VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SOCIAL INFORMATION AND NETWORKS (CSE 3021)
J-COMPONENT REPORT

CLUSTER ANALYSIS OF TWEETS

TEAM MEMBERS:
Stuti Tiwari (18BCE0950)
Rahul Anand(18BCE0953)

SUBMITTED TO:
Prof. Govinda K.
(Slot : B2)

ABSTRACT

The rapid evolution of web 2.0 technologies such as OSN applications, has led to the continuous generation of an enormous volume of digital heterogeneous data being published at an unprecedented rate. These technologies have significantly changed the way people communicate and share information among each other in various domains. Millions of people have shifted from the traditional media channels such as newspapers, to online social media. In this context, Twitter has gained massive popularity as it provides an informal platform where people can easily publish and broadcast messages on different areas across the world. It had a prominent role in spreading awareness of natural disasters such as Hurricane Sandy and socio-political events such as the Arab Spring. This has made Twitter an important source of information for synthesizing evidence in argumentation, and a goldmine of potential cross domain opportunities for both businesses and decision makers. However, the exponential amount of user generated content on this site is too vast for manual analysis. More than 500 million short-text messages, referred to as “tweets”, are published every day. This requires an automated and scalable mining process to discover patterns in the unstructured data. Cluster analysis is the unsupervised process of grouping data instances into relatively similar categories, without prior understanding of the groups structure or class labels. It is a prominent component of exploratory data analysis. A subfield of clustering includes text mining, where large volumes of text are analysed to find patterns between documents.

INTRODUCTION:

In this project, we'll cluster analyze the tweets or the words using the software RStudio. We'll take a look at how to read a text file that contains the tweets. After this, we'll be building the corpus that is the collection of the tweets where every single tweet will be treated as a single document. Once the corpus is built, we'll then make the term-document matrix from the corpus. Term-document matrix will identify each and every term in those tweets or the documents and counts how many times a single word appears in each tweet. After creating the term-document matrix, we'll also create the plot for the most frequent terms. We'll also focus on creating a dendrogram of hierarchical words/tweets clustering and the non-hierarchical k means clustering of the tweets/words.

OBJECTIVE OF THE PROJECT:

I. Read the text file.
II. Build corpus
III. Create the term-document matrix
IV. Plot frequent terms
V. Hierarchical word/tweet clustering using dendrogram
VI. non-hierarchical k means clustering of word/tweets

INNOVATIVE COMPONENT OF THE PROJECT:

We have taken a collection of tweets and performed its cluster analysis. While doing the clustering of the common words used in tweets, we created the dendrogram to check how many clusters could be formed and if the clusters were visible significantly or not. But when we plotted the dendrogram we got large numbers of clusters with very few elements in each of the clusters. Then we came up with a solution that, if we reduce the **sparsity** of the **term document matrix** and again plotted the dendrogram. We successfully grouped the words into clusters.

DATASET USED:

We have taken the dataset of the tweets from twitter and used it in our project and done the cluster analysis on it. Our dataset is based on Qantas airways which is an Australian based airlines travel booking company.

TOOLS USED:

We have used the RStudio software to get the output of all the objectives mentioned in the project and code that is used.

LITERATURE REVIEW:

1. **Author and Year :** Friedem ann , 2015
   Approach: Targeting advertisements
   Method Used: Partioning based clustering, Hard Partitioning
   Algorithm Used: K-means
   No. Of clusters: 5
   Dataset size: 10,000 Twitter user account
   Distance Measure: Euclidean distance
   Clustering Features : posted status, number of followers and account followings, latitude, longitude, whether a popular Twitter account (influencer) is followed.
   Evaluation Methods: Computing a metric of clustering quality q. The lower the value of q, the better clustering performance
   Results: Achieved clustering is midway between ideal and randomized data. Experiments emphasized the credibility of Twitter data for market analysis

2. **Author and Year :** (Soni and Mathai, 2015)
   Approach: Sentiment prediction
   Method Used: Partioning based clustering, Hard Partitioning
   Algorithm Used: K-means
   No. Of clusters: 2
   Dataset size: 1200 “Apple ” tweets

Distance Measure: Squared Euclidean distance

Clustering Features : Bag-of-Words (BOW) from twitter corpus (frequency of word occurrences

