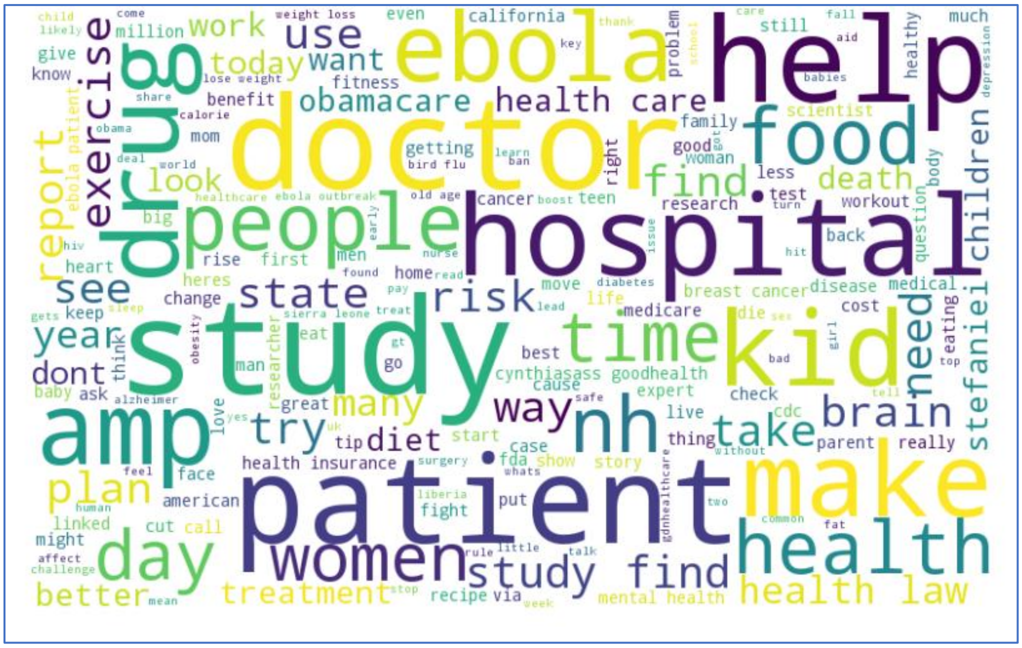
We have implemented this project on Python and its associated packages like Pandas, NumPy, Matplotlib, NLTK, WordCloud and Sklearn. The wordcloud python package which we used can be found here ( Word Cloud Package ).

### Initial Analysis

1. We imported all 16 datasets at once and cleaned the data. Cleaning the data included removing the Tweet ID, Timestamp before the tweet and the @ symbol as well that are generally used in many tweets. We also removed the hyperlinks in any of the tweets as well.

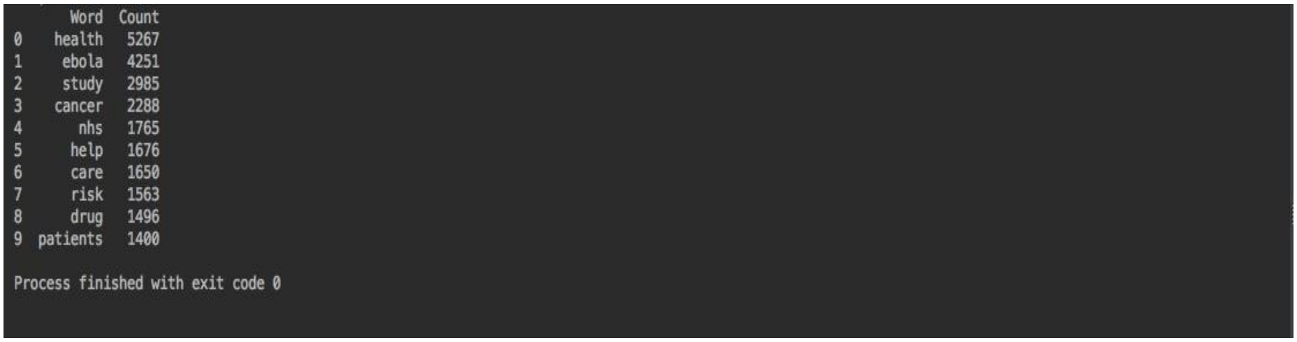
2. We cleaned the text we got by removing the punctuations and converted the entire text to lower case so that model does not identify the same words with different cases as two different words. We also removed the stop words that does not add any meaning to the text. Following the data cleaning we tokenized the text which essentially means breaking the sentences into pieces such as individual words.

