ASSIGNMENT-1 CLUSTERING

Group members : Xinzhe Deng, Yogeshwar Kansara, Aarthi Mahalakshmi Shankar

Workflow

1) Importing Data:

We used python to import all the .txt files as a pandas dataframe and also added the file name as another column to identify different classes (accounts)

2) Visualization Task :

After importing the data as a dataframe, we calculated the probable most common words in all the 10 accounts. Below are the top 10 words from all accounts :

i) KaiserHealthNews:
['health', 'report', 'insur', 'today', 'law', 'care', 'plan', 'hospit', 'state', 'medicar']
ii) NBChealth:
['studi', 'ebola', 'fda', 'health', 'cancer', 'kid', 'flu', 'work', 'help', 'food']
iii) bbchealth:
['video', 'ebola', 'nh', 'cancer', 'health', 'care', 'hospit', 'audio', 'patient', 'death']
iv) cbchealth:
['ebola', 'health', 'canada', 'doctor', 'cancer', 'outbreak', 'patient', 'studi', 'hospit', 'canadian']
v) cnnhealth:
['today', 'tip', 'help', 'amp', 'kid', 'health', 'know', 'make', 'brain', 'stori']
vi) everydayhealth:
['food', 'eat', 'health', 'way', 'weight', 'healthi', 'help', 'today', 'amp', 'diet']
vii) foxnewshealth:
['studi', 'help', 'risk', 'drug', 'patient', 'cancer', 'brain', 'diseas', 'heart', 'doctor']
viii) gdnhealthcare:
['nh', 'health', 'patient', 'miss', 'care', 'need', 'today', 'work', 'healthcar', 'nurs']
ix) goodhealth:
['tri', 'make', 'food', 'way', 'healthi', 'day', 'help', 'amp', 'eat', 'weight']
x) latimeshealth:
['studi', 'health', 'help', 'say', 'research', 'cancer', 'risk', 'drug', 'make', 'peopl']
xi) msnhealthnews:
['studi', 'risk', 'cancer', 'kid', 'heart', 'help', 'link', 'drug', 'women', 'patient']
xii) nprhealth:
['health', 'ebola', 'doctor', 'help', 'insur', 'care', 'patient', 'risk', 'drug', 'hospit']
xiii) nytimeshealth:
['health', 'ebola', 'age', 'doctor', 'cancer', 'old', 'care', 'patient', 'blog', 'drug']
xiv) reuters_health:
['ebola', 'drug', 'studi', 'health', 'patient', 'cancer', 'case', 'fda', 'hospit', 'risk']

xv) usnewshealth:
['amp', 'diet', 'eat', 'food', 'heart', 'know', 'health', 'way', 'healthi', 'help']
xvi) wsjhealth:
['health', 'drug', 'ebola', 'amp', 'law', 'fda', 'insur', 'cancer', 'patient', 'pharmalot']

We graphed the probability of occurrence for the 10 most common words from each Twitter account. Fig-1 to Fig-16 shows the bar chart for top 10 most occuring words in each of the 16 accounts. On the x-axis, we have terms and on the y-axis we have the frequency calculated by TfIdfVectorizer.

