Advances in Computer Science and Information Technology (ACSIT)
p-ISSN: 2393-9907; e-ISSN: 2393-9915; Volume 3, Issue 4; April-June, 2016, pp. 281-287

http://www.krishisanskriti.org/Publication.html

The Reliable Knowledge Discovery in Textual Database using R Infrastructure

Anu Yadav

University School of Information & Communication Technology, GGSIPU, New Delhi
E-mail: noor4anisha@gmail.com

Abstract—*In today's world, the internet and computer technology enormously increased the amount of stored information and unprecedented expansion in the amount of unstructured data in the textual formats, we cannot use the data for any processing to extract useful information, due to the rapid growth of digital data, and Information explosion and availability has changed the nature of information centers. Hence, knowledge discovery and text data mining have attracted an empirical attention with an imminent need for turning such data into useful information, patterns and knowledge. Text mining has become an interesting area in business intelligence application, healthcare, media and research. Text Mining can be defined as a technique which is a process used to analyze text to extract interesting and meaningful information from new or previously unknown information, non-trivial patterns or knowledge of the unstructured text documents or from different resources for particular purposes. The text mining is an interdisciplinary research held utilizing techniques from computer science, computational linguistics, information retrieval, data mining and statistics. Existing toolkits for text mining have low extensibility, lack of availability of application programming interfaces and provide less support for interacting with computing environments. Hence, in this paper, we propose a text mining in R infrastructure or computing environment, it provides intelligent methods for Meta data management and operations on documents, such as preprocessing, data cloud formation, frequency graphs, text clustering and text classification. This paper presents how text mining techniques can be applied in R infrastructure and better utilizing infrastructure features than other text mining products such as dtSearch, SPSS, SAS Text Miner, RapidMiner, weka, etc.*

Keywords*: Text Mining, classification, clustering and R.*

1. INTRODUCTION

The advancement in technology world expanding digital data but searching and extracting information for any kind of requirement is easily available on net (the world is one click away). The DM (data mining) can't mine unstructured data only TM (text mining) can extract knowledge from unstructured data and data available in digital world are 85% in unstructured form. To find relevant information, it is extremely difficult to read all document and explore whole data. Hence we applied TM techniques to convert unstructured data into relevant structured information, discovering useful and interesting trends and pattern from available data, to analysis collection of documents, to find relationship and associations among entities, identify and extract non trivial information by removing ambiguity & noisy data, discover knowledge from previously unknown or new data and provide browsing and navigation intelligence methods by using computer technology. There are different kinds of tools available for commercial tools and open source tools for TM process in market. Every tool have different purposes, features, disadvantages and advantages. The preprocessing of TM used different techniques to remove stop words, stemming, ambiguity, noisy data, etc. to find relevant information from raw data. There are different kind of text clustering and text classification algorithms used for TM. R infrastructure is a well-organized open source tool environment that provide TM techniques and infrastructure, and has a sophisticated modularized extension method for TM extraction purposes. The R infrastructure have high extensibility than other tools and providing intelligent method for TM preprocessing, text classification & text clustering.

2. PREPROCESSING

The TM from a preprocessed text is quite easy rather than directly from documents in natural languages. Hence, before applying TM techniques, preprocessing of documents are compulsory.

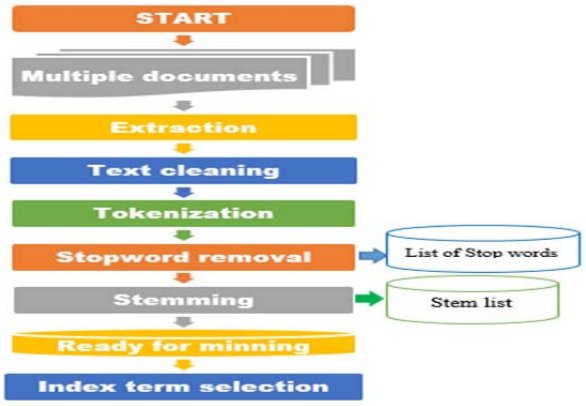

Fig. 1: Stepwise showing preprocessing activities for TM

The TM preprocessing convert unstructured data (85% data present on net are unstructured) into structured format used for pattern discovery by applying DM (data mining) algorithms. The Text preprocessing in TM include 9steps as shown in fig1.

