Project Overview:

For this project, I conducted text similarity analysis on both books and tweets from twitter of a specific hashtag using cosine similarity to indicate to what extent when one text has a high count for a particular word the other text also a high count for a particular word.

Implementation:

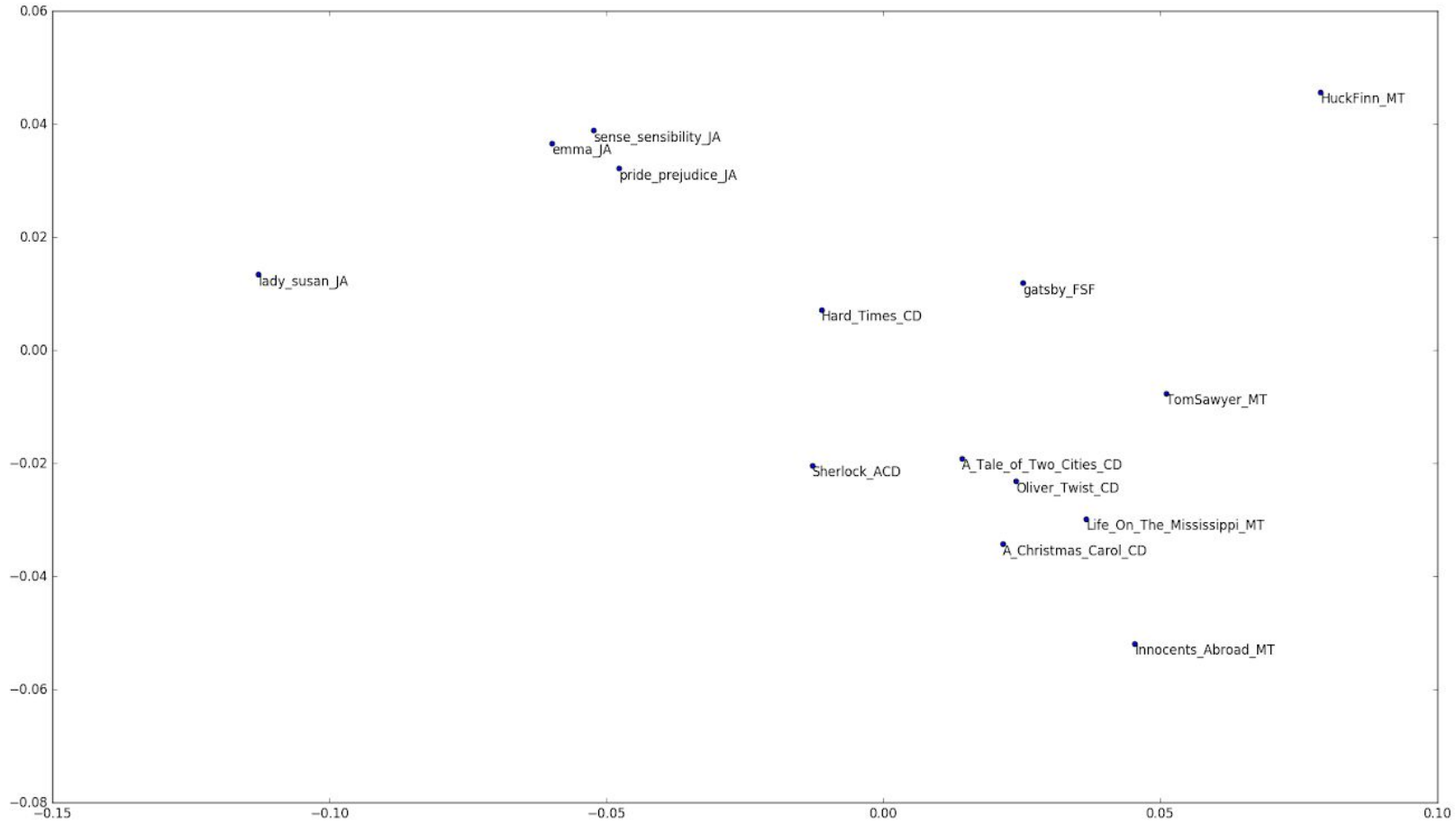
The overall process of text similarity analysis for books and tweets is the same. For both topics, the content needs to be pre-processed and returned as a list with each entry is a word appeared. In this word list, there are still duplicates for some words. The only difference between applying text similarity on books and tweets is the preprocessing stage on the text. For books, I need to strip the header information which has a pattern in terms of where it appears therefore it can be described by a function skip_header_gutenberg in the script. Apart from the header information, each word in the word list needs to be lower-cased and any characters belonged to string.punctuation and string.whitespace has to be stripped. For tweets, besides converting all words to lowercase and deleting whitespace and punctuation, other redundant information that needs to be deleted includes username, website link, and the hashtag of interest. However, unlike the header information, the location such information appears in the tweet is not as consistent as that in a book and therefore, I decide to use more conventional ways as stripping whitespace and punctuation to get rid of information not relevant to the tweet text itself.