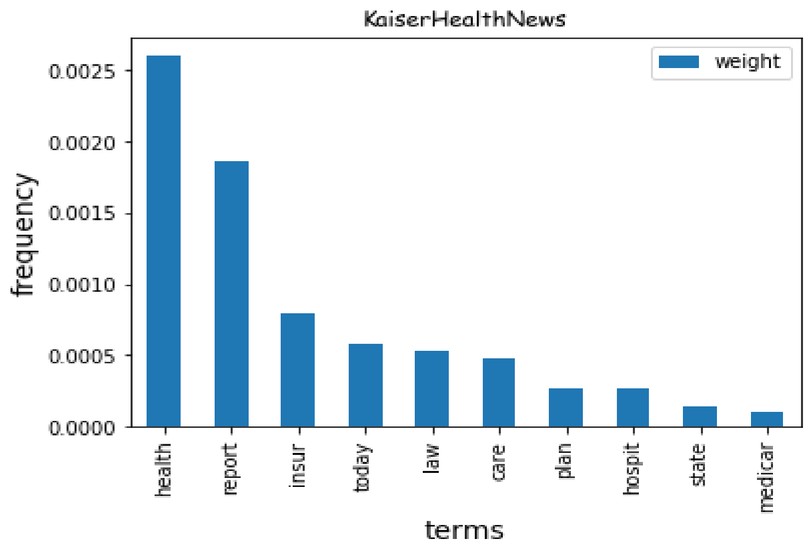

Fig-1

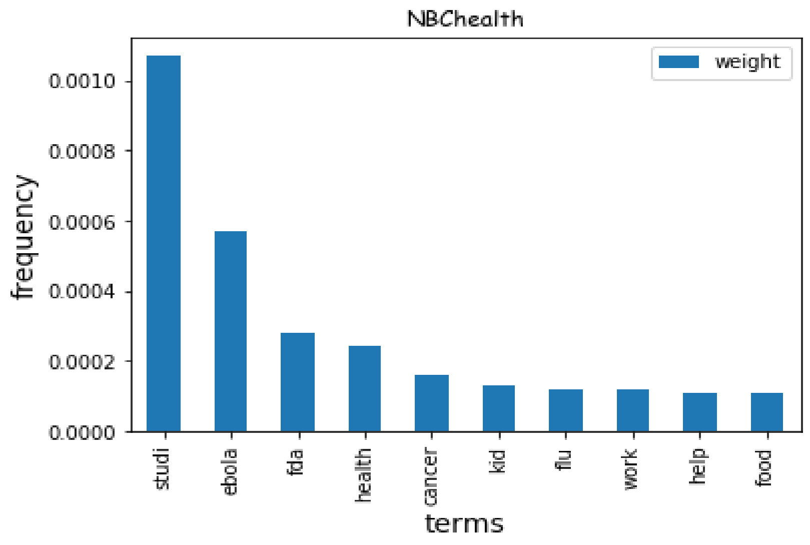

Fig-2

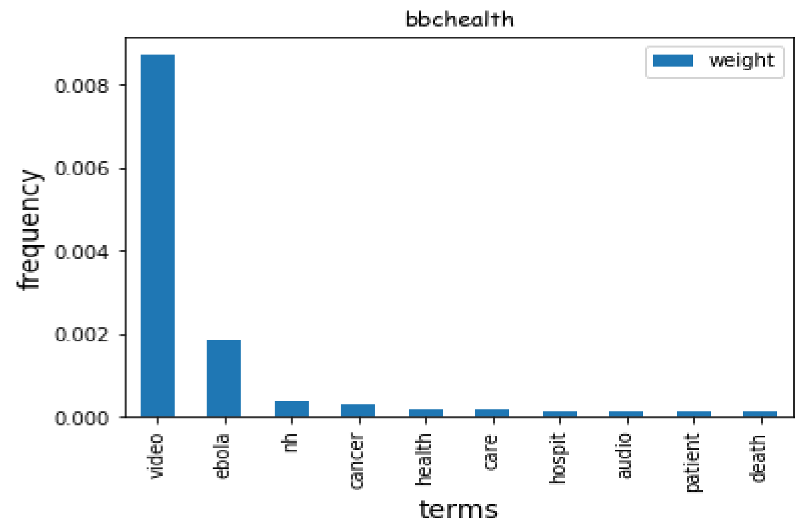

Fig-3

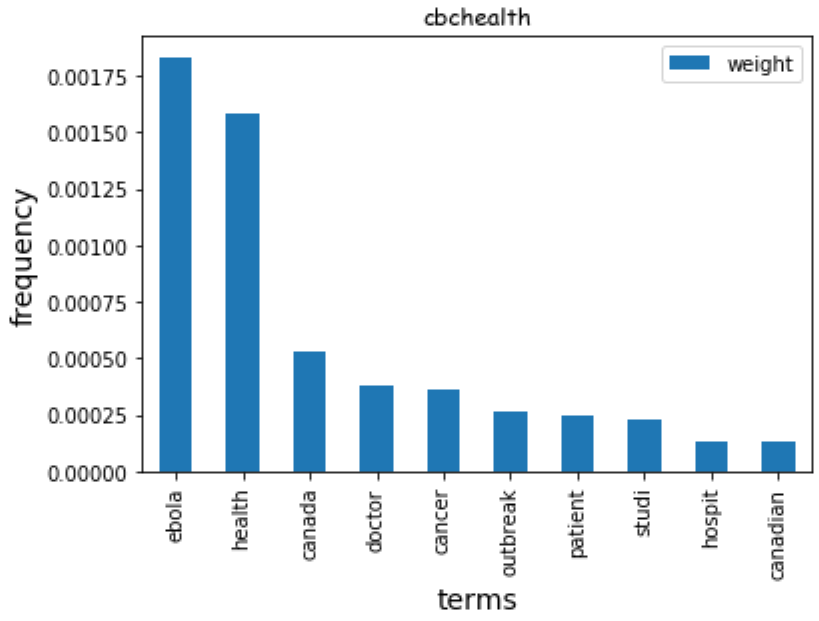

Fig-4

