Annelise Wittenberg & Cody Stammer

CSE 5243 – Assignment 1

September 18, 2014

# Approach

Initially, we decided to take the Java approach since both of us are very strong in Java. Neither of us had done more than a few lines of string manipulations and loops in the past, so switching to Python for completion of this project was a difficult decision.

Upon switching to Python, we did most of the development on Ubuntu and used the NLTK library for text processing, as well as Beautiful Soup for parsing and document extraction. Our main function iterates through each of the 22 documents and processes each article within each document individually. We first pre-processed the data by eliminating punctuation, words 2 letters or less, and numbers. Additionally, we converted all text to lowercase so no duplicate words could occur. After preprocessing, we created a feature vector consisting of bigrams. The vector only has 10 bigrams because we chose to go with the 10 most commonly occurring bigrams. This vector still includes stopwords, however. The second feature vector consists of the word frequency with the stopwords removed. Each of the two feature vectors is associated with the topics list and places list for each article. Once each feature vector was created, we output all of the processed data to an output file with the format of the document number – one number per article – followed by the topics list, places list, bigram vector, and word frequency vector.

# Rationale

Our rationale for choosing the python approach was that it was *significantly* easier to implement, even with having to spend some time learning more of the language. We chose these two types of feature vectors because they present two different types of information: commonly occurring phrases for a given topic and the frequency of words used for a given topic.

# Difficulties

We experienced several difficulties while creating this project. First and foremost, the instructions weren’t clear, so several days passed by before we were actually able to begin developing. We had to wait for instruction from multiple sources explaining what was actually being asked of us. Once we began coding, it was quickly discovered that there was nothing available to Java to easily help us parse the .sgm files. The XML parser was very strict, making it difficult to manipulate the file. Even when we tried to relax the parser, we had several issues with extracting portions of the document. At this point, we switched to Python and our next difficulty was a matter of syntax. We knew step by step what we wanted to do, it just became a matter of how to implement it in Python. Once we determined how to break up the functionality into different functions and how to use the NLTK/Beautiful Soup libraries, we began to code freely and much more easily.

# Assumptions

We made a few assumptions during the process of this assignment. The main assumption we made was that each article’s tags were exactly the same as all of the rest. We made this assumption in order to make parsing and processing each document much easier. We also assumed that stopwords and words of length 2 or less were impertinent to the articles themselves.