3. We tried to build a wordcloud to get an overall picture of the text which is attached below.



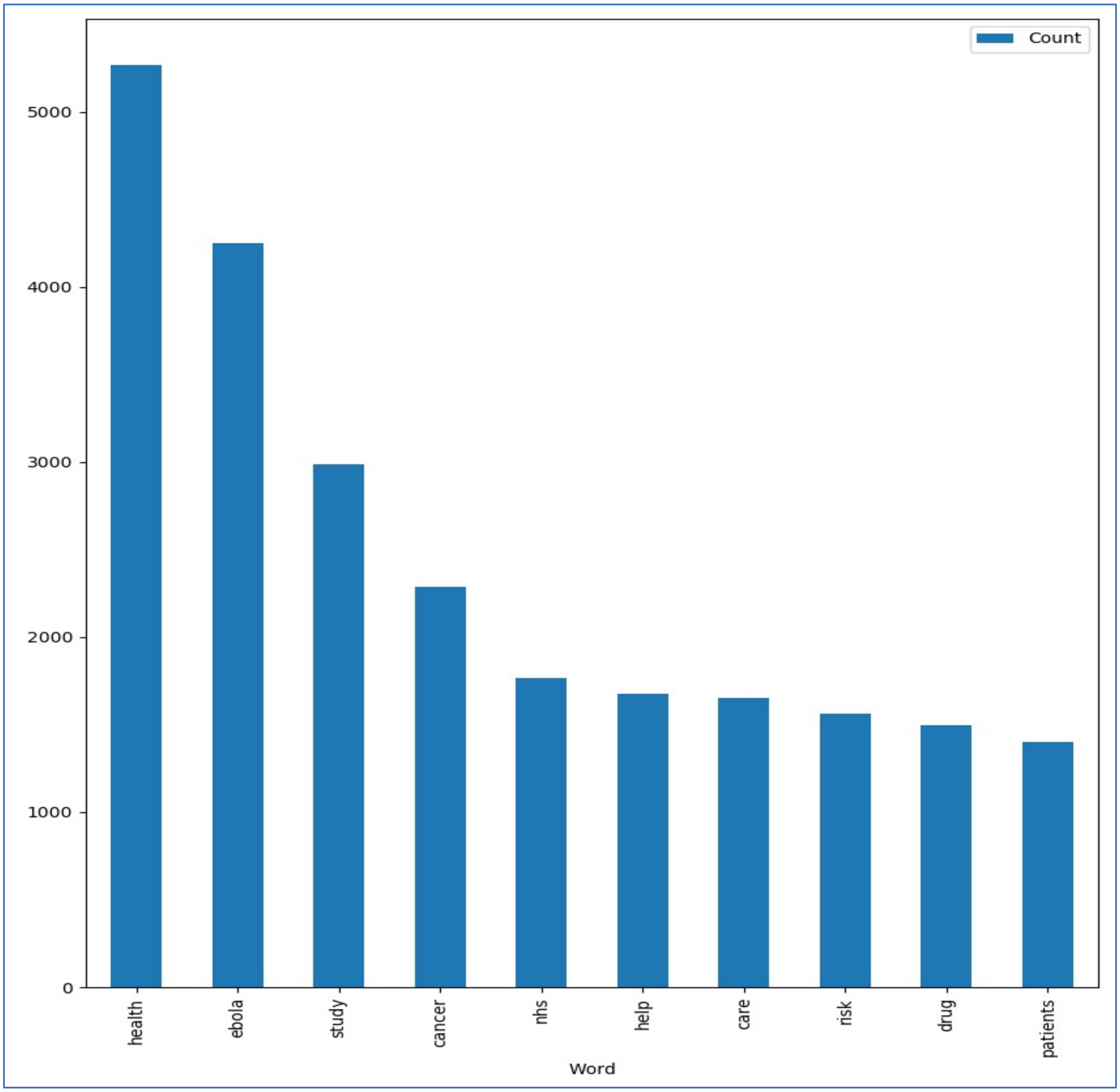
As it can be seen a lot of words related to health like hospital, patient, doctor can be seen in this word cloud.

