CSDS 435: Project 2

Clustering Methods and Distance Measures for Finding Tweet Topics

John Mays, "Group 13"

Submitted 04/05/23 for Dr. Jing Li
Revised 12/17/23 for the NYC Office of Management & Budget

1 Introduction

In this paper, the initial task at hand is to inspect the tweets from the @cnn twitter account in order to assess what kind of topics CNN has tweeted about in the past. The accompanying research question is to ask what generic clustering methods work best for this task. I explore two clustering methods along with two distance measures. The two clustering methods I chose were k-Medoids and Complete-Link Hierarchical Clustering, because both algorithms accept pre-generated distance kernels. This was necessary for the concurrent goal of also studying distance measures. I compare the algorithms and the distance measures by metrics of separation and cohesion, and engage in hyperparameter tuning and visualization in order to explore these methods.

2 Methods

2.1 How is Data Stored?

It does not make sense to discuss the next section, "distance measures", unless how the data is formatted and stored is understood. The raw tweet data comes as a CSV table of tweets. A matrix, X is created to represent these tweets, where each row is a tweet and each column is a word in the total vocabluary (every word in the dataset without stopwords). Each entry then signifies the frequency of a certain word for a certain instance (tweet). To illustrate, if the entry at $X(0,0)$ is 1, this means that there is 1 instance of word 0 for the first tweet in the dataset.

2.2 Distance Measures

2.2.1 Normalized Unordered Edit Distance

$$d_1(x,y) = \frac{\sum_k x_k - y_k}{\sum_k x_k + y_k}$$

This is a new distance measure conceived by me. When considering how you compare strings, one is reminded of edit distance (both for words and sentences). Of course, edit distance requires an ordered string, and the bag-of-words model doesn't preserve the order; however, the difference between two vectors is essentially a version of edit distance that does not pay attention to order (hence "unordered"). Consider this as well: two four-word sentences with two words in common are about as near as two six-word sentences with three words in common. The other distance measure in this paper *does not* treat the data this way. Hence, the distance is normalized through the operation of dividing by the difference by the total number of words in both. This distance measure is essentially

$$\frac{\text{\# of words not in common}}{\text{\# total number of words in both sentences}}$$

Therefore, the minimum value ($d_1(x,x)$) is zero, and the maximum value (for two vectors with no words in common) is one.

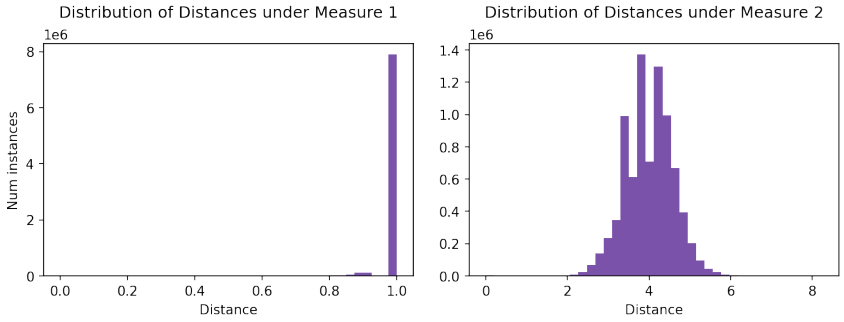
2.2.2 Euclidean Distance

$$d_2(x,y) = \sqrt{\sum_k (x_k - y_k)^2}$$

Euclidean distance seemes like an intuitive choice. It will, as opposed to normalized unordered edit distance, generally attribute greater distances to longer tweets. Two totally different sixty-word tweets will be much farther apart than two ten-word tweets (assuming they each have nothing in common).

2.2.3 Comparing Distributions

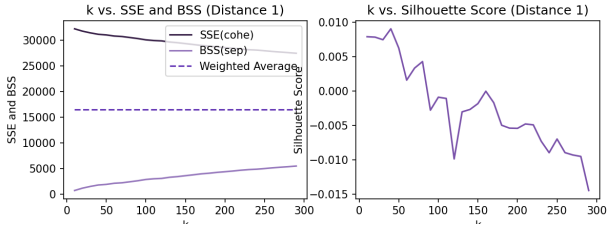
The distributions of the distance measures across the dataset are extremely different. The Euclidean distance (d_2)'s distribution is approximately Gaussian with $\mu \approx 4$, while d_1 has a outlying concentration of 1.0 (no words in common) and then just a few thousand distances clustered around 0.9. This implies that most tweets used entirely different sets of words, and when they had words in common, they were not often more than a few.

