TWITTER TEXT MINING USING CLUSTERED SEARCH

Anjali Malav (UG201310006)
Muttineni Navya (UG201310020)

I. Abstract

Text clustering is the application of cluster analysis to textual documents. It has applications in automatic document organization, topic extraction and fast information retrieval or filtering. A search query often returns thousands of results in response to a broad query, making it difficult for users to browse or to identify relevant information. Clustering methods can be used to automatically group the retrieved documents into a list of meaningful categories. In this project, twitter tweets are used as a database to search across with the aid of clustering them with kmeans and k-medoids and returning results divided according to content relevance.

II. Introduction

A. Overview and Problem Statement

Overview

With advent of web systems and Internet technologies, we can get a large amount of information. Especially, we can find a lot of research documents from the world via search engines such as Google, Yahoo and Bing.

Twitter is one of the information source on the web. Twitter is a major microblogging service. The users send text-based message of up to 140 characters, photograph and video.When searched these tweets with a search term, thousands of tweets are displayed. The user has to go through every tweet in order to see if that tweet is of his relevance or not. This burden would be reduced if the tweets were clustered first and then returned to the user topic wise who would then have to only go through the those tweets belonging to his topic of interest. So it is desirable to have good techniques to summarize large no of tweets. One intersting problem in twitter analysis is automatic detection of topics being discussed in tweets. The task of topic extraction can be achieved by clustering a group of news items, blogs, or tweets.

Existing search engines do an excellent and convenient job. They organize up to billions of documents which can be searched quickly for keywords and the plain keyword search forms the starting point for the majority of users. However, while this strategy works fine for the experienced human information miner, the typical user is faced either with an empty result list or with a list containing thousands of hits. The former situation is the result of misspelling or contradictory Boolean query formulation; it can be addressed with a syntactic analysis. The latter situation lacks a meaningful specification of context - it requires a semantic analysis, which can be provided by means of category narrowing.

So in this project we start with extracting text from twitter and follow the steps as given in the following diagram to reach the clustered output.

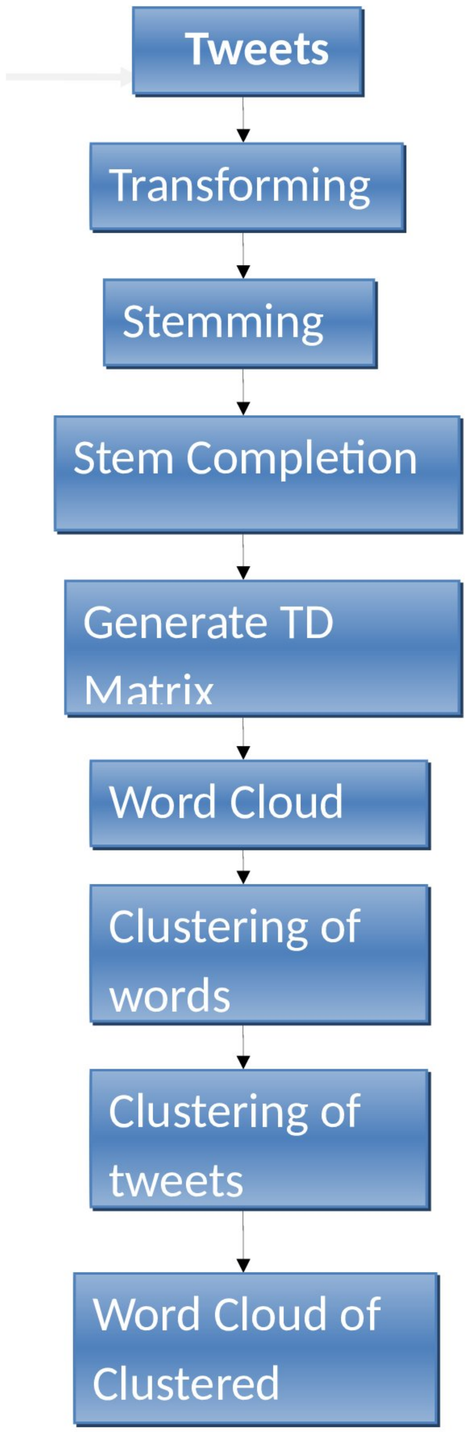


Fig. 1: Timeline

Problem Statement

Given a search word, the objective is to return results in this case tweets in clustered manner from a large database of tweets.

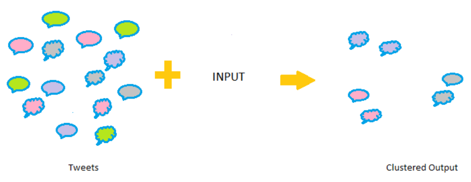


Fig. 2: Problem Statement

Motivation

Popular online social media sites like Facebook or Twitter allow users to post short message to their homepage. These are often referred as micro-blogging sites and the message is called a status update. Status updates from Twitter are more commonly called as tweets. Tweets are often related to some event, specific topic of interest like music, dance or personal thoughts and opinions. A tweet can contain text, emoticon, link or their combination.

Tweets have recently gained a lot of importance due to their ability to disseminate information rapidly. Popular search engines like Google and Bing have started including feeds from Twitter in their search results. Researchers are actively involved in analyzing these micro-blogging systems. Some research areas include understanding usage and communities, discovering user characteristics, detecting spam and so on.