Evaluation Methods: Confusion matrix and ROC (Receiver Operator Characteristic) graph

Results: Model integration of supervised and unsupervised kMeans learning improved twitter sentiment prediction

3. **Author and Year :** (Purwita sari et al., 2015)

   Approach: News summary

   Method Used: Partioning based clustering, Hard Partitioning

   Algorithm Used: K-Medoids

   No. Of clusters: 10

   Dataset size: 200 tweets (geolocation: Indonesia)

   Distance Measure: Cosine similarity

   Clustering Features : Term frequencies and weight in tweet text. Hashtags omitted

   Evaluation Methods: The larger ASW value, the more homogeneous the cluster result

   Results: Inclusion of retweets affected cluster result quality

4. **Author and Year :** (Zhao, 2011)

   Approach: R Data Mining

   Method Used: Partioning based clustering, Hard Partitioning

   Algorithm Used: K-means, K-medoids

   No. Of clusters: 8 for K-means , 9 for K-medoids

   Dataset size: 1st 200 tweets from @rdata mining account

   Distance Measure: Euclidian distance for K-means, Manhattan distance for K-medoids

   Clustering Features : Term frequencies in tweet text (document-term matrix). Hashtags omitted

   Evaluation Methods: Checked the top 3 terms in every cluster in case of K-means and ASW in case of K-Medoids.

   Results: Clusters of different topics of K-means and Clusters overlap and not

well separated in case of K- medoids.

5. **Author and Year :** (Vicente et al., 2015)
   Approach: Gender detection
   Method Used: Fuzzy Partitioning
   Algorithm Used: K-means, c- Means
   No. Of clusters: 2,2
   Dataset size: 242,658 unique Twitter users
   Distance Measure: Euclidean distance
   Clustering Features : Screen name and user name
   Evaluation Methods: Two experiments: 1st used labelled data for building clusters and evaluating performance. 2nd used unlabeled data for clustering and labelled for evaluation
   Results: C-Means provided better clustering performance than k-Means. More usage of unlabeled data significantly enhanced cmeans but got kMeans worse

6. **Author and Year :** (Zadeh et al., 2015)
   Approach: Events and facts detection
   Method Used: Fuzzy Partitioning
   Algorithm Used: FANNY (Kaufman and Rousseeuw, 2009)
   No. Of clusters: 6
   Dataset size: 40 distinct hashtag
   Distance Measure: Manhattan Distance
   Clustering Features : Temporal aspects of hashtags
   Evaluation Methods: Defined a misfit measure to identify elements' degree of "not fitting" into a cluster. Clustering performance measured using ASW
   Results: Insights into patterns associated with each cluster for hashtags changing popularities over time

7. **Author and Year :** (Ifrim et al., 2014)
   Approach: Topic detection
   Method Used: Hierarchical- Based Clustering
   Algorithm Used: Agglomerative (dendrogram cut at 0.5)
   No. Of clusters: 5

Dataset size: 1st dataset: 1,084,200 tweets. 2nd: 943,175 JSON format English tweets
Distance Measure: Cosine similarity
Clustering Features : Date, tweet id, text, user mentions, hashtags, URLs, media URLs, and retweet or not
Evaluation Methods: (1) a subset of ground truth topics, (2) google for the automatically detected topic headline, in the manual assessment of how many detected topics are actually published news in traditional media
Results: Application of agglomerative clustering can detect topics with 80% accuracy. However, not efficient for realtime data analysis.

8. **Author and Year :** (Ramas wamy, [no date])
   Approach: Impact of distance function choice on clustering behaviour
   Method Used: Hierarchical- Based Clustering
   Algorithm Used: Two Ward (Jr., 1963) algorithms
   No. Of clusters: 5
   Dataset size: 925 tweets
   Distance Measure: Ratio of tweets appearing in different clusters / Avg. distance between tokens and clusters
   Clustering Features : Tokenization of tweets' texts
   Evaluation Methods: Several experiments conducted to determine appropriate values of confidence and support levels which determine further clustering
   Results: Generally similar behaviour of the 2 algorithms. In terms of fewer, tightly packed clusters, 2nd algorithm fared better for confidence and support values

9. **Author and Year :** (Kaur, 2015)
   Approach: Noun-based tweet categorization
   Method Used: Hierarchical- Based Clustering
   Algorithm Used: Agglomerative, Divisive
   Dataset size: 15062 "Stem Cell" tweets
   Distance Measure: Inter-cosine similarity, Intra-cosine similarity

