function to extract all the `Documents`.

2 Classifiers

The Latin texts had to be pre-split in the `Corpus` as `train_data`, `dev_data`, and `test_data` to ensure that all authors were covered. Only authors with 7 or more works were used to (somewhat) avoid sparse data; this resulted in the ten possible author labels being Cicero, Horace, Livy, Ovid, Plautus, Prudentius, Seneca, Tacitus, Terrence, and Vergil.

Tests were written to run the corpus on the prior implementations of Naïve Bayes and MaxEnt, checking that classification was over 50% accuracy. The implementation of MaxEnt had to be adjusted to ensure it would converge on this data set, the learning rate was dropped

0.0001, the 1000 most common features were used to train, and the convergence check was change from accuracy to log-likelihood. This was also done with a lot of trial and error that resulted many times with arithmetic overflow in the `exp` function until the correct numbers were found. Because training takes several minutes for this model, the results are saved to `models/latin_maxent` and then loaded up in the test.

Feature engineering on this mostly either resulted in memory errors (in the case of trigrams) or did not improve results (in the case of bigrams and length features). Bag-of-words with punctuation removed ended up having the clearest results. Ideally a removal of Latin stopwords could improve results as well.

3 RESULTS

3.1 DISCUSSION

The results yielded a 55.55% accuracy with Naïve Bayes and a 69.44% accuracy with MaxEnt. Looking at the Recall and Precision values for each label, authors with the fewest works such as Horace, Prudentius, Tacitus, and Terrence tended to do the worst (all of them had 0% recall and precision for Naïve Bayes and all but Horace had 0% recall and precision for MaxEnt). Vergil surprisingly also did poorly with 0% recall and precision, probably because of the variety of works (for example, one is an epic poem on the journey of Aeneas from Troy to Italy, another is an account of agrarian practices). These both seems like cases were the data for their kind of writing is just too sparse to draw any sort of significant conclusion for the author label. In the future, it might be interesting to try spilling up works by chapters and using parts of the same work for training, development, and testing instead of treating the whole work as a single document to replicate having more data. Unsurprisingly, authors that had more data or more peculiar writing styles (such as Livy) tended to perform the best for classification.

The results proved to do far better than chance for ten labels ($\frac{1}{10}$) for both classifiers. MaxEnt performed much better than Naïve Bayes as expected for a small unbalanced corpus such as this one. Unfortunately, figuring out the right way to parse and save the corpus data for the experiments and the multitude of debugging when running through the classifiers took up way more time than anticipated. Given more time, it would be interesting to see results when testing them on other third party classifiers such as those in `scikit-learn`.

3.2 NAÏVE BAYES

	Recall	Precision	F_1
Cicero	100.0%	36.36%	53.33%
Horace	0.0%	0.0%	0.0%
Livy	100.0%	100.0%	100.0%
Ovid	100.0%	100.0%	100.0%
Plautus	80.0%	100.0%	88.88%
Prudentius	0.0%	0.0%	0.0%
Seneca	33.33%	50.0%	40.0%
Tacitus	0.0%	0.0%	0.0%
Terence	0.0%	0.0%	0.0%
Vergil	0.0%	0.0%	0.0%

Average $F_1 = 38.22\%$

Accuracy = 55.55%

3.3 MAXENT

	Recall	Precision	F_1
Cicero	100.0%	80.0%	88.88%
Horace	50.0%	100.0%	66.66%
Livy	100.0%	80.0%	88.88%
Ovid	50.0%	100.0%	66.66%
Plautus	100.0%	71.42%	83.33%
Prudentius	0.0%	0.0%	0.0%
Seneca	100.0%	50.0%	66.66%
Tacitus	0.0%	0.0%	0.0%
Terence	0.0%	0.0%	0.0%
Vergil	0.0%	0.0%	0.0%

Average $F_1 = 46.11\%$

Accuracy = 69.44%