We believe that clustering of tweets will help to easily categorize them based on their content. Using such clusters we would be able to identify the topic or particular event about which the tweet is.

B. Brief Literature Survey

Clustering

Clustering is an unsupervised learning techniques that takes a collection of objects such as tweets and organizes them into groups based on their similarity. The groups that are formed are known as clusters. We focus on hierarchical clustering. The two main types of which are:

- **Agglomerative (bottom-up):** Agglomerative algorithms begin with each individual document as a separate cluster, each of size one. At each level the smaller clusters are merged to form a larger cluster. It proceeds this way until all the clusters are merged into a single cluster that contains all the documents.
- **Divisive (top-down):** Divisive algorithms begin with the entire set and then the splits are performed to generate successive smaller clusters. It proceeds recursively until individual documents are reached.

The agglomerative algorithms are more frequently used in information retrieval than the divisive algorithms. The splits and merge are generally done using a greedy algorithm. A greedy algorithm is an algorithmic approach that makes the locally optimal choice at each stage of its run with the hope of finding the global optimum. The results are often represented using a dendrogram

Text Categorization

Text categorization (a.k.a. text classification) is the task of assigning predefined categories to free-text documents. It can provide conceptual views of document collections and has important applications in the real world. For example, news stories are typically organized by subject categories (topics) or geographical codes; academic papers are often classified by technical domains and sub-domains; patient reports in health-care organizations are often indexed from multiple aspects, using taxonomies of disease categories, types of surgical procedures, insurance reimbursement codes and so on. Another widespread application of text categorization is spam filtering, where email messages are classified into the two categories of spam and non-spam, respectively.

Use of Wikipedia concepts to determine closeness between texts was explained in (Gabrilovich and Markovitch 2007). Text categorization based on word clustering algorithms was described in (Bekkerman et al. 2003). k-means clustering for sparse data was introduced in (Dhillon and Modha 2001).

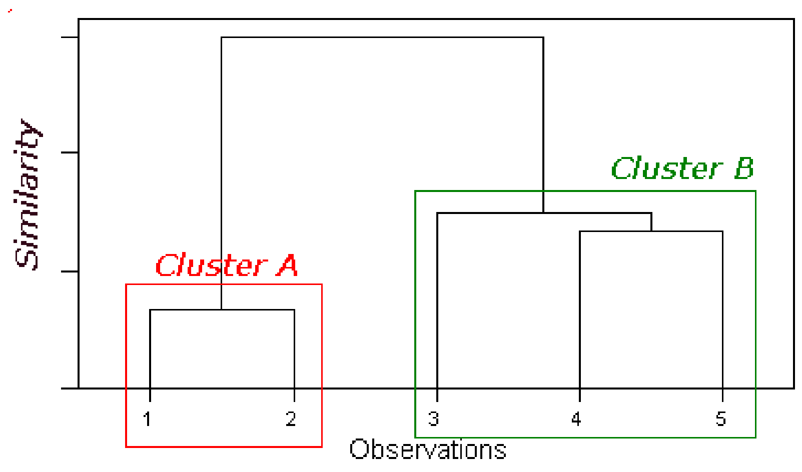


Fig. 3: Dendrogram depicting clusters

Twitter

Twitter is a favorite source of text data for analysis: its popular (there is a huge volume of variety on all topics) and easily accessible using Twitters free, open APIs which are easily consumable in JSON and ATOM formats.

III. Medhodology

The steps begin with extracting text from Twitter. The extracted text is then transformed to build a document-term matrix. After that, frequent words and associations are found from the matrix. A word cloud is used to present important words in documents. In the end, words and tweets are clustered to find groups of words and also groups of tweets.

1) Retrieving Text from Twitter
We collected around 200 tweets with our search term being ”rdatamining”. These tweets were then converted into a Data Frame and then to a Corpus which is a collection of documents containing text was created. In our case every tweet is a document.
2) Transforming Text
The following transformations were applied on the Corpus:
Converting to Lower case:
Since the case of the letters doesn’t change the meaning of the word they make up, words should be considered irrespective of the case.
Example: data and Data should be considered to be same.
Removing URL, numbers and punctuations:
All he URLs, i.e. text of the form ”http[[̂:space:]]*” were removed from the Corpus.
The Corpus was even made free of the numbers and punctuations. Since removing punctuations might concatenate few words (memory,mind –¿ memorymind) the punctuations are replaced by space character.
Removing stopwords:
Stopwords are common words that carry less important meaning than keywords. They don’t have any specific meaning and so don’t contribute towards searches and only increase the computing overload.
Examples: hence, actually, take, hesrelf etc.

3) Stemming Words
Words are required to be stemmed to get their radicals, so that the various forms of words which are derived from a stem would be considered to be the same when calculating word frequency.
Example: update, updated and updating would all be stemmed to updat” since all three mean the same.
After stemming data, the data also needs to be completed so that it looks normal.
Example: updat is completed to update.
The following steps were followed in stemming and stemcompletion:
1. A copy of the corpus obtained so far after the transformation is made which would serve as dictionary while stem completion.
2.Then the corpus was stemmed and stem completion was applied on it. The copy of the corpus made in the previous

step was used to complete the words in this step.