Clustering Features : Frequency of occurrences for nouns in tweets. Hashtags omitted

Evaluation Methods: Frequency of occurrences for nouns in tweets. Hashtags omitted

Results: Combinatorial approach provided higher accuracy compared to existing methodologies, however, at the cost of performance. Clustering runtime: 1hour

10. **Author and Year :** (Miyamo to et al., 2012)

    Approach: Keyword clustering

    Method Used: Hybrid-based clustering

    Algorithm Used: Hard c-Means (partitioning), Agglomerative (hierarchical)

    No. Of clusters: 2,2

    Dataset size: 1st dataset: 50 tweets (35 terms occur > 8 times) 2nd: 50 tweets (38 terms occur > 5 times)

    Distance Measure: Squared Euclidean distance

    Clustering Features : Sequence of word occurrences in a set of tweets

    Evaluation Methods: Several observations of: clusters with and without pair-wise constraints clusters obtained by cutting the dendrogram with and without pair-wise constraints

    Results: Application of pair-wise constraints improved clustering quality. However, dataset size is arguably small

11. **Author and Year :** (Baralis et al., 2013)

    Approach: Cohesive information discovery

    Method Used: Density-based clustering

    Algorithm Used: DBSCAN

    Dataset size: “Paralymp ics” dataset: 1969 tweets / “Concert” dataset: 2960 tweets

    Distance Measure: Cosine similarity

    Clustering Features : BOW of tweets including hashtags

    Evaluation Methods: ASW

    Results: Effective in discovering knowledge. Performance relatively low for

not very large dataset. Clustering runtime: 2min 9sec May not scale well to massive datasets

12. **Author and Year :** (De Boom et al., 2015)
    Approach: Event detection
    Method Used: Density-based clustering
    Algorithm Used: DBSCAN
    Dataset size: 63,067 tweets (geolocati on: Belgium)
    Distance Measure: Sum of avg. occurrences of both hashtags per day/2
    Clustering Features : Hashtags cooccurrence matrix
    Evaluation Methods: Precision, recall, and F measures
    Results: Improvement in event detection and clustering through high-level semantic information

13. **Author and Year :** (Anumol Babu, 2016)
    Approach: Sentiment Analysis
    Method Used: Density-based clustering
    Algorithm Used: DBSCAN
    Dataset size: 100 synthetic tweets
    Distance Measure: Jaccard similarity
    Clustering Features : Tweet text and publication time. Hashtags omitted
    Evaluation Methods: Evaluating tweets segmentation and its accuracy through an experiment
    Results: Enhancement of the present system as DBSCAN was integrated

CODE USED:

```
# Read text file
library(tm)
setwd("~/Deskt
op")
tweets <- readLines("Tweets.txt")
# Build corpus
corpus <- Corpus(VectorSource(tweets))

# Create term document matrix
tdm <- TermDocumentMatrix(corpus,
                  control = list(minWordLength=c(1,Inf)))
t <- removeSparseTerms(tdm,
sparse=0.98) m <- as.matrix(t)

# Plot frequent
terms freq <-
rowSums(m)
freq <- subset(freq, freq>=50)
barplot(freq, las=2, col =
rainbow(25))

# Hierarchical word/tweet clustering using
dendrogram distance <- dist(scale(m))
print(distance, digits = 2)
hc <- hclust(distance, method =
"ward.D") plot(hc, hang=-1)
rect.hclust(hc, k=12)

# Nonhierarchical k-means clustering of
words/tweets m1 <- t(m)
set.seed(22
2) k <- 3
kc <- kmeans(m1, k)
```

WORK DONE AND IMPLEMENTATION:

1. **Read the text file from the desktop.**
   - For this, we downloaded the library TM which stands for text mining.
   - After that, we set the working directory to desktop and then using the command 'readLines' we read the text file of tweets which we store in the variable 'tweets'

2. **Then we built a corpus for the data.**
   - It included the 2141 tweets

```
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 2141
>
```

Figure 1.

Figure 1 represents the total number of tweets present in the document and each tweet is treated as a single document.