TM preprocessing start with collection of multiple documents and extract structure and text data (pdf, Microsoft word, HTML/XML, etc.), it retrieve specific content text by investigating format and set of word. In this step text collected from different medium from different language (natural language used by human) and from data present in different formats (structured/ pdf/ semi structured/ unstructured / html/). The next step is text cleaning that used to clean characters and words and work on encoding for security purposes. It is a process that extract or clean noisy text such as spelling mistake, missing punctuation, repetitions, abbreviations, misleading/false information by text cleansing method. Now we need to do tokenization, it is a process of splitting sequence of string into phrases/keywords/words that is known as tokens (individual word/ whole statements) and removing punctuation, whitespace and line breaks that separate token/word from other token. Tokenization is a method used in text preparation for natural language documents that produce stream of processing units from stream of characters and named them tokens for data. The raw data undergoes in preprocessing for TM process and segmented into preprocessing textual units. Tokenization process used to clean and filter (whitespace). Tokenization have different kind of tokens, mainly divided into three types single tokens, multi tokens and user defined tokens. The single tokens are simplest type of token, it is a character string. The multi tokens are most commonly used tokens because 90% data is in complex natural language (unstructured) that can't be fit into single type of tokens. The multi tokens contains token delimiter character that formatted using blanks, tabs and new lines. The three basic types are Alphabetic, Numeric and Mixtures (hyphen, end markers). The third type of token is user defined tokens that used to reinterpret as well as group strings and basic token types. They used to support domain specific needs and knowledge such as structure of an organization, trends in database of organization and knowledge about data warehouses. They provide knowledge in gazetteer domain like name of country, state name, mountains name and river names. User tokens used for expert knowledge such as medical, scientist, or astronomy. Even they utilized in computational linguistic knowledge like subject and objective of sentences, morphological and syntactical rules. The examples of user-defined token types are stop words, phone numbers, abbreviations, dates and time, and email addresses. The removal of Stopwords from unstructured or semi structured data is required because 70% words occurring in natural language are non-relevant and don't carry any information for knowledge discovery. List of stop words that belongs to non-linguistic view and less relevant hence, these words are ignored or removed to reduced text data and they improve performance of systems. The most commonly occurring words that present in any unstructured document that don't provide meaningful knowledge like prepositions, pro nouns and articles. A, about, above, across, after, again, all, almost, along, already, also, always, among, amount, an, and, any, anyone, anything, anywhere, are, as, at, be, because, become, been, before, being, below, between, beyond, both, but, by, can, cannot, could, describe, detail, do, due, done, during, each, e.g., else, etc., ever, every, everyone, everywhere, find, for, found, from, further, get, give, go, had, has, have, hence, here, how, however, i.e., if, in, into, is, it, keep, least, less, made, many, may, me, might, more, moreover, most, much, must, neither, never, next, no, none, nor, not, nothing, now, of, off, on, once, one, only, or, other, otherwise, our, ours, over, part, per, put, rather, same, see, several, since, so, some, somehow, somewhere, something, sometime, someone, still, such, system, take, ten, than, that, the, their, them, themselves, then, thence, there, thereafter, thereby, therefore, therein, thereupon, these, they, this, those, throughout, thus, to, too, toward, un, under, until, up, us, very, was, we, well, were, what, whatever, when, whenever, where after, whereas, where, whereby, wherever, which, while, who, whoever, whole, whose, with, why, will, within, would, without, you. The Stemming is a technique that used in NLP, TM, and IR. It is defined as a method that reduce set of words into their stem/root word by eliminating prefixes and suffixes to get root word. The practical example that will clear this concept with well suitable example, the list of words present in English language that can be reduced into their base class or stem words, for example agrees, agreement, agreed, disagrees, agreeing, disagree, agreeable and disagreement all these have root/ stem word "agree". Other examples happiest, unhappy, happiness, happier, and unhappiness all are for word happy**.** There are lots of advantage of using stemming in preprocessing like Reducing indexing size, Improving effectiveness of IR and text mining, improve recall, and Matching similar words. The stemming process stated the identification of words those have similar meaning or these words come from same root word but presenting in different form.