4) Building a Term-Document Matrix
Term Document/ Document Term Matrix is a matrix with the rows depicting the terms/words in all the documents/tweets and the columns depicting the document number and it is vice versa in the case of Document-Term matrix. The entries in this matrix are 1/0. 1 if that word is present in that tweet else 0.
The term document matrix for the text in the Corpus so far obtained after transforming and stemming data is generated.

5) Data visualization with Bar Graph and Word Cloud Generation
To visualize the popular words and the association between them, the frequency of all these words was calculated by summing up the values in a row and the terms with frequency greater than 10 were plotted on a Bar graph with x-axis being terms and y-axis being frequency.
Then the frequencies calculated were sorted in decreasing order and a word cloud was plotted with the most frequent terms in the middle.

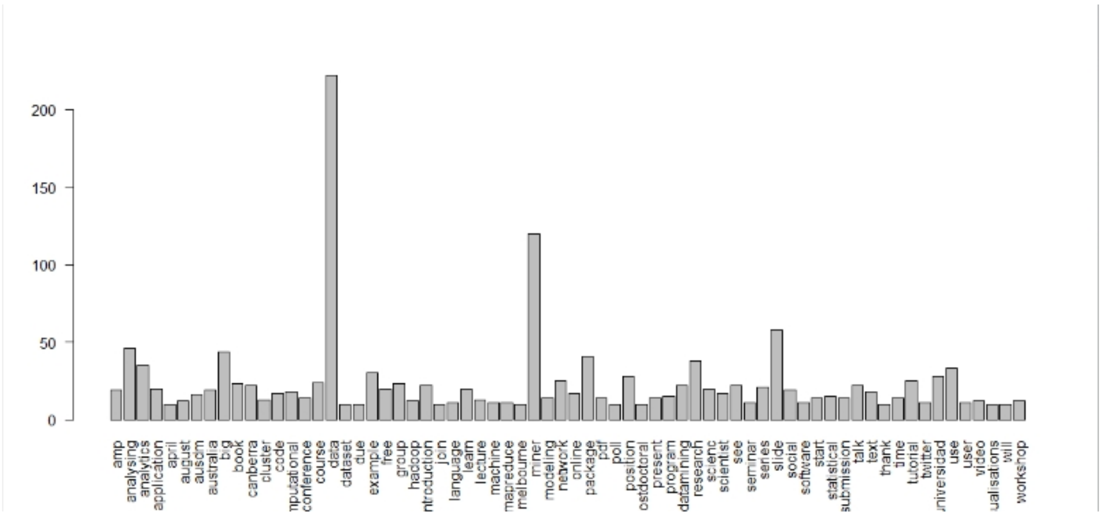


Fig. 4: Bar plot

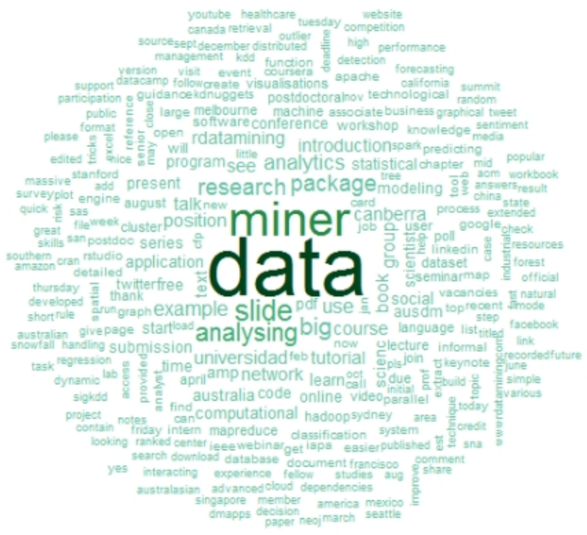


Fig. 5: Word cloud of all the words

6) Clustering Words
Clustering the words with maximum sparsity 0.95 was done in the following way:
We perform centering and scaling on the TDM. Centering is done by subtracting the column means of TDM from their corresponding columns.
Scaling is obtained by dividing the columns of centered TDM by their standard deviations.
Distance Matrix using Euclidean method was computed for the term document matrix obtained so far.
Hierarchical clustering is performed on this distance matrix generated. On an observation set, I, dene a dissimilarity measure. Set each of the observations, i,j,k, etc.$\in$ I to be a singleton cluster. Agglomerate the closest (i.e. leastdissimilar) pair of clusters, deleting the agglomerands, or agglomerated clusters. Redene the inter-cluster dissimilarities with respect to the newly created cluster. If n is the cardinality of observationset I then this agglomerative hierarchical clustering algorithm completes in n1 agglomerative steps. Through use of the Lance-Williams update formula, dissimilarities relative to a newly created cluster are updated.
Lance-William Formula:
Δ(i $\bigcup$ i',i") =)= (wi + w" / wi + wi' + wi").δ(i,i") + (w' i + w" / wi + wi' + wi").δ(i',i") (wi"/ wi + wi' + wi").δ(i,i')
wi: Cluster cardinality of set i.
δ(i, i'): Dissimilarity between cluster i and i'.
Then plotted a dendogram to visualize these clusters.

