Hyomin Seo

Natural Language Processing 2021 Spring

Project 1 Text Categorization

Professor Sable

The text categorization assignment is written in Python, applying the Natural Language ToolKit (NLTK) package to use tokenization and stemming to maximize the text categorization accuracy. In order to execute the code successfully, one must install ‘numpy’, ‘nltk’ , and ‘math’ in advance on his pyton , and has to import several tools: ‘word\_tokenize’, ‘PorterStemmer’, ‘log’, and ‘string’. When one has to install nltk.word\_tokenize, he also (might) has to install ‘punkt’ as well. One can easily install ‘nltk’ on python by running the line ‘pip install --user -U nltk’, or follow the instructions on <http://www.nltk.org/install.html>. The user of the program must provide the name of the file that contains a labeled list of training documents and a list of testing documents to be categorized.

The text categorization system is largely based on one of the most frequently used basic machine learning methods - Naive Bayes approach. The system first applies ‘nltk.word\_tokenize’ to tokenize (to split the sentence into smaller units), both training and test files, after both were appropriately treated (split) into lines and are stored separately. The additive smoothing was implemented by adding a unigram smoothing, which is constructed as a function of the test sets with a smoothing coefficient. The smoothing coefficient was tuned until it showed a competitive accuracy when test-ran.

The conditional categorial probability is calculated with the set of equations, then the final decision (prediction) is made based on the result of these calculated conditional probabilities. This course can be found under the # Conditional Probability in the code, more specifically, calculated the total conditional probability of article in the specific category. (conditional probability in the system isn’t calculated using nltk.probability.ConditionalFreqDist())

Once the documents were processed and tokenized, the first loop goes through the set category of the train/test file, counting the number of viewed categories presented in the file.

The second loop is there to strengthen the accuracy, during which the stemming happens - Porterstemmer dedicated strongly to yielding a competitive accuracy. When implemented, it increased the accuracy significantly. Trying Porterstemmer and LancasterStemmer, (assuming that I implicated Lancaster stemmer right), the majority of the result from Lancaster stemmer turned out to be obfuscated, to the point that it can not be distinct. Speaking to some classmates and upperclassmen, they suggested that I drop Lancaster stemmer and adopt porter stemmer, even though Lancaster stemmer does reduce the working size significantly.

References and Help:

<https://wikidocs.net/21707> (lemmatization and stemmeing analyzation)

<https://pythonprogramming.net/stemming-nltk-tutorial/> (Porterstemmer tutorial)

<https://excelsior-cjh.tistory.com/67> (Porterstemmer and lancasterstemmer analyzation)

<https://medium.com/data-from-the-trenches/text-classification-the-first-step-toward-nlp-mastery-f5f95d525d73> (vectorization was interesting but did not use it in this project)

[https://www.datacamp.com/community/tutorials/stemming-lemmtization-python](https://www.datacamp.com/community/tutorials/stemming-lemmatization-python)

(porterstemmer , text normalization and general text stemming)

<https://www.nltk.org/book/ch06.html> ( ntlk)

<https://www.datacamp.com/community/tutorials/text-analytics-beginners-nltk> (nltk tokenization)

<https://towardsdatascience.com/machine-learning-nlp-text-classification-using-scikit-learn-python-and-nltk-c52b92a7c73a> (Naive Bayes Unigram)

<https://web.stanford.edu/~jurafsky/slp3/slides/7_NB.pdf> (Text classification and Naive Bayes Concept)

<https://github.com/banjodayo39/Naive-bayes-nlp/blob/master/implement-naives-bayes-from-scratch.ipynb> (Naive Bayes from scratch/ did not implemented for this system but good source to understand the logic)

<http://www.katrinerk.com/courses/intro-to-computational-linguistics-ug/schedule-introduction-to-computational-linguistics-undergraduate/demo-conditional-probability>

(conditoinal probablity nltk from scrtach )

* How does your system tokenize training and test files?
* What weighting scheme, if any, is used for tokens?
* If you used Naïve Bayes, what method of smoothing is used?
* Which optional parameters or features did you experiment with (e.g., possibilities might include case sensitivity, POS tagging, stop lists, etc.)? Which parameters or features made a significant difference, and how are they set in your final system?
* How did you evaluate your system's performance for the second and third data sets?
* You may include any additional information that you wish.
* Built a text categorization system in Python using a Naïve Bayes approach on unigrams