Daniel Choo
CMSC476 – Information Retrieval
05/09/20

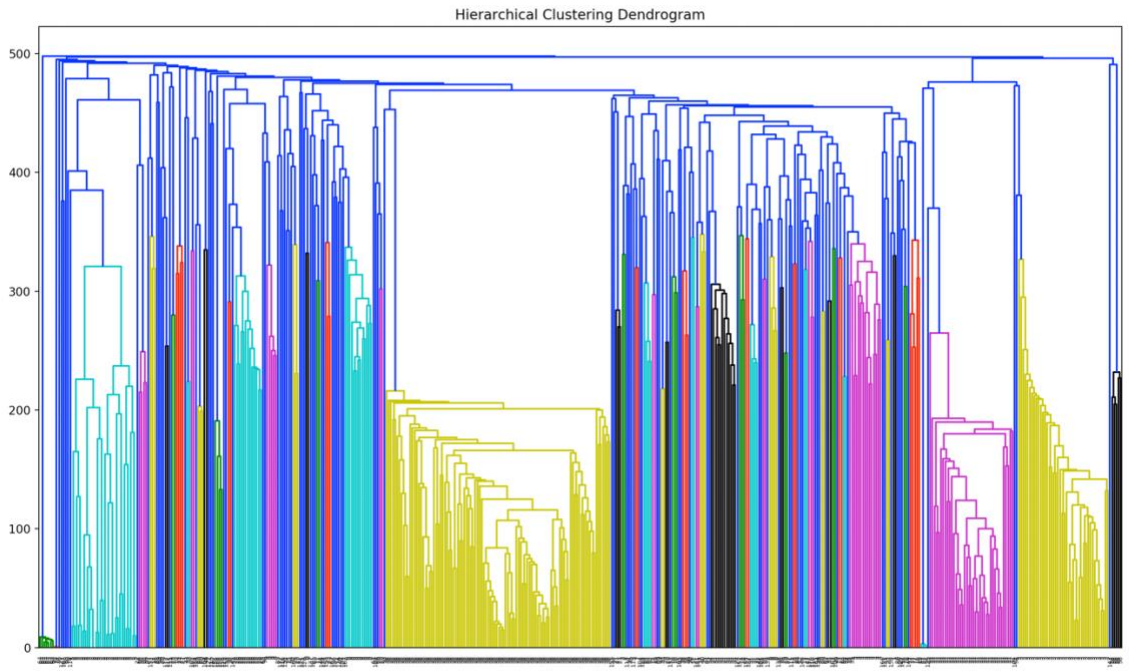
For Phase 5 for the project, I encountered some difficulties while coding up the project. More specifically, I encountered some problems creating the centroids and clusters while doing hierarchical agglomerative clustering. Due to this, unfortunately, I could not determine the which pair of HTML documents were most similar, dissimilar, or closest to the centroid. I believe I could not traverse the documentation that well and had some difficulties with the parameters as well. I created five new functions to create the clustering portion of this project: namely *open_postings( ), cosine_sim( ), matrix( ), plot_dendogram( ), clustering( ), cluster_strapper( ).*

The *cluster_strapper( )* is the bootstrapper for the entire clustering portion of the project. It will check whether or not there exists a file called similarity.txt; if this exists, the clustering will take less than five seconds. If not, then it will check if there is a file called postings.txt. If this exists, then it will call the *open_postings( )* function and it take approximately 30 seconds to calculate the TF-IDF of all of the documents. It will then conduct the cosine similarity calculation following the creation of the TF-IDF dictionary. If none of this exists, the program will prompt the user if they would like to engage in starting from scratch.