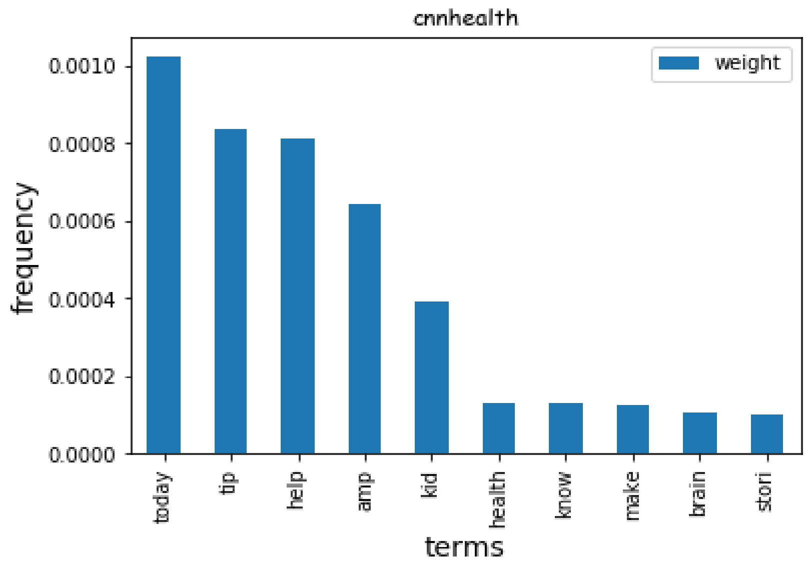

Fig-5

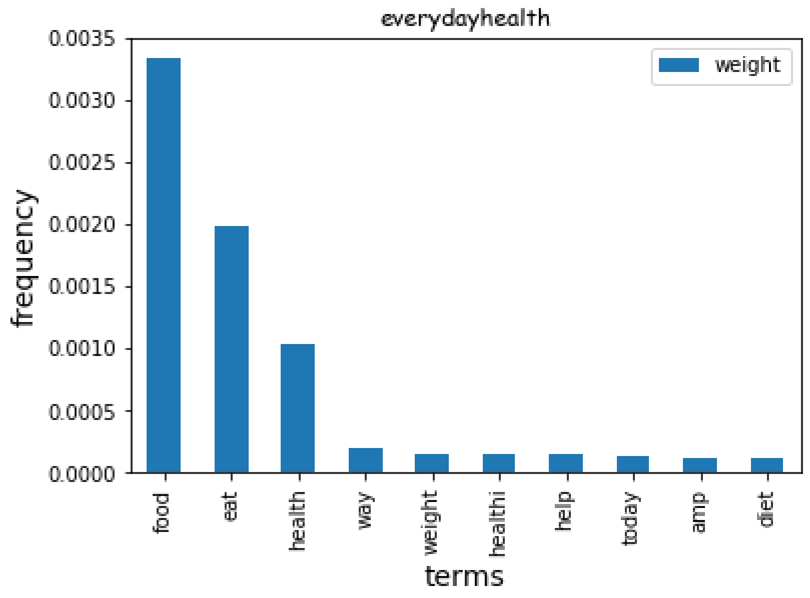

Fig-6

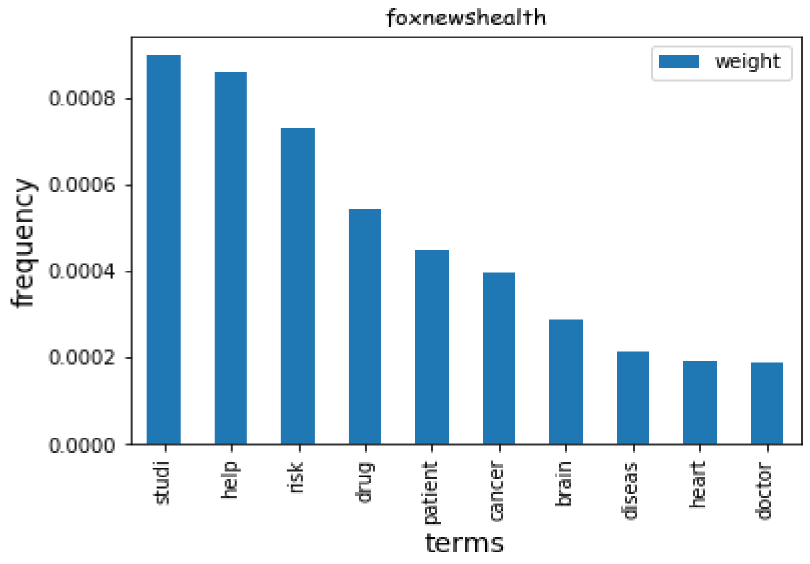

Fig-7

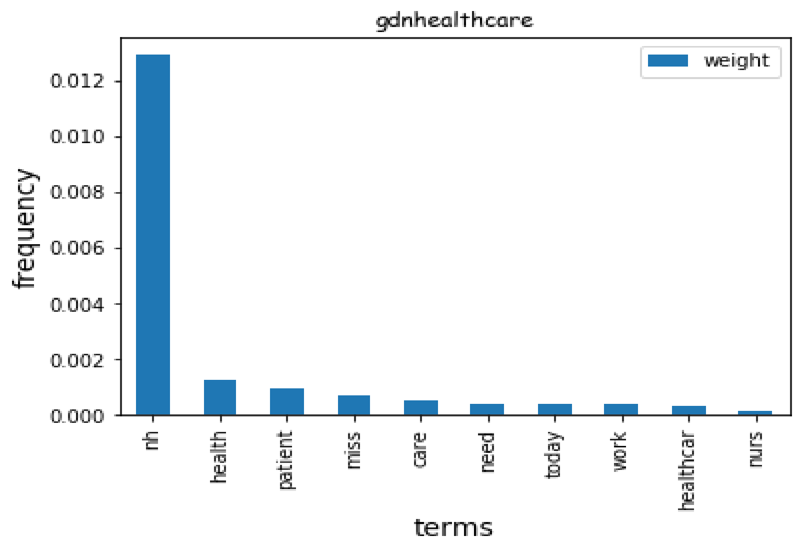

Fig-8

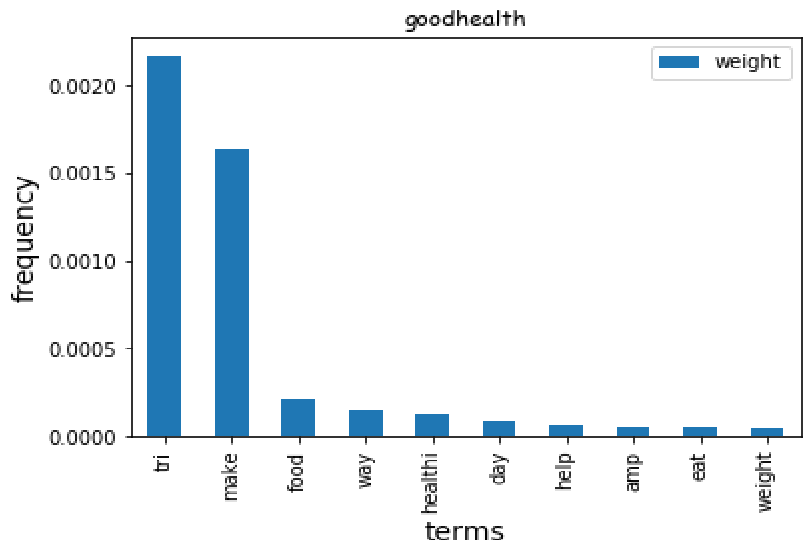