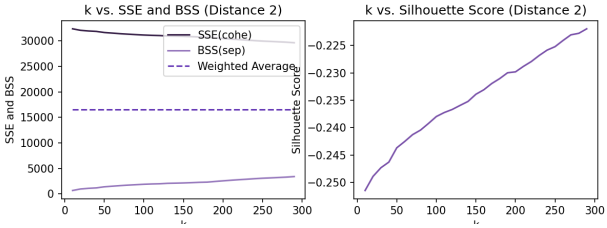


2.3 Choosing Hyperparameters

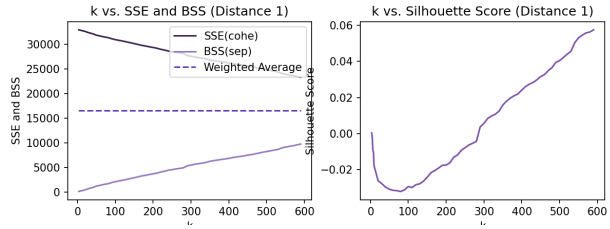
Both of my clustering methods required me to choose k, the number of clusters, which is also considered the primary hyperparameter for both. In order to do this, I tried to observe how different metrics and objective functions varied with k. Some results were surprising, and revealed to me that different distances worked very differently with different clustering algorithms. Observing the sum-squared error (SSE, a measure of intra-cluster cohesion) and the between-cluster sums of squares (BSS, a measure of inter-cluster cluster separation) turned out not to be very helpful, but observing the silhouette score was. In general, a silhouette score closer to one is better. The only two combinations that demonstrated a positive correlation between silhouette score and k were {Distance 1 and Hierarchical} and {Distance 2 and k-Medoids}. The other two were negatively correlated, and less so. For k-Medoids I chose a default k of 50 as that is the max silhouette achieved with either distance. And for Hierarchical, smaller choices were not producing coherent clusters, so I chose a much higher k based on the d_1 figure: 600.

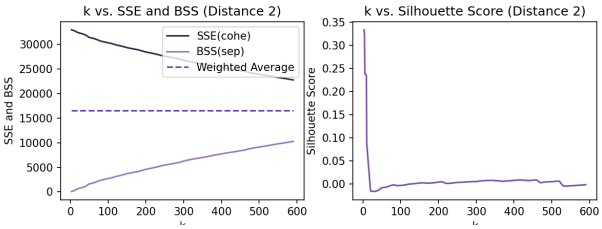
2.3.1 Choosing k for k-Medoids:





2.3.2 Choosing k for Hierarchical:





3 Data

The data is initally a table of tweets tweeted by @cnn given to the class by Dr. Li.
Note: consult section 2.1 for an explanation of how the data is stored in the program.

3.1 Dataset Summary

# Num of Tweets	# Num of Words (total)	# Num of Tokens	# Avg Num of Words
4,061	32,612	9,609	8.031

Top Ten Legal Tokens:

"health", "getfit", "new", "@cnnhealth", "today's", "cancer", "know", "kids", "@drsanjaygupta", "ebola"

3.2 Stopwords:

Initially, I tried manual stopword selection, but I found this task to be cumbersome, and to leave in less frequent stopwords that were still important. So I chose most of the stopwords from the *Natural Language Toolkit* (NLTK), electing to leave a certain few out, while also trying my best to include ones specific to the dataset. After subtracting the NLTK words, I scanned through the most popular 100 words still left. The words *"rt"* (retweet) and *"w/"* (with) were removed, along with spurious characters that somehow made it through my regex cleaning.

When choosing what to keep and what to remove, I had a difficult time deciding. Common stopwords like *"was"* and *"is"* contain tense information that could hypothetically be related to a cluster's meaning, but I chose to go with precedent, which suggests that they are more noise than they are worth. I elected to remove the URLs since they were almost never in common between tweets. However, I chose to leave "@" mentions and "hashtags" in. I removed the hashtag character, however, as I judged "#ebola" and "ebola" could be treated as one & the same. Hashtags and *"@"*-mentions were left in because I thought that clusters could potentially be formed around them. Quite often, a hashtag's sole purpose is to distill the subject or message of the tweet. And although not all textit"@"-mentions feature a certain doctor or expert, the ones that do and are the same are intrinsically related to one another.