4. Top 10-word frequency list was built and below is the list and count of the most frequent words,



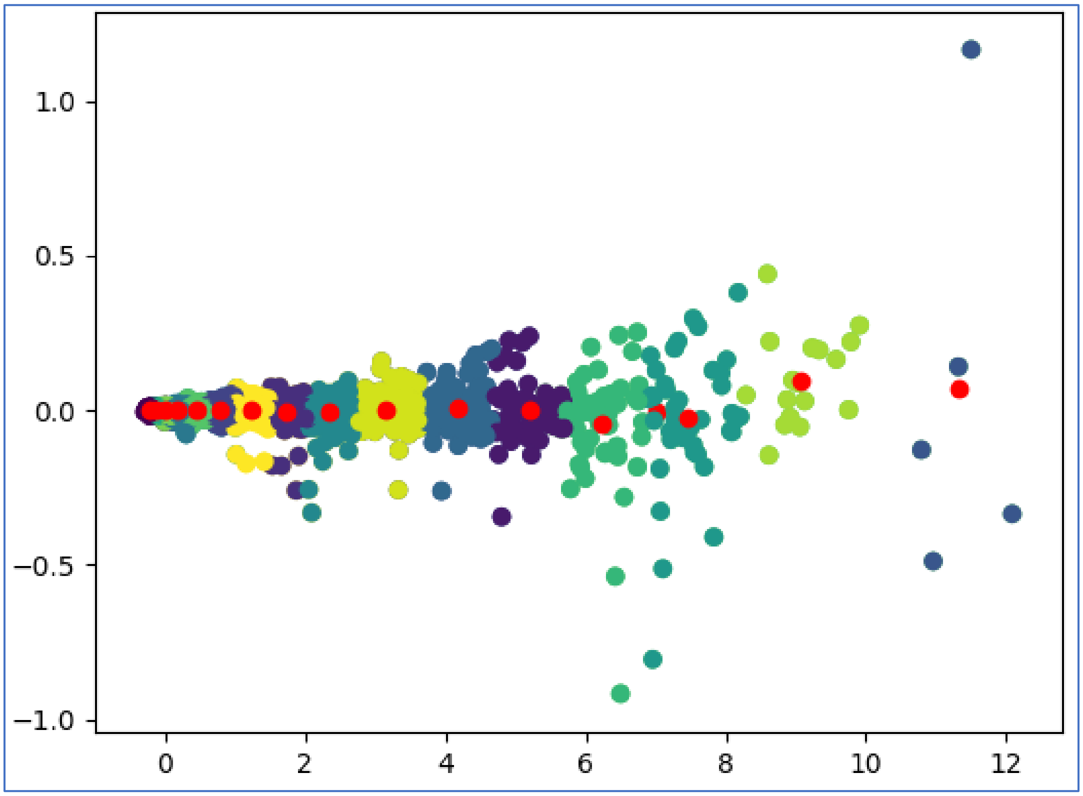
```
       Word  Count
0    health   5267
1     ebola   4251
2     study   2985
3    cancer   2288
4       nhs   1765
5      help   1676
6      care   1650
7      risk   1563
8      drug   1496
9  patients   1400

Process finished with exit code 0
```

Below is the plot of the 10 top word frequency.