Fig-9

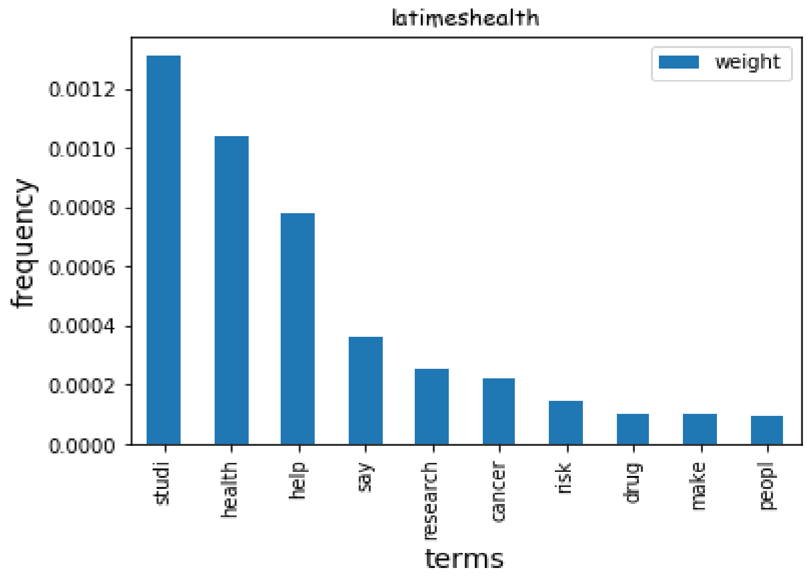

Fig-10

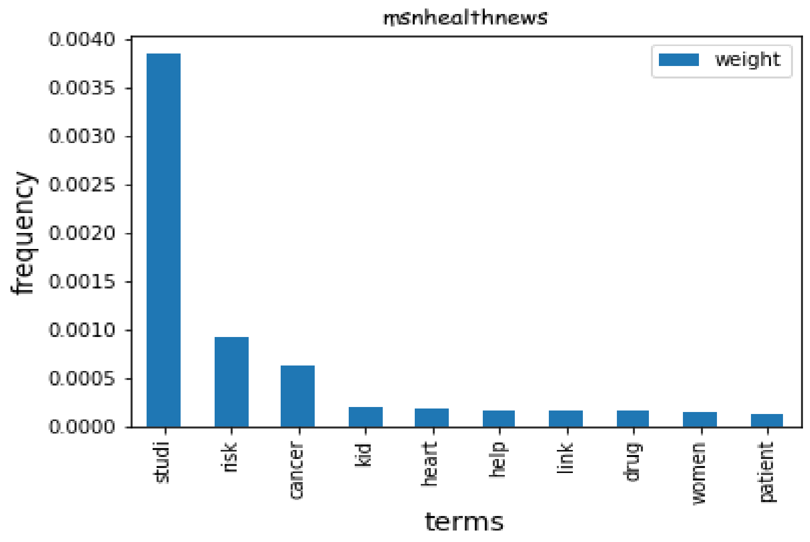

Fig-11

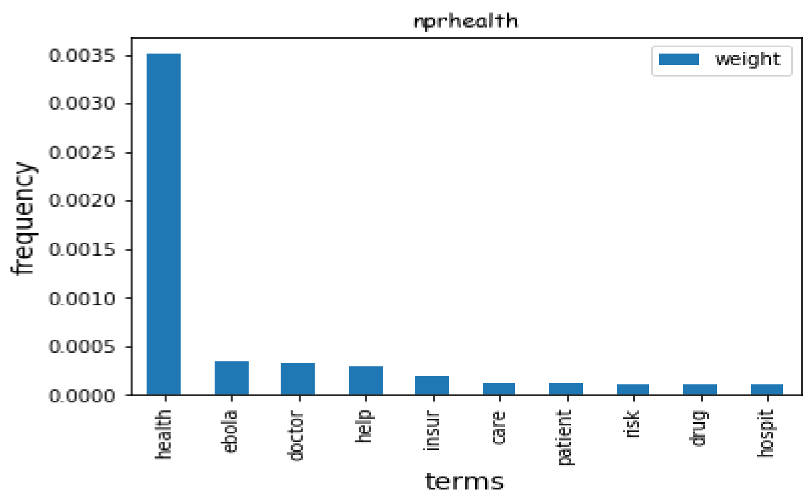

Fig-12

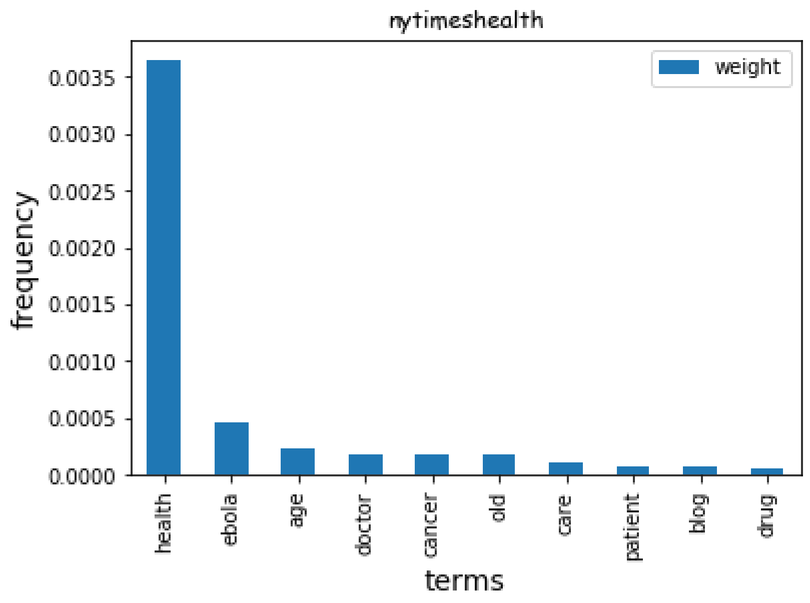