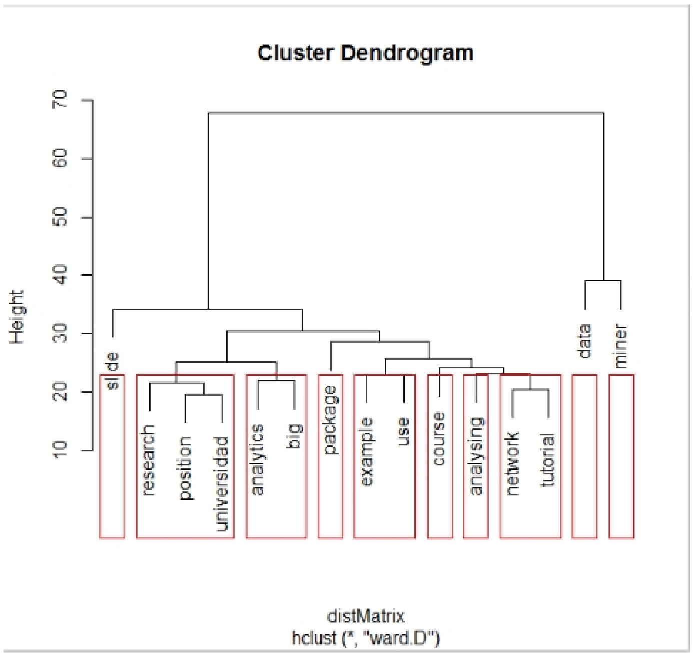


Fig. 6: Dendrogram depicting clusters

7) Clustering Tweets
Clustered tweets using the TDM's transpose i.e. Document Term Matrix. Two algorithms were used to cluster tweets, k-means and k-medoids.

k-means:
The data is clustered by the k-means method, which aims to partition the points into k groups such that the sum of squares from points to the assigned cluster centres is minimized. At the minimum, all cluster centres are at the mean of their Voronoi sets (the set of data points which are nearest to the cluster centre).
The Lloyd's algorithm, mostly known as k-means algorithm, is used to solve the k-means clustering problem and works as follows.
1.Decide the number of cluster:
The number of clusters should match the data. An incorrect choice of the number of clusters will invalidate the whole process. In k-means we set the number of clusters to 8 (chosen randomly).
2. Initialize the center of the clusters:

$$\mu_i = some\ value, i = 1....k$$

The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other.
3.Attribute the closest cluster to each data point:
The next step is to take each point belonging to a given data set and associate it to the nearest center.
4.Recalculation of centroids:
Recalculates the centroids as the average of all data points in a cluster
5.Assigning data points:
Assigns data points to their closest centroid.
Repeat step 4,5 until the sum of squares from points to the assigned cluster centers is minimized.

$$min \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i^j - \mu_j||^2$$

where $||x_i^j - \mu_j||^2$ is a chosen distance measure between a data point x_i^j and the cluster center μ_j , is an indicator of the distance of the n data points from their respective cluster centers.

k-medoids:
Dissimilarity matrix is calculated fom the Document Term Matrix using Manhattan distance metric. Then the following two steps are followed. The number of clusters is estimated by optimum average silhouette width method.
1. Build
Constructing medoids initially:
m1 is the object with the smallest

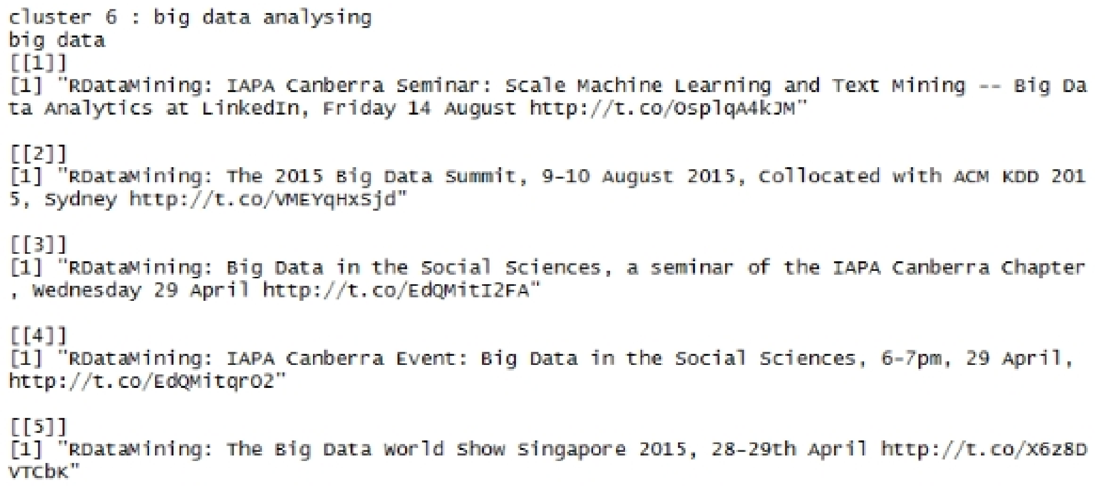
$$\sum_{i=1}^{n} d(i, m1)$$

m2 minimizes the objective function
.
.
.
mk minimizes the objective function
objective:

$$\sum_{i=1}^{n} min_{t=1,...,k} d(i, m_t)$$

where $d(i, mt)$ is the dissimilarity between i and medoid mt.
Then allocate all the objects to these medoids depending on the least dissimilarity between them.