After the preprocessing stage, the word list for either a book or tweets of a specific hashtag is used to convert to a histogram which is a dictionary object whose key is the unique word and value is its frequency. This histogram, which is also a multidimensional vector can be used to calculate the cosine similarity between two books or tweets of two hashtags. To calculate the cosine similarity between two histograms, I need find the set of words which is also the key for the dictionary that appears in book histograms. This set of words is then used to compute the cosine similarity based on the equation of finding cosine similarity between two vectors. When dealing with more than two books or two hashtags and calculating the pairwise cosine similarity for the list, I used two functions: *find_cosine_similarity_multiple_books_nonduplicates* and *find_cosine_similarity_multiple_books* for different purpose. Since in order to plot the metric Multidimensional Scaling (MDS) to visualize the texts in a two dimensional space, the list containing the cosine similarity needs to be a symmetric matrix which means the list would contain duplicate pairwise cosine similarity. For example, in the list, it would contain both the cosine similarity of book a and book b and that of book b and book a. However, such list is not useful for visualization. Therefore, *find_cosine_similarity_multiple_books_nonduplicates* creates a dictionary whose key is a unique pair of subject of interest, and the value is their cosine similarity. On the contrary,

find_cosine_similarity_multiple_books creates a list with a shape that can be used to plot MDS. This analysis is also implemented similarly for twitter.

Results:

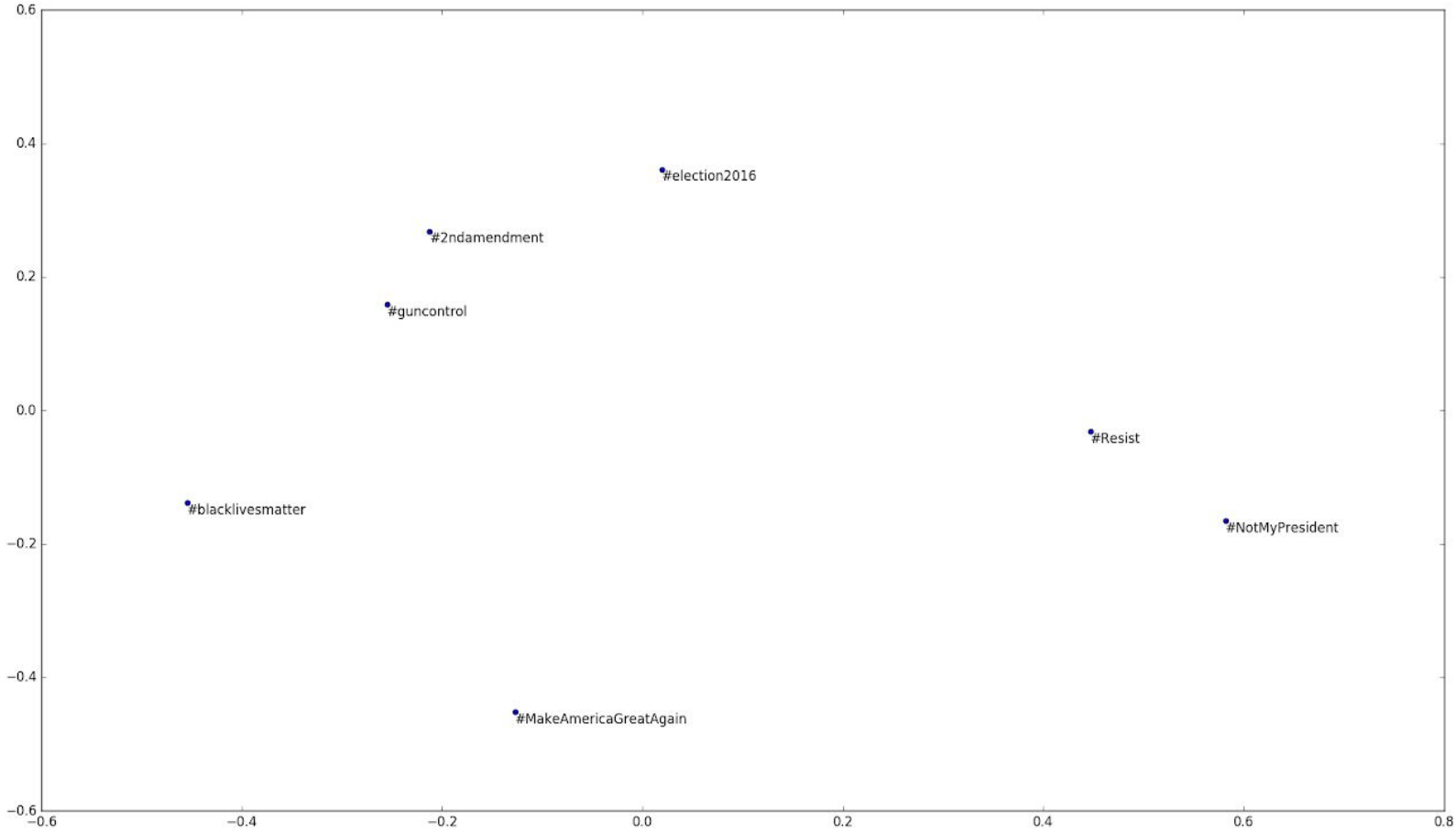
Books Analysis:



For finding text similarity among different books, I am specifically in three authors: Jane Austen, Charles Dickens and Mark Twain. Besides interested in finding the style similarity or difference among these three authors, I also choose *Sherlock* and *The Great Gatsby* to find out which of the three genres each of the text belongs closer to. From the result, most of Jane Austen's text is of similar style except *Lady Susan* and is very different from Mark Twain's style. Mark Twain does not have a consistent text style but is closer to that of Charles Dickens than to Jane Austen. Charles Dickens has a more consistent text style with the exception of *Hard Times*. Lastly, *Sherlock* is closer to genre of Charles Dickens than to the two authors and *Gatsby* remotely shares similar style with Mark Twain and Charles Dickens.

Twitter Tweets Similarity Analysis for different hashtags:

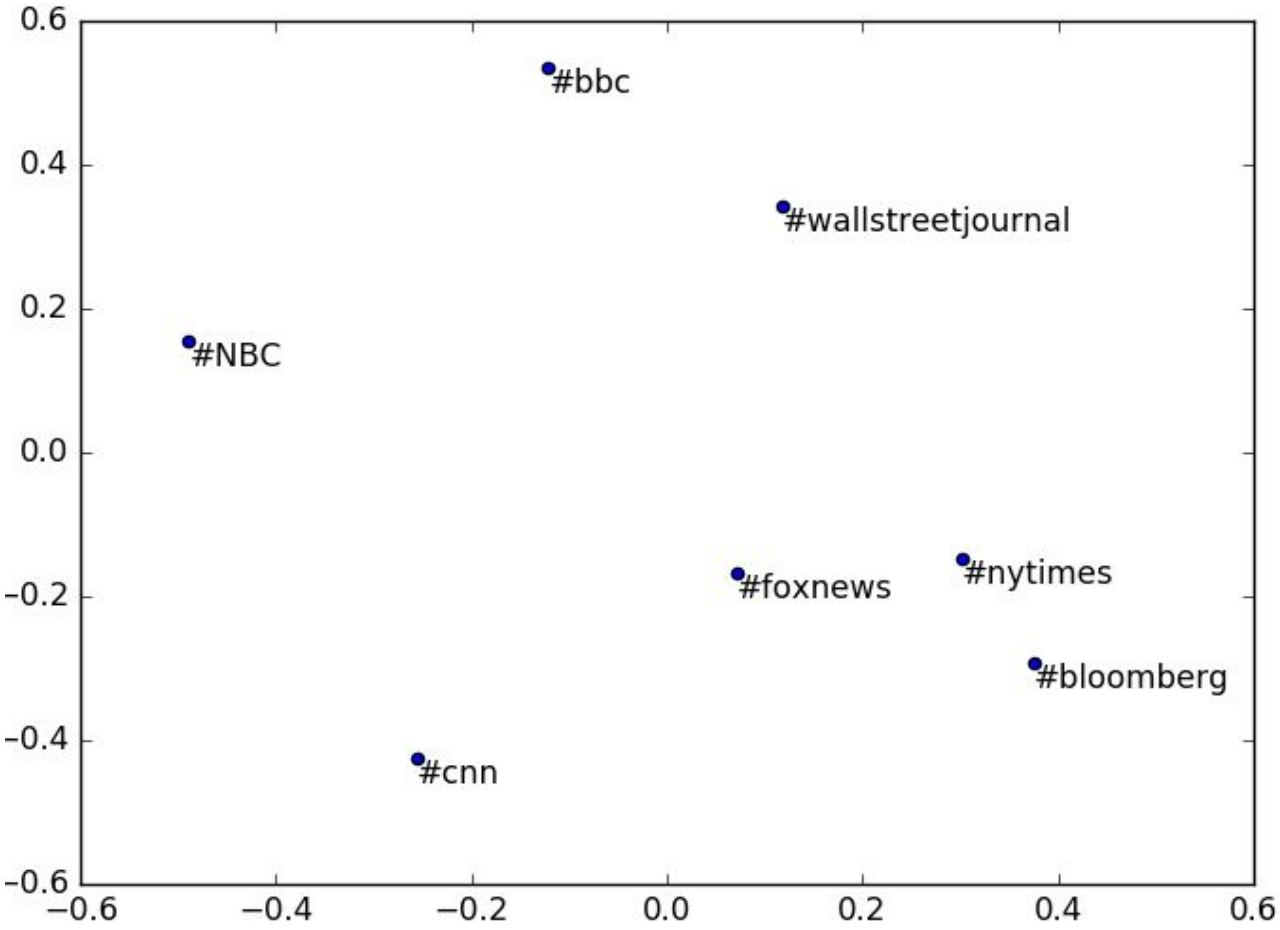
Political Topics:



For twitter, I am especially interested in finding the relationship between tweets of different hashtags of current controversial or prevalent political topics or issues. Surprisingly, even though #election2016, #NotMyPresident and #MakeAmericaGreatAgain are all related to the same event, tweets that include these hashtags have very low pairwise cosine similarity. Tweets with #Election2016 is tend to use similar words as tweets with #2ndamendment and #guncontrol.

Lastly, the graph below shows text similarity analysis of tweets from different news sources. From the result, nytimes, cnn and bloomberg tends to report similar content and use more similar words than the rest. *BBC news* and *The Wall Street Journal* have a greater pairwise cosine similarity than they are to the rest. NBC news and CNN have a relatively low pairwise similarity with the rest.

News Topic:



Reflection:

Overall my project went well since I completed the word frequency analysis toolbox beforehand and I was able to use it to conduct a text analysis of my choice and consolidate my skill in dealing with dictionary and string object. The part for calculating cosine similarity was not difficult once I understood the math equation. However, extracting data from twitter and making sure it is the right shape and size for plotting the MDS graph requires a lot of debugging. This can sometimes be frustrating because I feel like I was not actively learning new things but instead spending a lot of time fixing errors. I also wrote the docstrings and comment after finishing everything which can be inefficient since it takes time to have a system level understanding of the overall functions. For going beyond, I will apply the sentiment analysis to my topics of interest since especially for twitter, the relationship between the hashtags is not really apparent from calculating the cosine similarity.