Fig-13

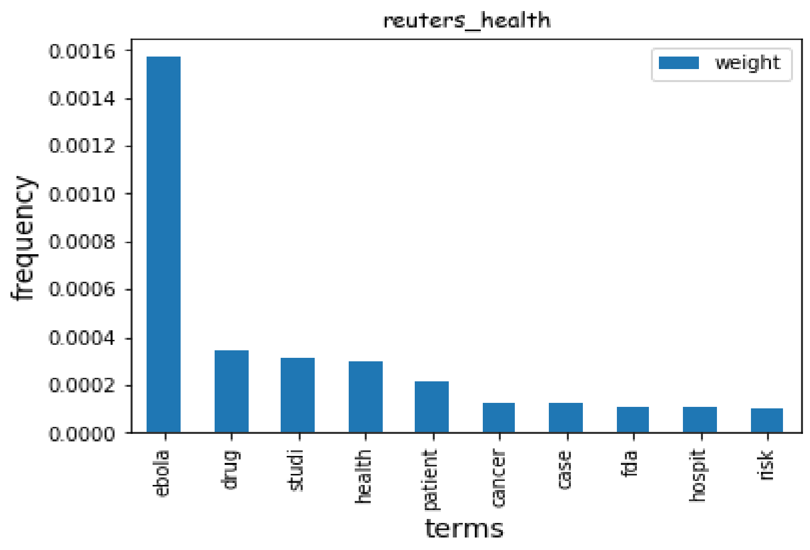

Fig-14

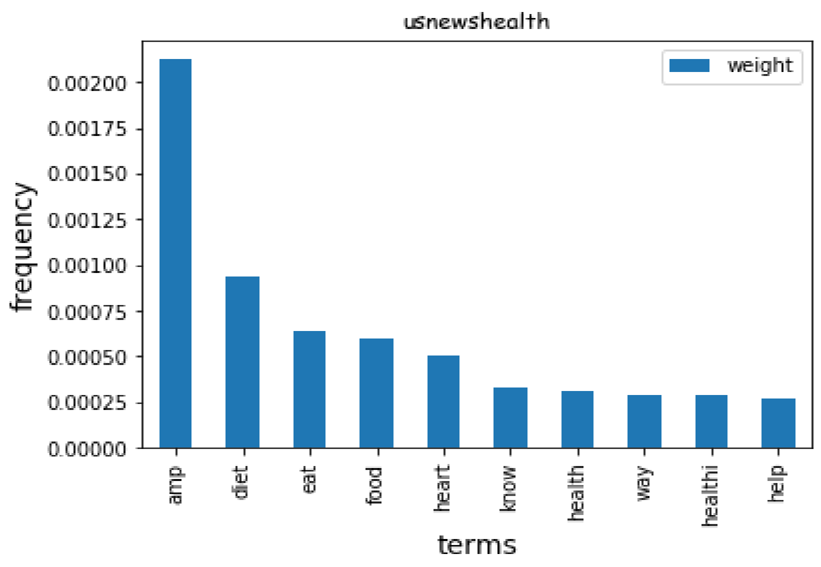

Fig-15

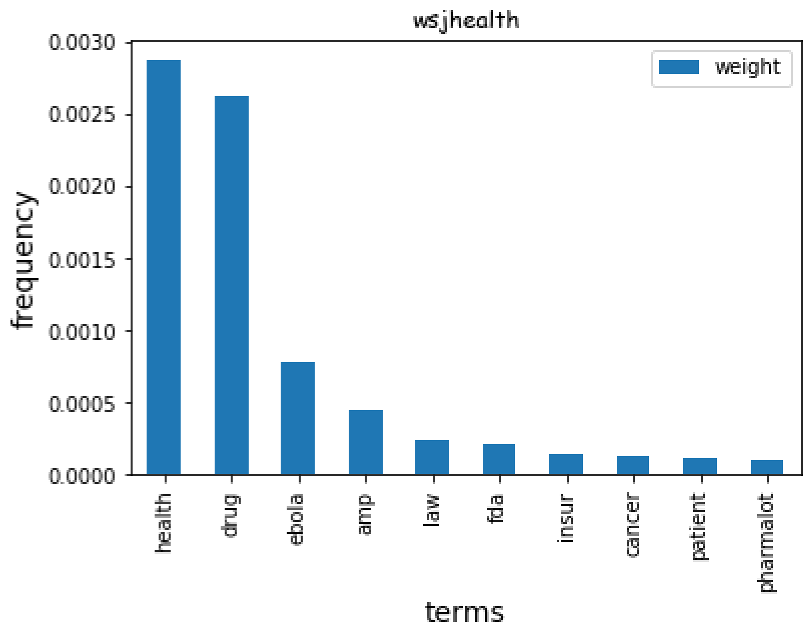

Fig-16

Q 1) Are these most probable words related to health?