2. Swap:
Consider all pairs of objects (i,j) with
i∈{m1,...,mk} and j !∈{m1,...,mk} and perform the i and j swap if the objective function is minimized further because of that swap.
If any medoid changed due to the swapping, then we need to allocate the objects to these medoids again and follow with the Swap step.
This needs to be repeated until there is no change in the medoids in swap step.
Then finally once the swap converges, we have our clusters.
The top 3 popular words from every cluster are being displayed so as to get an idea about what that cluster is about.
For a better understanding of the cluster topic, wordcloud is also plotted for every cluster so that even the remaining important words are also plotted which would help the user better decide the topic of that cluster.



```
cluster 6 : big data analysing
big data
[[1]]
[1] "RDataMining: IAPA Canberra Seminar: Scale Machine Learning and Text Mining -- Big Da
ta Analytics at LinkedIn, Friday 14 August http://t.co/Osp1qA4kJM"

[[2]]
[1] "RDataMining: The 2015 Big Data Summit, 9-10 August 2015, Collocated with ACM KDD 201
5, Sydney http://t.co/VMEYqHxSjd"

[[3]]
[1] "RDataMining: Big Data in the Social Sciences, a seminar of the IAPA Canberra Chapter
, Wednesday 29 April http://t.co/EdQMitI2FA"

[[4]]
[1] "RDataMining: IAPA Canberra Event: Big Data in the Social Sciences, 6-7pm, 29 April,
http://t.co/EdQMitqrO2"

[[5]]
[1] "RDataMining: The Big Data World Show Singapore 2015, 28-29th April http://t.co/X6z8D
VTCbK"
```

Fig. 7: clusters

And the word cloud for this tweet is:

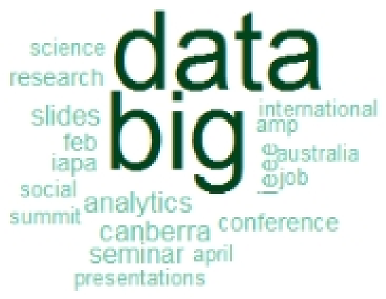


Fig. 8: word cloud

IV. Experiments and Results

We ran our code on many search terms like comp_science, rdatamining, timesofindia etc.
Here we are showing the results of the experiment of clustering tweets belonging to two search terms rdatamining and compscience.

rdatamining:
Word cloud of all the tweets:

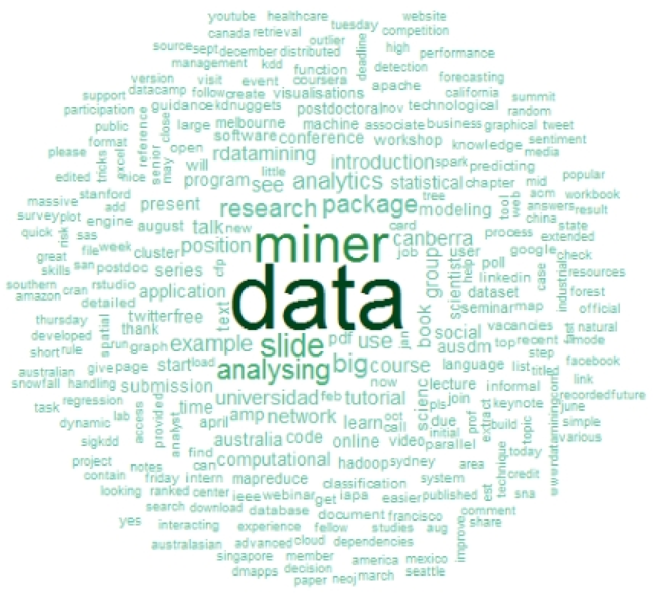


Fig. 9: Word Cloud of all words

Dendogram of the clustered words:

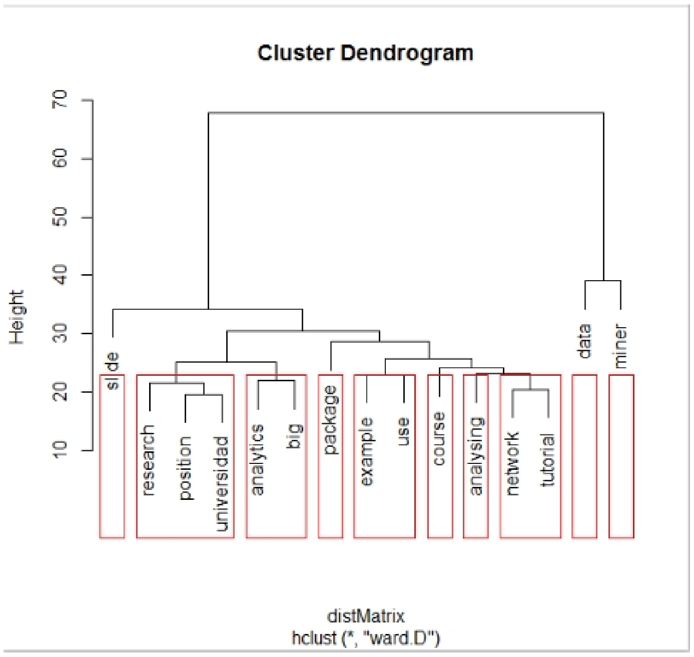
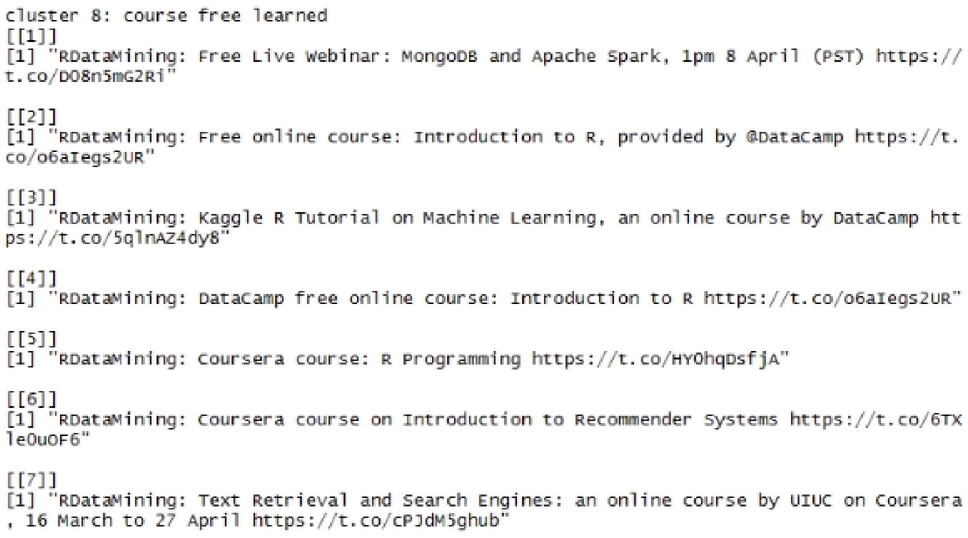