## Clustering Task

1. After the initial analysis, we have cleaned the text and converted it into vectors now. Upon vectorizing the dimension of text data was very large which leads to the usage of PCA for dimensionality reduction. We reduced the dimensions and then applied KMeans. Principal Components Analysis (PCA) is a technique that finds underlying variables (known as principal components) that best differentiate your data points. Principal components are dimensions along which your data points are most spread out. PCA is very handy when reducing the dimensions and removing unwanted features. It is one of the methods of feature extraction which creates new combination so attributes from the features.
2. After performing PCA, we implemented KMeans using the SciKit learn. K-means clustering is unsupervised learning algorithm. It works by classifying a given data set into several clusters, defined by number of clusters which is defined beforehand. The clusters are then positioned as points and data points are associated with the nearest cluster, computed, adjusted and then the process starts over using the new adjustments until a desired result is reached. Since we need to give the number of clusters before, it was essential to perform the PCA before implementing KMeans.
3. As per the instructions we performed the KMeans with 16 as number of clusters and the plot can be seen below.



The red points are the centroids and clusters formed around it.

From the above plot each cluster is not related to the twitter account and there is no correlation between them because when we vectorized the data for the model to work, the account information is not maintained.

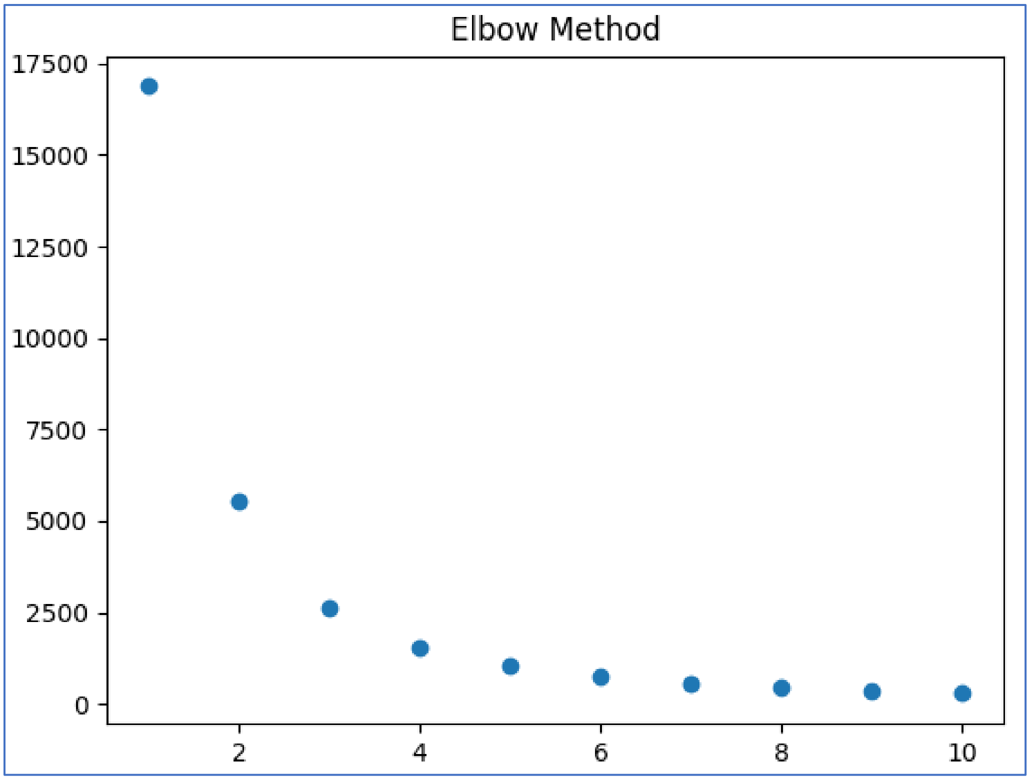
Machine learning algorithms most often take numeric feature vectors as input. Thus, when working with text documents, we convert each document into a numeric vector. This process is known as text vectorization. For instance, “This is clustering project for ML team projects” the word vector which was formed might be like <1,1,1,2,1,1,1> the two represents project since it was repeating twice. So, clusters formed might not be correlated to the twitter accounts, since similar words can be used in two twitter accounts and similar texts from two different twitter accounts would have been clustered together with the number of clusters which we gave was 16. The 16 is not an optimal solution to the clustering.