From the above graphs, we can see that some of the terms that occur frequently in the tweets are not related to health. The two words that are repeated here most are video and audio , which are given in a particular text format (VIDEO:, AUDIO:) without any pre-processing. These words needs to be removed to get more health related terms.

Q 2) If not, can you propose a way to improve the results?

In order to improve the results, we did data cleaning and preprocessing. Used Tfidf weights to find out the importance of the word in the document and generate the top 10 common words. The steps for data cleaning and preprocessing of data and elucidated below:

Data Cleaning and preprocessing:

1. Preprocessed the data into a dataframe containing ID, date/time and Text by using regular expression for every tweet
2. Removed the URL from content
3. Removed stop words, common words - Sometimes, some extremely common words which would appear to be of little value in helping select documents matching a user need are excluded from the vocabulary entirely. These words are called stop words. The general strategy for determining a stop list is to sort the terms by collection frequency (the total number of times each term appears in the document collection), and then to take the most frequent terms, often hand-filtered for their semantic content relative to the domain of the documents being indexed, as a stop list , the members of which are then discarded during indexing.
4. Removed all punctuations
5. Removed mentions (Ex: @)
6. Used Stemming - PorterStemmer to change all words to their base or dictionary form.
   Stemming is useful because it changes all words in a text into its root form. When all these words get changed, we get words that are mostly easy to read and process.

   The process of linguistic normalisation, in which the variant forms of a word are reduced to a common form is what a stemming algorithm does. One example is given below:
   connection
   connections
   connective ---> connect
   connected
   Connecting

   Stemming usually uses rules that are language specific and does not need to have a complete knowledge of complete vocabulary.

7. Figure-17 represents the box plot of the number of characters in every tweet.

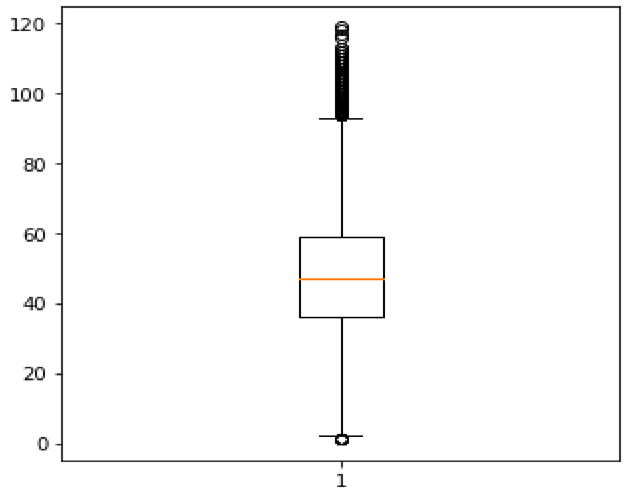

Fig-17

Below is the description of the given box plot:

	Detail_length
count	63326.000000
mean	48.935398
std	17.176790
min	0.000000
25%	36.000000
50%	47.000000
75%	59.000000
max	119.000000

8. Used TfidfVectorizer from sklearn and got the weights of top 10 words in separate documents as well as from the total concatenated document.

Fig-18 shows the word cloud of the top most common words from all accounts consolidated together:

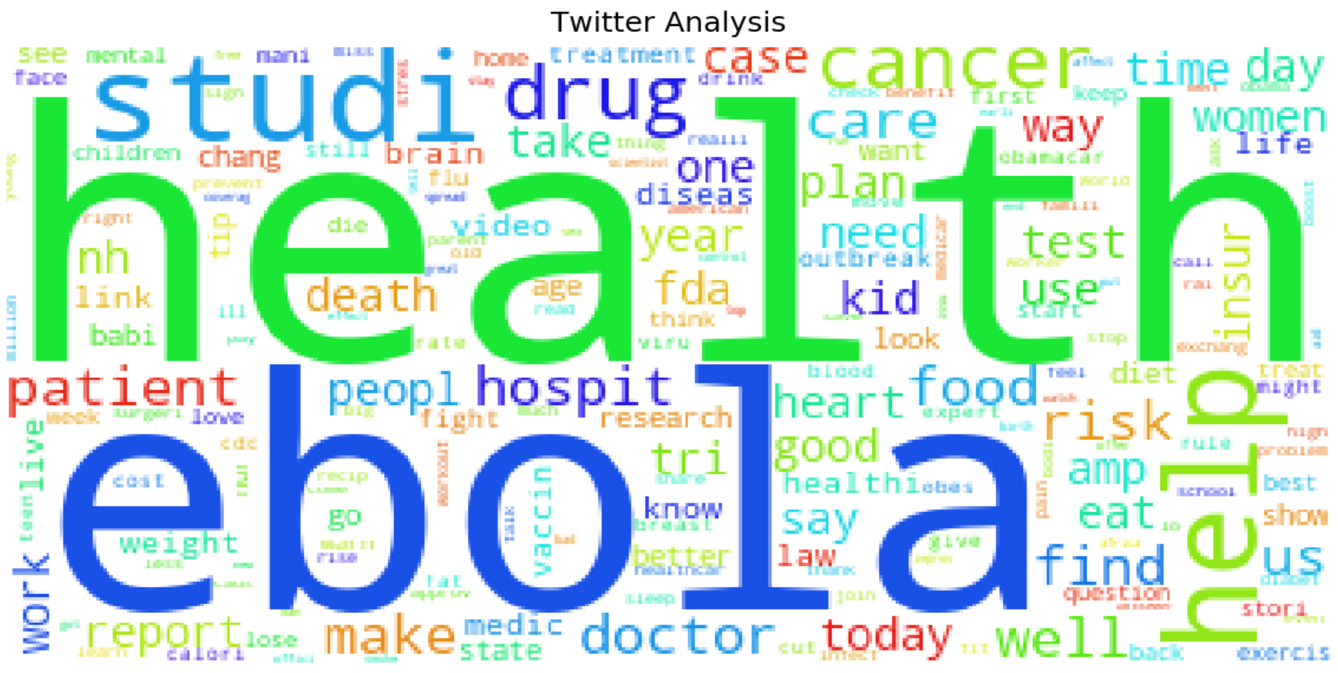

Fig-18

Clustering Task:

Feature extraction:

TF-IDF :

The Term Frequency-Inverse Document Frequency is a numerical statistic that determines the importance of a word in a document. This value increases proportionally with the number of times that particular word appears in the document. These scores can be used for filtering stop words as well.

Term frequency

To find out if a certain phrase is in a set of documents, for example “The brown fox”, the most easiest way to start is to eliminate the documents that do not contain the words in the phrase. Once those documents are eliminated, there will still be more documents left that contain the required phrase. In order to differentiate them, counting the number of times a particular term appears in a document refers to term frequency.

Inverse document frequency

Since stop words like ‘The’ in the phrase “The brown fox” are so common, term frequency calculates the number of times ‘the’ appears by giving it more importance, which is wrong. Thus inverse document frequency takes off the weight of the terms that occurs very frequently and increases the weight of the terms that occur rarely, that is the terms that add value to the document or the phrase.

Tokenized the words in the cleaned tweets and calculated TF-IDF scores with maximum of 15000 features. Used those features and performed PCA (Truncated SVD) to reduce the dimensionality to 2. PCA did not work in this case because the TF-IDF features gave us a sparse matrix and PCA doesn't take in a sparse matrix. Truncated SVD in particular is used on term count/tf-idf matrices as returned by the vectorizers.

Finally, performed k-means clustering on the transformed data with number of clusters as 16. We created a pipeline of truncated SVD and k-means and predicted the labels from that. We also found centroid of these clusters. The resulting output was used to create a graph in 2-dimension.

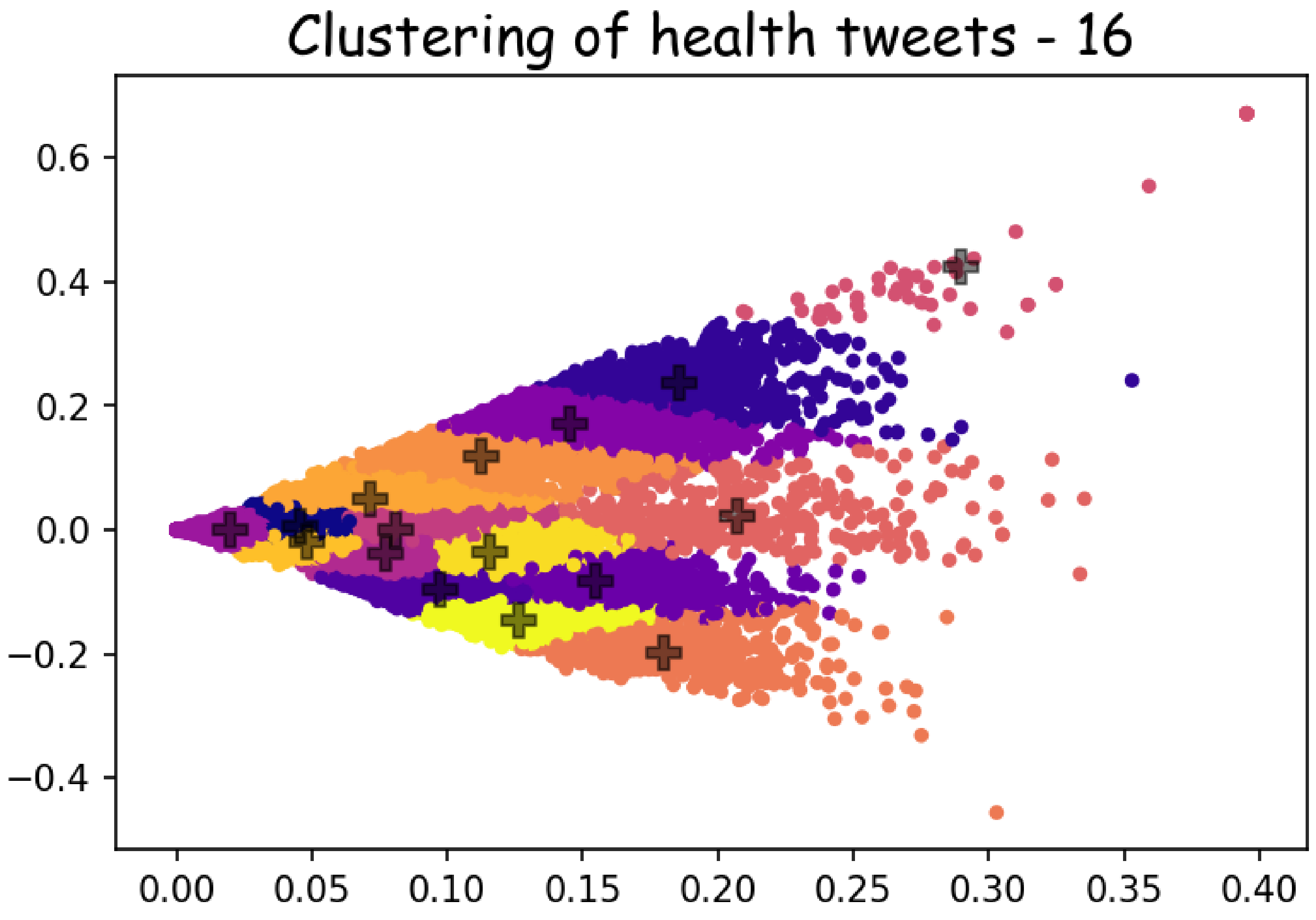

Fig-19

Q 3) Does each Twitter account form its own cluster? Why or why not is this the case?