The most expensive computational process was creating the similarity matrix or the *matrix( )* function. Since there was approximately 40,000 words in our lexicon, the TF-IDF matrix size was $500\ x \sim 40{,}000$ and I constructed the similarity matrix from this intial matrix. This process took approximately 30 minutes to complete. If we had to do cosine similarity for each index, this would have taken an hour to complete. However, since the matrix is equivalent to its transpose, we can cut costs by taking the [*j*][*i*] of the matrix to be our [*i*][*j*]. The cosine similarity was calculated by taking the dot product of *a* and *b* and divide it by the norm of *a* and *b;* this was handled in the function called *cosine_sim( )* and was achievable using numpy's math and linear algebra modules. The similarity matrix had the size 500x500 as there were 500

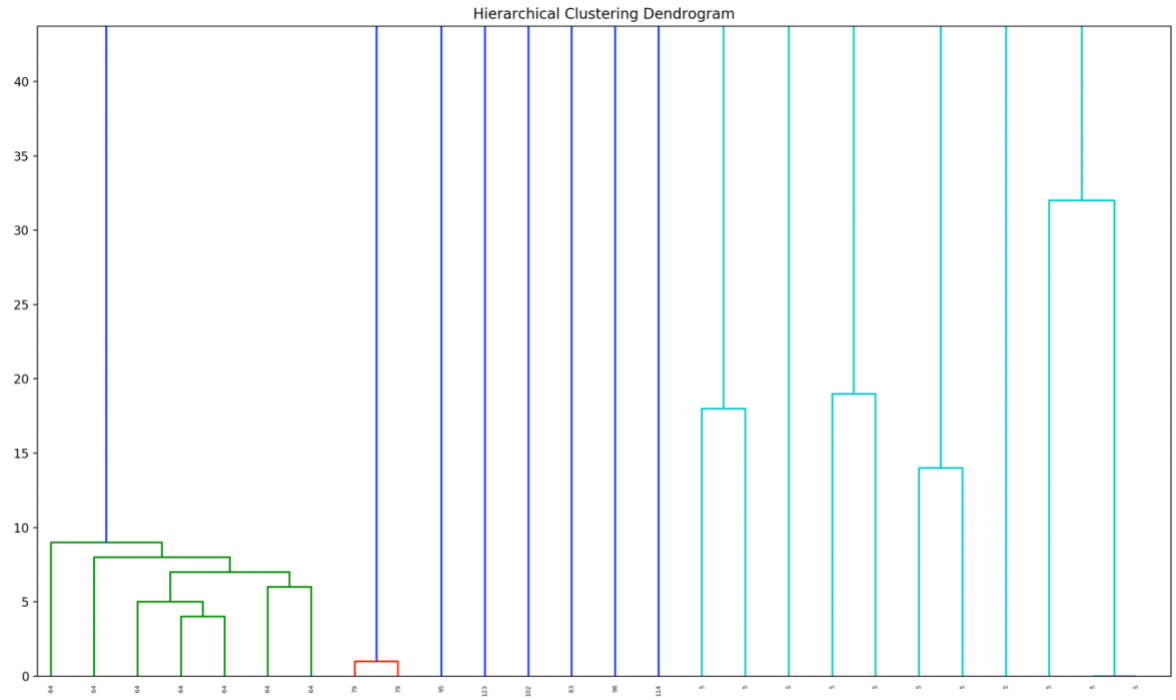
documents and we compared each document to each other. After we have created the similarity matrix, it will create a file, named similarity.txt, and call the clustering function to commence agglomerate clustering.

A contributor to scikit-learn, Mathew Kallada, posted on GitHub a way to demonstrate the clusters in a dendrogram. However, the output of this dendrogram is incorrect as it is demonstrating the document number on the *y*-axis and the clusters on the *x*-axis. Whereas, it should technically be document similarity (0.0 – 1.0) on the *y*-axis and the documents on the *x*-axis. Once we zoom into the dendrogram, we can also see some hidden clusters as the scale of the graph is so large. I believe all of the clusters are also color-coded in this dendrogram. However, I was not able to figure out how to traverse through the clusters as the documentation seems pretty sparse. I also could not find any other solutions people have posited online as no one seemed to have a problem similar to me. It appears that clusters were named arbitrarily by scikit-learn. Also, no other large data structures were used except Python lists and dictionaries.



Whole Dendrogram

Daniel Choo



Zoomed-in Dendrogram