Fig. 10: Dendrogram depicting clusters

Above is the dendogram plotted on the clustering of words from all the obtained tweets.

tweets clusters and their corresponding word clouds:



```
cluster 8: course free learned
[[1]]
[1] "RDataMining: Free Live Webinar: MongoDB and Apache Spark, 1pm 8 April (PST) https://
t.co/DO8n5mG2Ri"

[[2]]
[1] "RDataMining: Free online course: Introduction to R, provided by @DataCamp https://t.
co/o6aIegs2UR"

[[3]]
[1] "RDataMining: Kaggle R Tutorial on Machine Learning, an online course by DataCamp htt
ps://t.co/5qlnAZ4dy8"

[[4]]
[1] "RDataMining: DataCamp free online course: Introduction to R https://t.co/o6aIegs2UR"

[[5]]
[1] "RDataMining: Coursera course: R Programming https://t.co/HYOhqDsfjA"

[[6]]
[1] "RDataMining: Coursera course on Introduction to Recommender Systems https://t.co/6TX
le0uOF6"

[[7]]
[1] "RDataMining: Text Retrieval and Search Engines: an online course by UIUC on Coursera
, 16 March to 27 April https://t.co/cPJdM5ghub"
```

Fig. 11: cluster 8

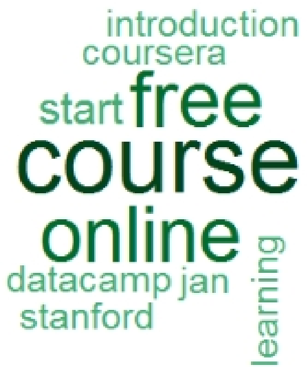
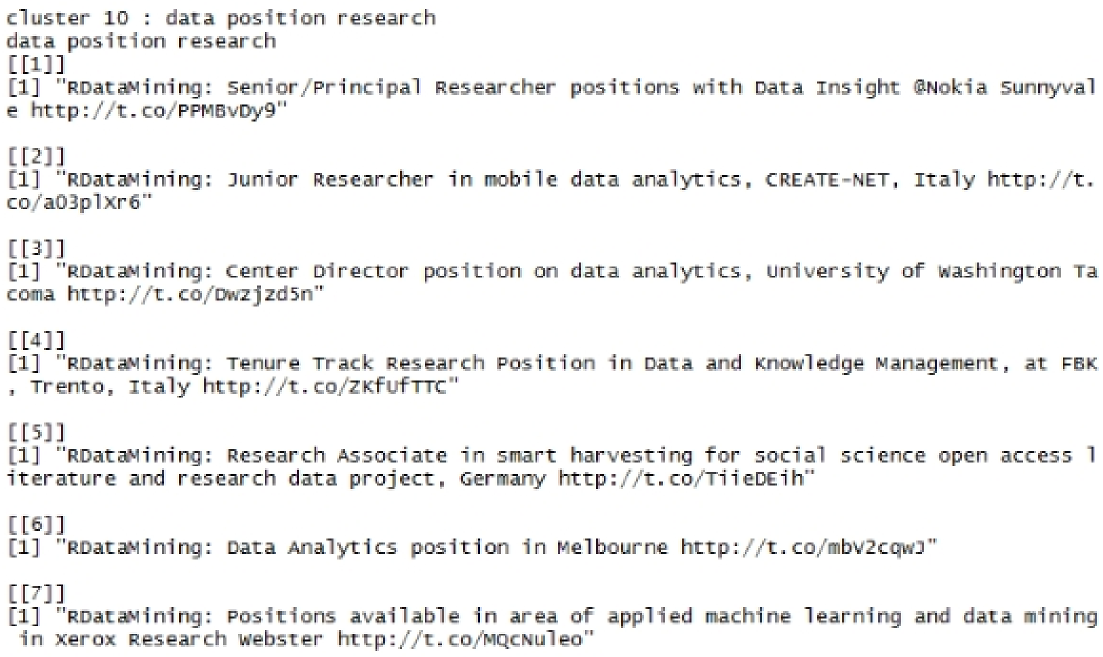