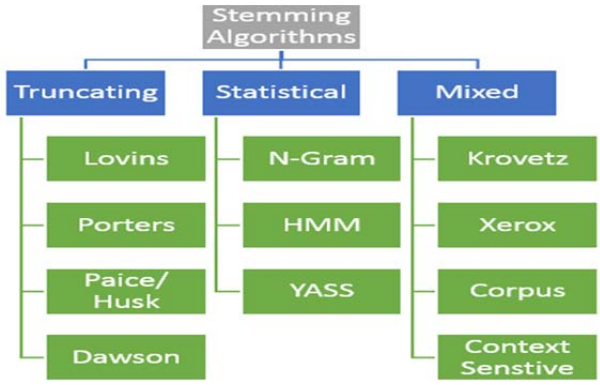

Fig. 2: List of Stemming algorithms in Hierarchical way.

To perform stemming process there are different kinds of algorithms invented till now (fig2). The POS (part of speech tagging) is a process that assign lexical class marker to each word of documents in a corpus. The Part Of Speech Tagging is the word annotation with the appropriate POS tags based on the context in which they appear. POS mainly deal with number, article, verb, nouns (thing in intangible & tangible form), prepositions (relation between nouns), and adjectives (they are tags of POS).

2.1 Preprocessing in R programming

R language open source tool is used for this research paper that doing some preprocessing and produce some text analytical representation from text. The Preprocessing techniques used here in code are removing of punctuation from text files, removing of numbers from text files, Converting uppercase into lowercase, elimination of stop words, stemming by removing prefix or suffix or common word endings (ing, s) and Stripping of sentence by blank lines and whitespace. This step allow us to prepare our texts for analysis. These methods take time & picky, but it pays in the end in terms of high quality analyses. The code for preprocessing is given below (Fig. 3).

```
## Preprocessing

# *Removing punctuation:*
docs <- tm_map(docs, removePunctuation)

# *Removing numbers:*
docs <- tm_map(docs, removeNumbers)

# *Converting to lowercase:*
docs <- tm_map(docs, tolower)

# *Removing "stopwords"
docs <- tm_map(docs, removeWords, stopwords("english"))

library(Snowballc)
# *Removing common word endings* (e.g., "ing", "es")
docs <- tm_map(docs, stemDocument)

# *Stripping whitespace
docs <- tm_map(docs, stripWhitespace)
docs <- tm_map(docs, PlainTextDocument)
## *This is the end of the preprocessing stage.*
```

Fig. 3 Preprocessing by R Programming

3. TEXT ANALYSIS BY R PROGRAMMING

Text analytics (TA) is also known as TM (text mining) and TDM (text data mining), it is the process of analyzing structured/ semi structured/ unstructured text documents, deriving high-quality (means providing novelty, relevance, and interestingness) of information from documents, extracting useful as well as relevant information, producing trends and patterns as outcome by using statistical learning, and transforming it into useful knowledge . TA include concept/ name entity extraction, text categorization, pattern recognition, document summarization, text clustering, IE (information extraction), sentiment analysis, tagging/annotation, IR (information retrieval), word cloud formation, DM (data mining like visualization, predictive analytics, and link & association analysis), lexical analysis to study word frequency distributions, and entity relation modeling. The analytical methods and NLP they produce unstructured text into data that can be analyzed. These techniques discover/present /visualize knowledge in terms of fact, rule, & relationships.