No, from the two dimensional scatter plot, we don't think each account has formed its own cluster. We performed SVD on the Twitter content using parameter 'n_component' = 2 to get two of the most representative words in the text. After word dimension was reduced to 2, we tried to plot a scatter plot for these two selected dimensions. From the scatter plot, it is obvious that there is a lot of overlapping among two or more different accounts. In order to verify that it is not the noisy words that affect the result, we cleaned the text again by selecting more words, which seemed to be irrelevant and showed words that were not related to the topic of health. After that, we plotted 2D scatter again and the clusters seemed to keep overlapping with each other. We also tried different combinations among top 10 principal components, but overlapping kept happening. Consequently, we made a conclusion that it may be because of the short length of tweets, there is not enough information provided to distinguish each account according.

Q 4) Optional Task:

Change one of the parameters used in the clustering task. How do your results differ, and why?

We changed the number of clusters to 8 and recreated the clustering again using the same procedure described above.

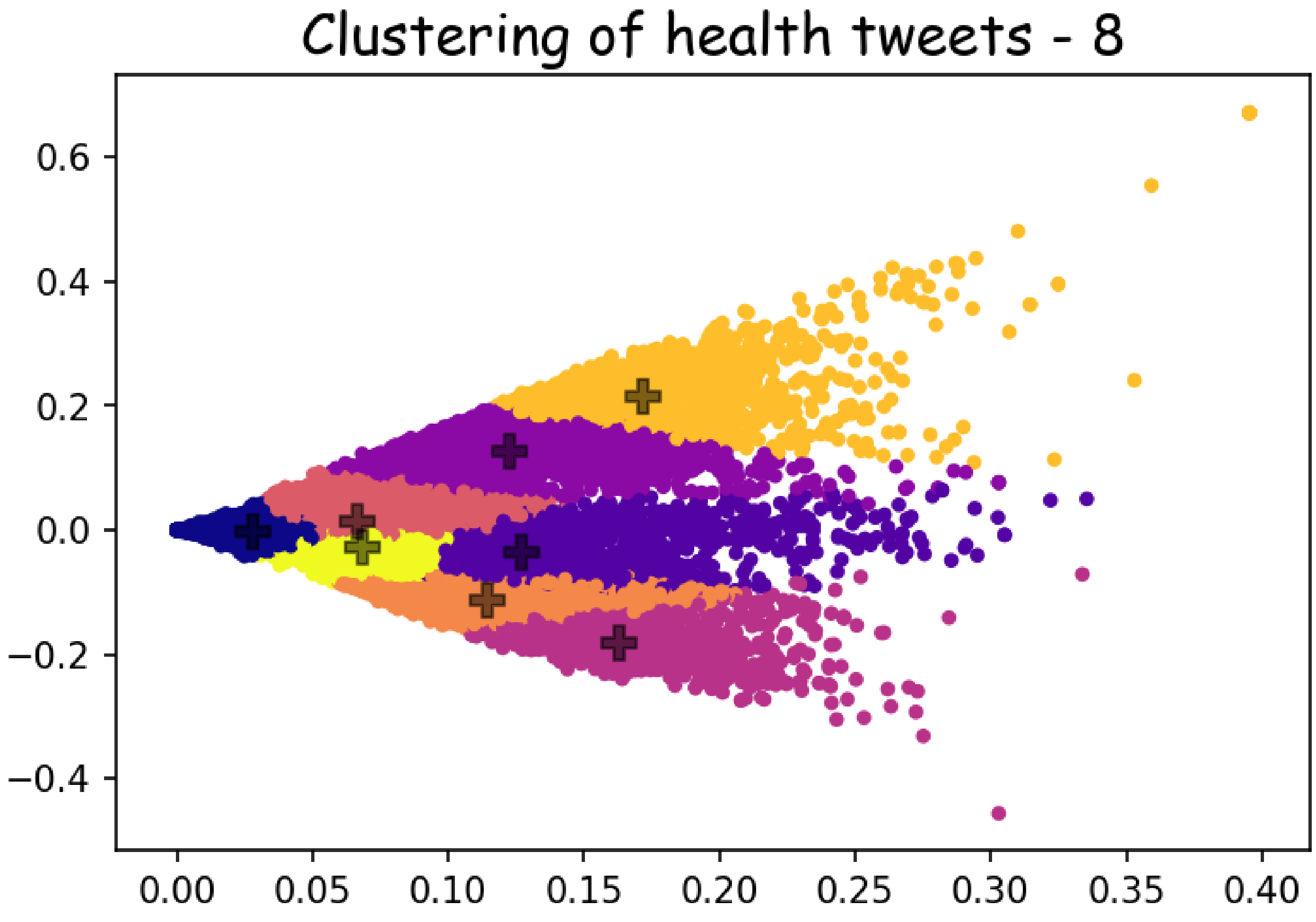

Fig-20

By reducing the number of clusters, we can see that some words have been merged into a single cluster and the words have been identified together. The clusters can now be easily identified with respect to the centroid.