1. Parsing the file

Since every file contains more than one piece of REUTERS news, every file isn't a standard XML file. When we read the file, we add the <REUTERSList> tag at the beginning of the file and </REUTERSList> tag at the end of file to make sure it is a standard XML file.

We use **Beautiful Soup(http://www.crummy.com/software/BeautifulSoup/)** to parse every file. For every router news, we get the NEWID, TOPICS, TITLE and BODY contents.

2. Determine the vocabulary of terms

**NLTK(**[**http://nltk.org/**](http://nltk.org/)**)** is used to determine the vocabulary of terms in the TITLE and BODY contents.

(1) Tokenization

We divide the text into sentences. For every sentence, we chopped it up into tokens at the same time throwing away the stop words, punctuations and numbers.

(2) Normalization

We normalized the tokens into the lower cases except for the entities.

(2) Stemming and Lemmatization

We use stemming and lemmatization to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form.

3. Constructing the feature vectors

We construct two feature vectors: the term frequency vector and tf-idf vector.

We use the class labels as provided in the TOPICS tags of each article.

**The term frequency vector:**

For every token in the document, we calculate the number of times that term t occurs in the document.

**The tf-idf vector:**

tf–idf is the product of two statistics, term frequency and inverse document frequency. Considering that sometime, the term frequency is so large that it has a dominant impact on the final value, we use the augmented frequency, which is shown below.



f(t, d) : the frequency of term t in document d.

The inverse document frequency is obtained by dividing the total number of [documents](http://en.wikipedia.org/wiki/Documents) by the number of documents containing the term, and then taking the [logarithm](http://en.wikipedia.org/wiki/Logarithm) of that [quotient](http://en.wikipedia.org/wiki/Quotient).



*D*: the total number of documents in the corpus

4. Implementation Details

(1) Since the text in the TITLE is much more important than the text in the BODY, we assign a higher weight for the tokens which occure in the TITLE text.

(2) For the news without topics, we assign 'None' to the class label.

For the news with multiple topics, we assign multiple topics to the class label, each topic separating by semicolon.

In order to reduce the length of vector and avoid the unnecessary computation, we only include the terms with high frequency in the vector. The frequency threshold is 1.