3. **Then we created a term-document matrix.**
   - We create the term-document matrix that is stored in the tdm, to which we feed the corpus that is created
   - We then use the control command to control the word length between 1 and infinity
   - Hence the term-document matrix is created.

```
<<TermDocumentMatrix (terms: 8425, documents: 2141)>>
Non-/sparse entries: 32089/18005836
Sparsity           : 100%
Maximal term length: 72
Weighting          : term frequency (tf)
>
```

Figure 2.

Figure 2 represents the term document matrix which counts the total number of terms used in the dataset by counting total number of terms in all the tweets or documents.It

gave us the output that it contained 8425 terms in 2141 tweets or documents. We find that the sparsity is really high, which means they are many tweets with the cell as zero hence we reduce the sparsity.

4. **We found that the sparsity of the matrix was high, so we reduced the sparsity.**

```
<<TermDocumentMatrix (terms: 83, documents: 2141)>>
Non-/sparse entries: 11741/165962
Sparsity           : 93%
Maximal term length: 15
Weighting          : term frequency (tf)
>
```

Figure 3.

Figure 3 represents that Using the command removeSparseTerms we specify the tdm to have sparse value as say 0.98. The basic concept is that most entries in a tdm are empty, meaning that most terms do not appear in most of the documents and there are Lots and lots of zeros in the matrix. Typically, 90% or more are zeros in a large corpus. If you set the threshold value at, say, 98%, the tm package drops enough terms that are very infrequent -- which drove up the sparseness percentage -- so that the resulting less-sparse set of terms has only a measure of 93%.
Hence the sparsity is reduced to 93% with the number of terms as 83.

5. **Then we converted the sparse terms into a matrix to do further analysis.**

6. **The sparse term-document matrix is stored in variable m**

```
> m
                Docs
Terms            1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
                Docs
Terms            25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
                Docs
Terms            46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
                Docs
Terms            67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87
                Docs
Terms            88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106
                Docs
Terms            107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122
                Docs
Terms            123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138
```

Figure 4.

Figure 4 represents the term-document matrix created so as to perform further analysis. As there are many terms, plotting the graph of each term will not be possible, hence we set up the frequency of terms greater than 50 so as to plot only the most frequent terms.

7. **We plotted the frequent terms in the tweets.**
    - We'll calculate the sum of the rows using the command rowSums and store it in freq variable.
    - We give a condition of plotting the terms whose frequency is greater than or equal to 50.
    - Once done this we make a bar plot with different terms on the x-axis and it frequency on the y-axis.

BAR- GRAPH

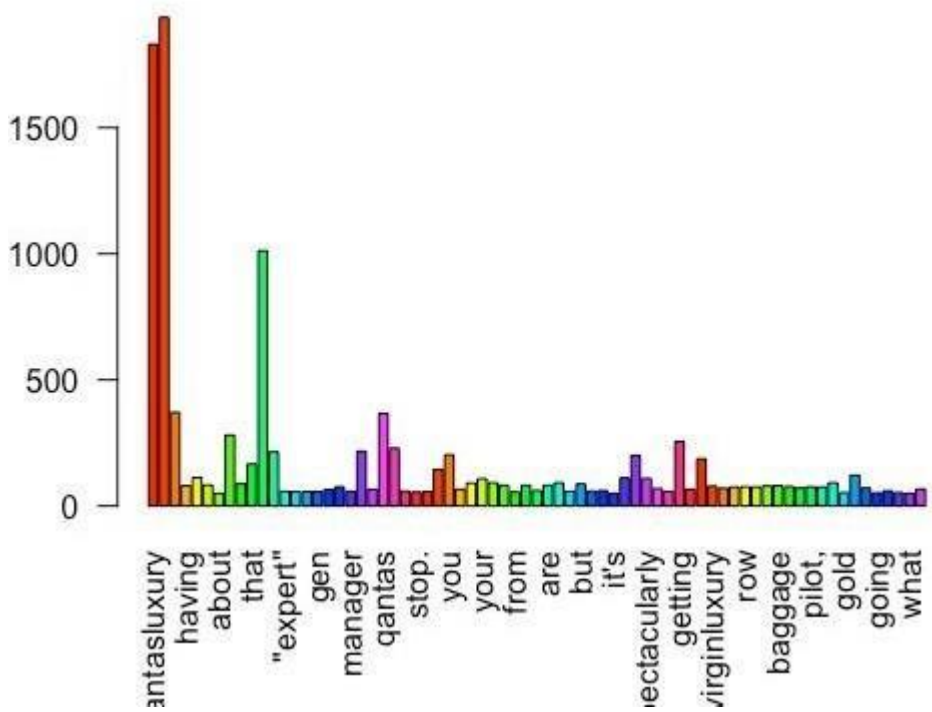

Figure 5.

