The corpus I selected is a collection of Senate and House of Representatives minutes between the years 1861 to 2018. I interned for Carl Albert Research Center this semester and worked on political archives so this corpus grabbed my attention. It doesn't exactly cover all the years, just the presiding Presidents - Lincoln, Madison, McKinley, Monroe, Nixon, Obama, Polk, Reagan, Roosevelt, Taft, Taylor, Truman, Tyler, Washington, Wilson, and Trump. Whoosh module contains Indexing and searching libraries. I used Whoosh instead of Lucene or the other options because it is built purely in python. I also used NLTK, SciKit Learn, re library which is the regular expression operations, os for operating system interaction, and the glob module for file handling. If “indexdir” file doesn’t already exist, we create one using os.path library which helps it navigate into the operating system. Using whoosh fields, we define the schema for the index. If you notice, I have made the Title sortable so that I will be able to sort through the results in the main code by the title. The default is ascending order. Now we are starting to write into the indexdir file- we add the path to our corpus - we use glob for that. We add all the documents from the corpus to get indexed and then commit the indexdir writer. Open Indexdir file >>Accept user query and parse it into query parser >>Search through indexdir for the respective query >>As I made Title sortable in the index code, the results are sorted by Title here so like A..B..C. >>We save the relevant text snippets in results, and then we loop through the results to print the relevant >> then I saved the relevant snippets into a file called Results.

Sentiment analysis makes most sense for my corpus as it is very political, and we can determine polarity back in those days. It can be helpful for political science researchers to study the trends in polarity in the Senate and House of Representatives over time. The sentiment analysis variable is first initialized and then we use that variable on the text that needs to be analyzed and print it. For clustering, piece of raw text is passed as argument into the function >>Tokens is initialized for text passed into the function using the NLTK module >>We use the regular expressions operations library to filter the tokens that don't contain letters >>This function then returns the filtered tokens >>The highlighted part is where I clean my text of the new lines. The corpus is then transformed into vector space model (using Sklearn package) >>we obtain the vector space matrix >>initialize k for k-means >>initiate KMeans model >>use KMeans model to fit the vector space representation of the corpus >>obtain the cluster assignment for the documents as a list >>print out the cluster frequencies >>obtain centroids of the clusters with features sorted from largest value to smallest value >>get the terms >>print top terms for each cluster.