Fig. 12: cluster 8 word cloud



```
cluster 10 : data position research
data position research
[[1]]
[1] "RDataMining: Senior/Principal Researcher positions with Data Insight @Nokia Sunnyval
e http://t.co/PPMBvDy9"

[[2]]
[1] "RDataMining: Junior Researcher in mobile data analytics, CREATE-NET, Italy http://t.
co/aO3plXr6"

[[3]]
[1] "RDataMining: Center Director position on data analytics, University of Washington Ta
coma http://t.co/Dwzjzd5n"

[[4]]
[1] "RDataMining: Tenure Track Research Position in Data and Knowledge Management, at FBK
, Trento, Italy http://t.co/ZKfUfTTC"

[[5]]
[1] "RDataMining: Research Associate in smart harvesting for social science open access l
iterature and research data project, Germany http://t.co/TiieDEih"

[[6]]
[1] "RDataMining: Data Analytics position in Melbourne http://t.co/mbV2cqwJ"

[[7]]
[1] "RDataMining: Positions available in area of applied machine learning and data mining
 in Xerox Research Webster http://t.co/MQcNuleo"
```

Fig. 13: Cluster 10

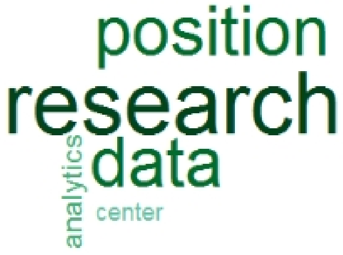


Fig. 14: Cluster 10 word cloud

.

compscience:

Dendogram of the clustered words:

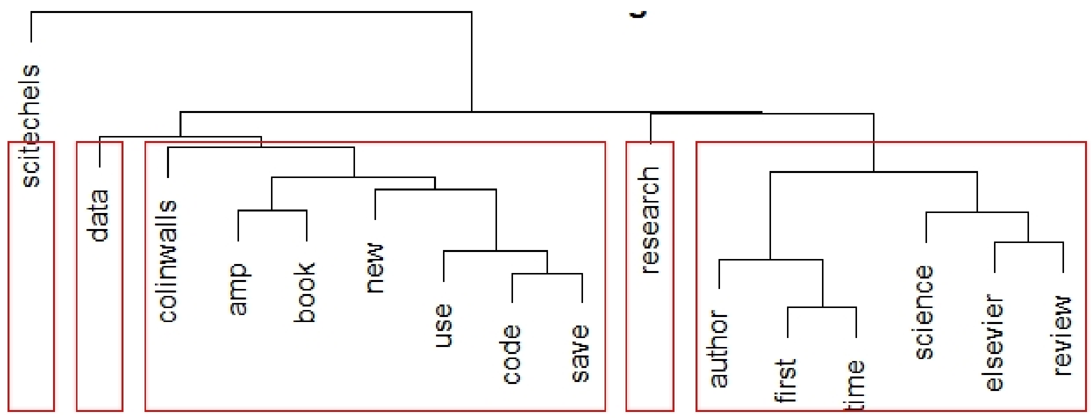
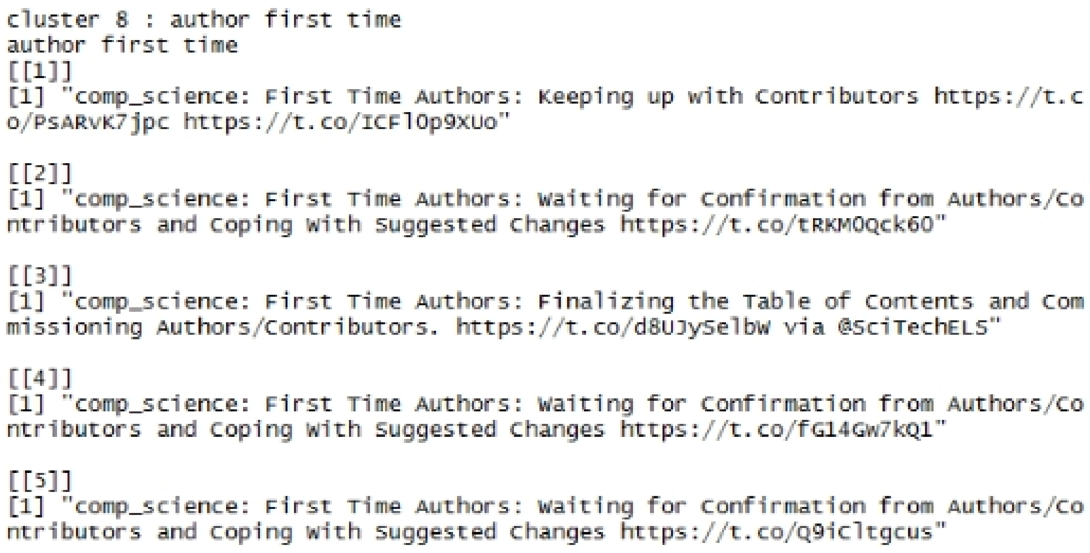