Figure 5 represents the bar plot created using the matrix in figure 4. The x-axis represents the most frequent terms and the y-axis represents the frequency of each term. The higher the frequency of term, higher is the graph's height.

8. **Then we did the hierarchical clustering of the tweets using dendrogram.** We then create the dendrogram using the dist function to calculate the distance between the words and print the distance between the words present in each document.

   - We find that if the distance between the words is high then those words are not in the same cluster and if the distance is low, then they are in the same cluster.

   - After calculating the distances, we do the hierarchical clustering using hclust function and we store this in a variable hc
   - Once running the hc we'll plot the cluster dendrogram with hang=- 1 which will give the output showing the clusters.
   - Rectangles around the clusters are shown using the command rect.hclust

- As we increase the k value in the command rect.hclust(hc, k=12), we can see more clusters in the dendrogram.

Distance Matrix

	time	2011	and	having	not	qantasluxury
2011	181.8					
and	81.4	176.8				
having	44.6	180.7	80.9			
not	49.0	179.9	85.0	47.0		
qantasluxury	182.9	63.4	171.9	181.2	180.3	
downfall	56.3	177.7	88.0	58.6	60.5	178.0
had	46.3	180.1	85.2	49.0	53.3	180.7
http	112.5	156.5	128.4	113.8	111.2	157.3
about	38.4	181.9	79.0	41.7	46.9	182.3
disaster	37.2	181.4	81.8	43.9	48.2	182.0

Figure 5.1

Figure 5.1 represents the distance between the words in all the documents. If the distance between the words is high then it means that the words are present in different documents/tweets. For e.g., time and http in the above picture. Similarly, if the distance is low means they are in same document, for e.g., about and having. The given matrix is used for hierarchical clustering of tweets.

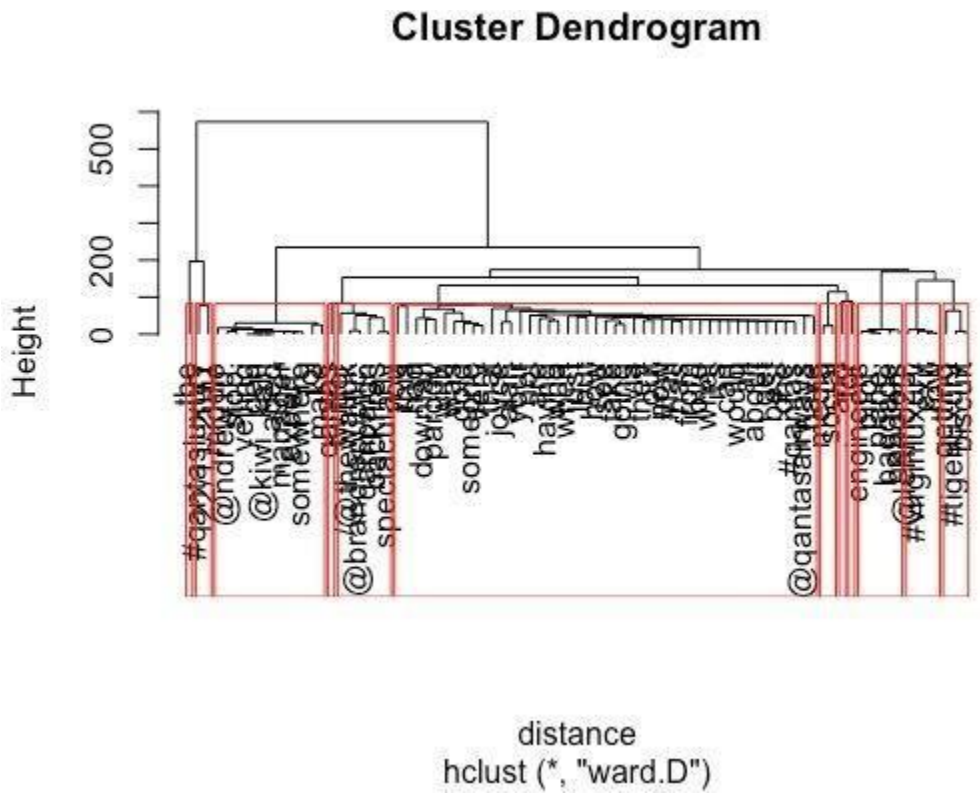

Figure 6.

