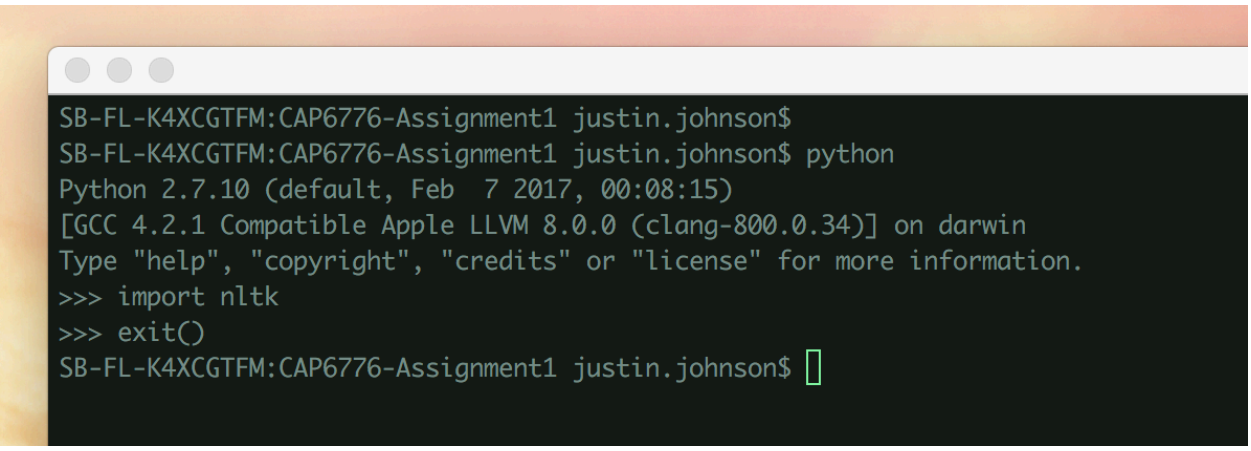
Information Retrieval
CAP6776, FALL 2017

Justin Johnson
Z23136514

Assignment 1
Using NLTK To Conduct Text Processing and Calculate Cosine Similarity

Part I) Install Python 2 and NLTK



As seen in above screenshot, python 2.7 was successfully installed and NLTK was imported without error.

Part II) Tokenize the documents into words, remove stop words, and conduct stemming

```python

# define stemming mechanism and stop words
ps = PorterStemmer()
nltk_stop_words = set(stopwords.words('english'))


# build corpus: list of documents
# stop words are ignored, words are stemmed using PorterStemmer
corpus = []
for f in files:
    strm = open(DATA_SET_DIR + '/' + f, 'r')
    # using nltk word tokenizer to split file into word list
    words = word_tokenize(strm.read())
    # using filter to remove stop words from word list
    words = filter(lambda w: w not in nltk_stop_words, words)
    # using map to stem words in word list
    words = map(lambda w: ps.stem(str(w)), words)
    # joining words into string and adding to corpus list
    corpus.append(' '.join(words))

```

Each file is read from data set directory one at a time. As each file is read, it is split into a list of words using NLTK's word_tokenize. The list is filtered using NLTK's stop words, removing stop words from the list. Each word is then transformed to it's stem using NLTK's Porter Stemmer. Once stop words are removed and stemming is complete, the words are added to the corpus.

Part III) Calculate tf-idf for each word in each document and generate document-word matrix.

The tf-idf matrix was calculated using sklearn's TfidfVectorizer. The corpus (collection of documents) is passed to TfidfVectorizer instance's fit_transform method, generating the tf-idf matrix.

Unlike the course textbook which calculates idf values as log(N/df), the TfidfVectorizer calculates idf values as log(N/df) + 1. According to sklearn's documentation, the + 1 is applied to prevent terms that appear in all documents from being completely ignored.
http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html

Below is a screenshot that captures a small portion of the document-word matrix. Columns 1 - 5 correspond to Document 1 - Document 5. Each row corresponds to a unique term in the corpus. Each element is the tf-idf value for the (term, document) pair.

```
tycoon          [ 0.          0.          0.          0.          0.03851868]
uncertain       [ 0.          0.          0.          0.04522772  0.        ]
uncertainti     [ 0.03005168  0.03731543  0.03875656  0.          0.        ]
underway        [ 0.          0.          0.          0.          0.03851868]
union           [ 0.          0.          0.          0.04522772  0.        ]
unit            [ 0.          0.          0.          0.13568316  0.        ]
unlik           [ 0.          0.          0.          0.03648944  0.03107663]
unpaid          [ 0.04487256  0.          0.          0.          0.        ]
unspecif        [ 0.          0.          0.          0.          0.03851868]
urg             [ 0.04487256  0.          0.          0.          0.        ]
use             [ 0.          0.          0.          0.          0.11555603]
volum           [ 0.          0.          0.          0.04522772  0.        ]
want            [ 0.08974513  0.          0.          0.          0.        ]
warrant         [ 0.03005168  0.03731543  0.03875656  0.          0.        ]
we              [ 0.0362029   0.          0.          0.          0.03107663]
wednesday       [ 0.          0.          0.          0.          0.03851868]
week            [ 0.03005168  0.11194628  0.11626967  0.          0.        ]
welcom          [ 0.          0.          0.          0.          0.03851868]
well            [ 0.0362029   0.          0.          0.07297888  0.        ]
whether         [ 0.          0.          0.          0.          0.03851868]
wide            [ 0.02528042  0.03139091  0.03260324  0.          0.0434015 ]
will            [ 0.          0.          0.          0.04522772  0.        ]
wipe            [ 0.03005168  0.03731543  0.03875656  0.          0.        ]
within          [ 0.03005168  0.03731543  0.03875656  0.          0.        ]
without         [ 0.          0.          0.          0.04522772  0.        ]
work            [ 0.02528042  0.03139091  0.03260324  0.02548051  0.        ]
worker          [ 0.          0.          0.          0.04522772  0.        ]
worldwid        [ 0.          0.          0.          0.04522772  0.        ]
worth           [ 0.          0.          0.          0.04522772  0.        ]
would           [ 0.14967412  0.10620094  0.11030244  0.0431025   0.03670871]
wrangl          [ 0.          0.04495345  0.04668957  0.          0.        ]
wrestl          [ 0.03005168  0.03731543  0.03875656  0.          0.        ]
year            [ 0.1282921   0.02655023  0.02757561  0.02155125  0.03670871]
yet             [ 0.          0.          0.          0.          0.03851868]
```

Part IV) Calculate the pairwise cosine similarity for the documents

Sklearn's cosine_similarity calculates the cosine similarity for all document pairs in the corpus. The cosine_similarity function accepts 2 vectors, or 1 matrix as it's input. In our case, we were able to calculate a cosine similarity matrix by passing sklearn's cosine_similarity function the tf-idf matrix result from Part III. As expected, the matrix diagonal consists of 1s, because every document is an exact match to itself.

Cosine Similarity Output:

```
Printing cosine similarity matrix:
         Doc1         Doc2        Doc3        Doc4        Doc5
Doc1    [ 1.         0.75706994  0.77559183  0.18308252  0.12413457]
Doc2    [ 0.75706994 1.          0.98273319  0.140519    0.10709193]
Doc3    [ 0.77559183 0.98273319  1.          0.14227134  0.11323988]
Doc4    [ 0.18308252 0.140519    0.14227134  1.          0.08401891]
Doc5    [ 0.12413457 0.10709193  0.11323988  0.08401891  1.        ]

Complete.
SB-FL-K4XCGTFM:CAP6776-Assignment1 justin.johnson$
```