Here in this paper we discuss word frequency, word cloud, frequency graph, text clustering and classification for TA.

3.1 Word Frequency in R

There are lot of terms in unstructured documents, so we will just check out the most and least frequently occurring words. The below code chunk will produce the output in two types of output (Fig. 4). The first output is of 2 rows, the top row reflecting that which words appear more than a more or less frequently & the bottom row number reflects how many times words appear that term frequently. The below code chunk give a matrix form that was created using the techniques of data transformation. An alternate view of term frequency or word frequency is shown by the second output that produce a list of all types of words that occur more than 50 times frequently occur in text file.

```
> freq <- colSums(as.matrix(dtms))
> freq

   boolean    buildid     called       case       code    complet   complete       core    details
        17       1071         23         12         26          6          7          4         12
      diff       done        els      entri     except       exit       fail     failed    failure
        18         19          6          5          8          4          7          5          4
      file       find      force   function       grep installing      local  localized        mac
        71         26          4          5          4         15         17          3          8
modifying       name    nonzero        old     packag    package    partial       path   platform
         6         84          2         16          7         19         15         16         11
   product     releas    release     return   returned        sed      setup     source        tar
       170         10       1092         13          5          2        142         12          5
    target       type    unknown     update        url       valu       warn    warning        win
        15         32          2         10         29         27         27          5          4

> #
> findFreqTerms(dtm, lowfreq=50)
 [1] ""ach""          ""af""           ""beta""         ""complete""
 [5] ""darwinxgccuix"" ""de""          ""firefox""      ""http""
 [9] ""linux""        ""linuxxgcc""    ""winnt""        "america"
[13] "american"       "ast"            "bdmg"           "beta"
[17] "bexe"           "blinuxx"        "bmac"           "bnbd"
[21] "bnin"           "build"          "bwin"           "can"
[25] "change"         "channel"        "csb"            "engb"
[29] "enus"           "enza"           "esar"           "escl"
[33] "eses"           "esmx"           "firefox"        "firefoxbtarbz"
[37] "ftp"            "fynl"           "gaie"           "gcc"
[41] "guin"           "hiin"           "hyam"           "jajpmac"
[45] "lij"            "linux"          "locale"         "locales"
[49] "mai"            "mozillaorg"     "mozillaorg""    "msvc""
[53] "nbno"           "new"            "nnno"           "one"
[57] "pain"           "past"           "patch"          "platform"
[61] "product"        "productfirefox" "promise"        "ptbr"
[65] "ptpt"           "pub"            "release"        "releases"
[69] "server"         "setup"          "son"            "stage"
[73] "svse"           "test"           "time"           "types"
[77] "update"         "will"           "zhcn"           "zhtw"
```

Fig. 4. Shows word frequency of document in R language

3.2. Word Cloud in R

The chart, info graphic & graph used for data visualization (DV), it produces valuable way to communicate & extract important & useful information at a glance but when raw data provided in text documents then we use word clouds that give us stunning visualization, highlight useful textual data points, and immediately convey crucial information present in documents. The word cloud (fig5) also known as tag cloud or text clouds, it is a pictorial representation of words according to their repetitions in documents. The more specific words appears in text documents are bolder and bigger they appear in the word cloud representation. To show more interesting we give it color as human eye are more attracted towards colors. Humans are generally strong at visual analytics.

```
> dtms <- removeSparseTerms(dtm, 0.15) # Prepare the data (max 15% empty space)
> freq <- colSums(as.matrix(dtm)) # Find word frequencies
> dark2 <- brewer.pal(6, "Dark2")
> wordcloud(names(freq), freq, max.words=100, rot.per=0.2, colors=dark2)
```

Fig. 5. The word cloud for text documents generated by R.