Figure 6 represents the cluster dendrogram created using the hierarchical clustering of tweets. Here the words are there on x axis and the heights on y-axis. Difference between the bar heights of words represents the distance between the words. Higher the distance words belong to different clusters. And if the distance is low, the words belong to same cluster. The clusters formed in above case, are highlighted via red rectangles. So there, are total 12 clusters.

9. **Then we finally did the non-hierarchical k- means clustering of the tweets.** We create the transpose of the matrix m as m1 and fix a seed, k=12 as 12 clusters and do the k-means clustering using the command kmeans(m1, k)
    - The output gives the size of clusters and cluster means as shown

```
K-means clustering with 12 clusters of sizes 220, 239, 167, 90, 336, 543, 44, 110, 121,
 51, 145, 75

Cluster means:
   #qantasluxury 22/11/2011        and     having        not        had
1      0.3409091  0.3409091 0.33636364 0.00000000 0.00000000 0.00000000
2      0.9414226  0.9958159 0.12970711 0.05020921 0.05020921 0.05439331
3      0.9940120  1.0000000 0.00000000 0.00000000 0.01197605 0.00000000
4      0.9888889  1.0000000 0.12222222 0.00000000 0.10000000 0.00000000
5      0.9255952  0.9791667 0.08928571 0.03571429 0.01785714 0.11011905
6      0.8342541  0.9171271 0.00000000 0.03867403 0.05340700 0.05340700
7      0.9772727  1.0000000 0.00000000 0.00000000 0.47727273 0.00000000
8      0.9363636  1.0000000 0.08181818 0.05454545 0.07272727 0.00000000
9      0.9586777  0.9834711 0.06611570 0.01652893 0.09917355 0.01652893
10     1.0000000  1.0000000 0.03921569 0.01960784 0.01960784 0.00000000
```

Figure 7.

- **Figure 7** represents the output of Non- hierarchical K-means clustering of the given tweets. The above figure gives the size of each cluster and cluster-means of each word in each cluster. If the cluster means is high the frequency of that word in the respective cluster is high, else it is low. For e.g, frequency of #qantasluxury is high in cluster 10 whereas it is low in cluster 1.

  - The output also gives the averages of each word that we are analyzing in every cluster.
  - High average means that particular word has appeared with high frequency in that cluster.
  - The clustering vector signifies the cluster number where the word has gone into.

```
Clustering vector:
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
  6  11   6  11   2   3   5   3   5   8   6  12   6   5   5  11
 17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32
  6   6  11   6   6   6   2  11   6  11   6   6   6   6   6   6
 33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48
  3  10   6   5   3   8   5   9   3   4   2   1   1   1   6   6
 49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64
  1   1   1   6  11   3   3  11   6   6   5   5   6   6   2   9
 65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
  6  12   4   4   3   6   5   6   7   3   1   1   1   9   4  12
```

Figure 8.

Figure 8 represents the entries of each of the most frequent 83 terms and the cluster in which they appear maximum times. For e.g., term1 appears in cluster 6 and so on.

- The picture below shows the within sum of square of all the 12 cluster

```
Within cluster sum of squares by cluster:
[1] 8352.1413994    0.8571429   31.1052632
 (between_SS / total_SS =   7.0 %)

Available components:

[1] "cluster"     "centers"     "totss"       "withinss"    "tot.withinss"
[6] "betweenss"   "size"        "iter"        "ifault"
>
```

Figure 9.

Figure 9 tells that if Within cluster sum of squares value is low, the variability within the cluster is low and elements are closer to each other. And the between_SS/ total_SS value is high then the cluster to cluster difference is high.

10. We calculate the sum of squares divided by the total sum of squares which has the output as 7(k=3)

11. The lower number of 'sum of squares' signifies that elements in the cluster are closer to each other.

12. The higher number of total 'sum of squares' signifies that the cluster to cluster differences or distances are higher.