The code for the above part can be found in the link ( clustering ) for the Python code and refer to the plot files (Clustering_plot_16) for the plot which was also pasted above.

### Optional Task

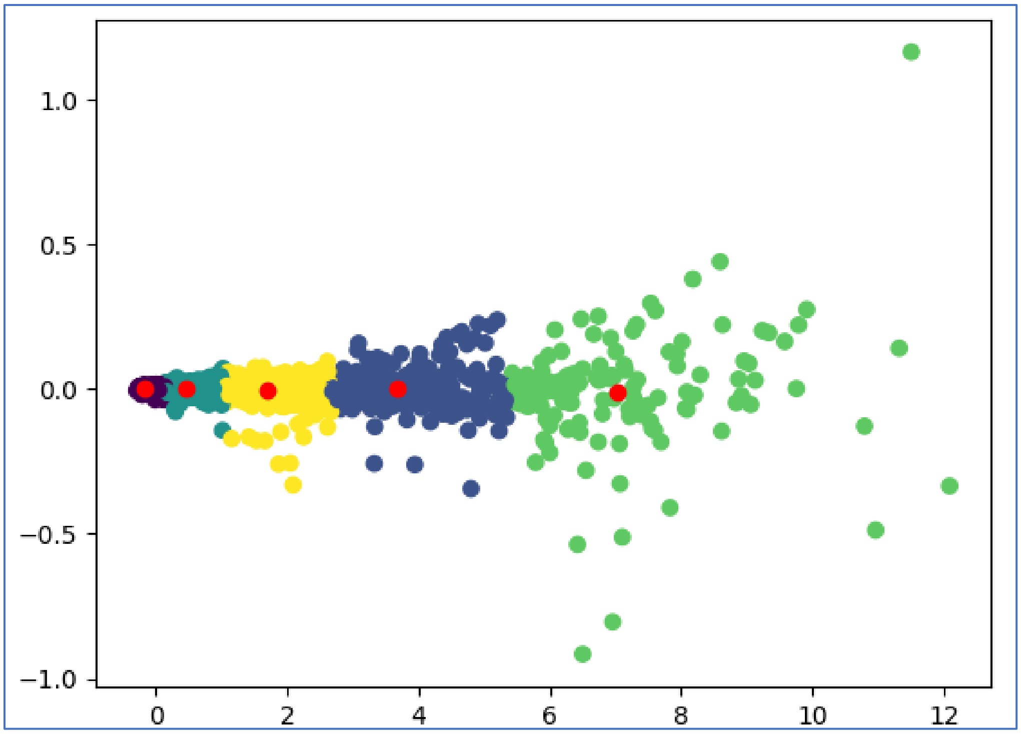
We decided to change the number of clusters parameter for the optional task as we mentioned above that the number of clusters(16) was not the optimal one, we implemented a code to get the lowest WCSS which is “Within cluster sum of squares”. The within-cluster sum of squares is a measure of the variability of the observations within each cluster. In general, a cluster that has a small sum of squares is more compact than a cluster that has a large sum of squares. Clusters that have higher values exhibit greater variability of the observations within the cluster. To find a better WCSS generally elbow method is followed where the algorithm is run for a range of clusters and WCSS is noted and plotted. The point which has lowest WCSS can be considered as the optimal cluster. Generally, after a certain low point the WCSS does not change much which can be used as optimal cluster.

Below plot shows the elbow method plot which was used to determine the cluster. The code which was used to get the plot for elbow method and implement the clustering based on the optimal number of clusters we obtained from the elbow method can be found in ( Optimal Clustering ) and the plot of elbow can be found below ( Elbow Plot ).



From the elbow method plot, the X axis is the number of clusters and the Y axis is the WCSS. The WCSS as seen does not change much for clusters greater than 5.

We implemented the Kmeans with number of clusters as 5 and the visualization for the clustering can be found ( Optimal Clustering plot ) below. This task also proves why the each twitter account does not correspond to each cluster.


1.0
0.5
0.0
−0.5
−1.0
0
2
4
6
8
10
12