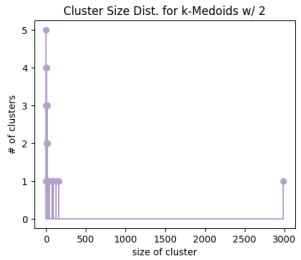
4 Results

k-Medoids with d_1:

Number of Clusters	SSE	BSS	Silhouette Score
50	31061.623	1960.594	0.006

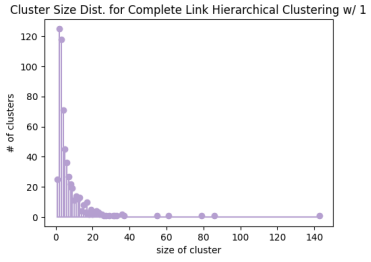
k-Medoids with d_2:

Number of Clusters	SSE	BSS	Silhouette Score
50	31631.14	1391.078	-0.244



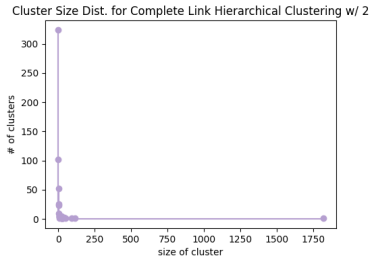
Hierarchical with d_1:

Number of Clusters	SSE	BSS	Silhouette Score
600	23151.933	9870.284	0.058



Hierarchical with d_2:

Number of Clusters	SSE	BSS	Silhouette Score
600	22643.094	10379.124	-0.001



4.1 Cluster Deciphering

For Complete-Link Hierarchical Clustering, I had what looked like the best clusters, so here some of them are:
Note: this was run with $k = 600$ and d_1.

Cluster:	**Cluster 3**	**Cluster 1**	**Cluster 109**	**Cluster 86**
size:	143	86	61	55
Popular tokens:	getfit	health	cancer	ebola
	today's	minute	breast	american
	eat	insurance	one	@jechristensen
	make	good	survival	infected
	eating	weight	scary	@elizcohencnn
	calories	crisis	prevent	ebolaqanda
	@shape_magazine	today's	new	@who
	day	loss	come	patients
	every	know	@cnnhealth	like
	healthy	food	you?	experimental

There are some really nice clusters here. **Cluster 3** has a prevailing theme of diet and weight loss. **Cluster 1** seems to be about modern weight problems. **Cluster 109** is mostly scare-tweets about breast cancer and survival rates. **Cluster 86** is about the ebola outbreak in America.

4.2 Cluster Consistency:

I used entropy and purity to compare k-Medoids to Complete Link Hierarchical Clustering, both using distance measure 2 (Euclidean). Here are my results:

- When k-Medoids is the label, Entropy = 0.822, Purity = 0.81
- When Complete Link Hierarchical Clustering is the label, Entropy = 3.164, Purity = 0.457

5 Conclusion

Even with our self-imposed limitation of only kernel methods, we managed to run a successful comparison and try a somewhat novel distance measure. Keeping the limited scope of this report in mind, at least on the two distance measures here, Hierarchical clustering (with Complete Linkage) seemed to consistently outperform k-Medoids in measures of cohesion and separation, silhouette score, and common understanding. The clustering was generally much more coherent coming from Hierarchical Clustering, *especially* with distance metric one. Although this report is by no means a complete dive into clustering, let alone these two metrics, it gave us some insight into how the two methods work in limited cases, and despite having a questionable distribution at first, novel distance measure d_1 = Normalized Unordered Edit Distance proved functional and excelled against Euclidean distance.

Statement of Participation

- John Mays: 100%

All team members agree with the specified effort.

Sources

[1] *Text pre-processing: Stop words removal using different libraries(NLTK)*
[2] *Introduction to Data Mining (2nd ed.)* Tan et al.
[3] *Fuzzy Approach Topic Discovery in Health and Medical Corpora* by Karami, Gangopadhyay, Zhou, and Kharrazi