CONCLUSION:

When we opened the tweets file tweets.txt, we showed that there were 2141 tweets and all tweets were shown in the console. After which we built the corpus which was stored in the function corpus itself to make corpus ready with 2141 tweets.
Term-document matrix contained 8425 terms in 2141 tweets/documents. Since many terms were with the value, we remove sparsity to 93 percent and the number of terms is 83. We made a barplot with different terms on the x-axis and frequency on the y-axis. We then made cluster dendrograms where clusters were grouped calculating the distances between the words into red rectangle boxes.
We calculated the non-hierarchical clustering of k means clustering, the sum of squares divided by the total sum of squares is 7, when no of clusters k=3.

REFERENCES

14.https://en.wikipedia.org/wiki/Cluster_analysis

15.https://www.statisticssolutions.com/directory-of-statistical-analyses-cluster-analysis/

16.https://www.techopedia.com/definition/30391/cluster-analysis

17.https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning- 6a6e67336aa1

18.https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm

19.https://www.geeksforgeeks.org/k-means-clustering-introduction/

20. Aggarwal, C. C. & Zhai, C. 2012. Mining Text Data, Springer Science & Business Media.

21. Anumol Babu, R. V. P. 2016. Efficient Density Based Clustering of Tweets and Sentimental Analysis Based on Segmentation. International Journal of Computer Techniques, 3, 53-57.

22. Baralis, E., Cerquitelli, T., Chiusano, S., Grimaudo, L. & Xiao, X. Analysis of Twitter Data Using A MultipleLevel Clustering Strategy. International Conference on Model and Data Engineering, 2013. Springer, 13-24.

23. Bora, D. J., Gupta, D. & Kumar, A. 2014. A Comparative Study Between Fuzzy Clustering Algorithm and Hard Clustering Algorithm. Arxiv Preprint Arxiv:1404.6059

24. Breiman, L. 2001. Random Forests. Machine Learning, 45, 5-32.

25. Castillo, C., Mendoza, M. & Poblete, B. Information Credibility On Twitter. Proceedings Of The 20th International Conference On World Wide Web, 2011. ACM, 675-684.

26. De Boom, C., Van Canneyt, S. & Dhoedt, B. SemanticsDriven Event Clustering In

Twitter Feeds. Making Sense Of Microposts, 2015. Ceur, 2-9.

27. Ester, M., Kriegel, H.-P., Sander, J. & Xu, X. A DensityBased Algorithm for Discovering Clusters in Large Spatial Databases with Noise. KDD, 1996. 226-231.

28. Friedemann, V. 2015. Clustering A Customer Base Using Twitter Data.

29. Go, A., Bhayani, R. & Huang, L. 2009. Twitter Sentiment Classification Using Distant Supervision. Cs224n Project Report, Stanford, 1, 12.

30. Godfrey, D., Johns, C., Meyer, C., Race, S. & Sadek, C. 2014. A Case Study in Text Mining: Interpreting Twitter Data from World Cup Tweets. Arxiv Preprint Arxiv:1408.5427.

31. Han, J., Pei, J. & Kamber, M. 2011. Data Mining: Concepts And Techniques, Elsevier.

32. Ifrim, G., Shi, B. & Brigadir, I. Event Detection in Twitter Using Aggressive Filtering and Hierarchical Tweet Clustering. Second Workshop on Social News on the Web (Snow), Seoul, Korea, 8 April 2014, 2014. ACM.

33. Jr., J. H. W. 1963. Hierarchical Grouping to Optimize an Objective Function. Journal of the American Statistical Association, 58, 236-244.

34. Kaufman, L. & Rousseeuw, P. J. 2009. Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley & Sons.

35. Kaur, N., 2015. A Combinatorial Tweet Clustering Methodology Utilizing Inter and Intra Cosine Similarity (Doctoral Dissertation, Faculty Of Graduate Studies And Research, University Of Regina).

36. Krestel, R., Werkmeister, T., Wiradarma, T. P. & Kasneci, G. Tweet-Recommender: Finding Relevant Tweets for News Articles. Proceedings of the 24th International Conference on World Wide Web, 2015. ACM, 53-54.

37. Kumar, S., Morstatter, F. & Liu, H. 2014. Twitter Data Analytics, Springer.