Fig. 15: Bar plot

tweet clusters and their corresponding word clouds:



```
cluster 8 : author first time
author first time
[[1]]
[1] "comp_science: First Time Authors: Keeping up with Contributors https://t.c
o/PsARvK7jpc https://t.co/ICFl0p9XUo"

[[2]]
[1] "comp_science: First Time Authors: Waiting for Confirmation from Authors/Co
ntributors and Coping with Suggested Changes https://t.co/tRKM0Qck6O"

[[3]]
[1] "comp_science: First Time Authors: Finalizing the Table of Contents and Com
missioning Authors/Contributors. https://t.co/d8UJySelbW via @SciTechELS"

[[4]]
[1] "comp_science: First Time Authors: Waiting for Confirmation from Authors/Co
ntributors and Coping with Suggested Changes https://t.co/fG14Gw7kQ1"

[[5]]
[1] "comp_science: First Time Authors: Waiting for Confirmation from Authors/Co
ntributors and Coping with Suggested Changes https://t.co/Q9icltgcus"
```

Fig. 16: Cluster 8

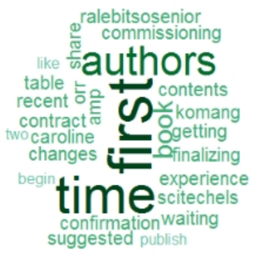
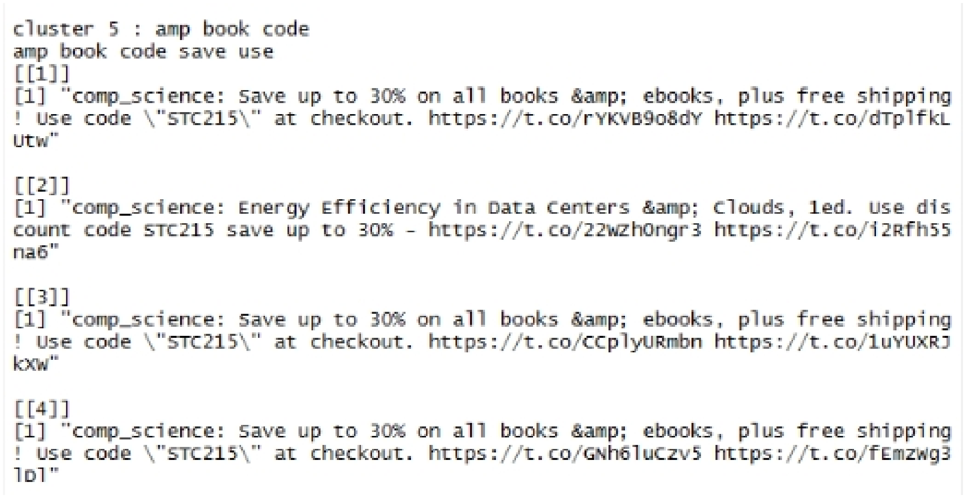


Fig. 17: cluster 8 word cloud



```
cluster 5 : amp book code
amp book code save use
[[1]]
[1] "comp_science: Save up to 30% on all books &amp; ebooks, plus free shipping
! Use code \"STC215\" at checkout. https://t.co/rYKVB9o8dY https://t.co/dTplfkL
Utw"

[[2]]
[1] "comp_science: Energy Efficiency in Data Centers &amp; Clouds, 1ed. Use dis
count code STC215 save up to 30% - https://t.co/22wZhOngr3 https://t.co/i2Rfh55
na6"

[[3]]
[1] "comp_science: Save up to 30% on all books &amp; ebooks, plus free shipping
! Use code \"STC215\" at checkout. https://t.co/CCplyURmbn https://t.co/1uYUXRJ
kXW"

[[4]]
[1] "comp_science: Save up to 30% on all books &amp; ebooks, plus free shipping
! Use code \"STC215\" at checkout. https://t.co/GNh6luCzv5 https://t.co/fEmzWg3
lDl"
```

Fig. 18: cluster5

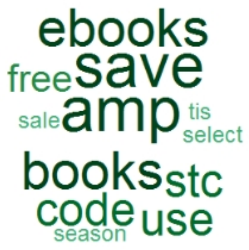


Fig. 19: cluster5 word cloud

V. Conclusions

Developed a prototype application using R programming language, that generates cluster based responses of tweets to queried words.
For k-means the user needs to give the input of the number of clusters where as for k-medoids the optimum number is calculated and used.
k-medoid relies on medoids computing by trying to minimize the absolute distance between the objects and the selected medoid, rather than trying to minimize the square distance.
The clustering produced by k-means is not that effective as that of k-medoids.
As a result, k-medoids is more robust than k-means.

VI. References

1) Gabrilovich, E., and Markovitch, S. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In Proceedings IJCAI.
2) Dhillon, I., and Modha, D. 2001. Concept decompositions for large sparse text data using clustering. Machine Learning. 29(2-3):103130.
3) Bekkerman, R.; El-Yaniv, R.; Tishby, N.; and Winter, Y. 2003. Distributional word clusters vs. words for text categorization. JMLR 3:11831208.
4) Twitter API.http://dev.twitter.com/
5) Hartigan, J. A. and Wong, M. A. (1979). A K-means clustering algorithm.Applied Statistics28, 100108.