3.3. Word Frequency Graph in R

The number of times a certain type of class or values occur in text document that used for TM are presented in graph that display the data by using y axis (vertical bars) of various heights (No. of times word occur in text document) to represent frequency & the x-axis (horizontal axis) of graph plot words. The word frequency graph implemented in R by using code and produce output shown in fig 6.

```
> wf <- data.frame(word=names(freq), freq=freq)
> p <- ggplot(subset(wf, freq>50), aes(word, freq))
> p <- p + geom_bar(stat="identity")
> p <- p + theme(axis.text.x=element_text(angle=45, hjust=1))
> p
```

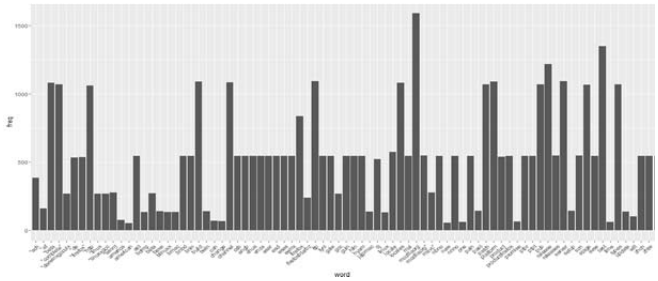

Fig. 6. The word frequency for documents generated by R.

3.4. Text Clustering & Text Categorization

There are different kinds of methods that can be apply on text documents for TM process. We can implement these methods on text documents by using various algorithms.

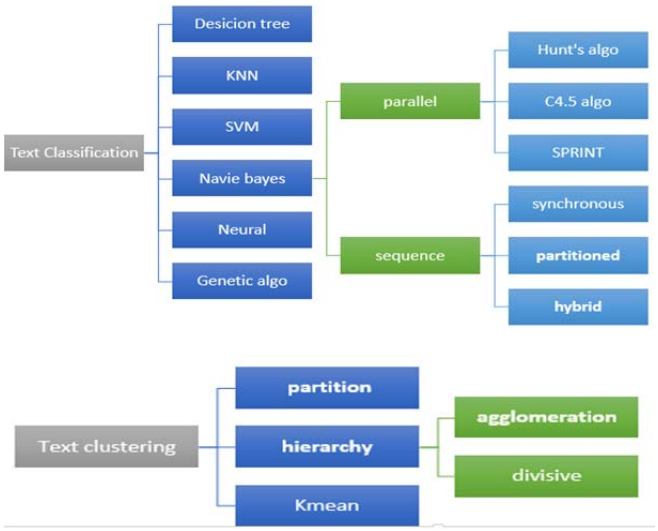

Fig. 7. Hierarchy str. for text clustering & categorization.

There are 5 famous methods that used in TM (table1) are Text categorization (TCA), summarization and Text clustering (TC). The Text Categorization (TCA) is defined as the process of finding correct and useful topic of text document from a set of subject/object/categories/topics and a collection of text documents. It is a type of supervised learning that classify and categories text documents. The applications of TCA are Document Sorting, Hierarchical Web Page Categorization, Text Indexing and Text Filtering. The problems present in text document are Classify and categorized based on Single-Label Categorization or Multi-label Categorization, Hard Categorization or soft Categorization & Document-Pivoted Categorization or Category-Pivoted Categorization.

Table 1: TM technique, characteristics, algorithms, model & tools.

Techniques	Text/Information Retrieval (IR)	Text Summarization	Text Categorization	Text Clustering
Characteristic	Retrievals valuable information from unstructured text. Document retrieval	Reduce length by keeping its main points and overall meaning as it is.	Deal with large amount of text document. Used in indexing. Documents to assist IR tasks in classifying email.	It is Used as an unsupervised learning. Goal is descriptive Cluster. Clustering, classification & sentimental analysis of text document

Algorithms	Stop Word Removal Stemming Word (term) extraction Inverted Index Signature file	Keyphase Extraction TextRank PageRank KEA ROUGE	K-NN Support Vector Machine Decision Tree Induction	K-Mean & K-Medoids Agglomerative & Divisive DBSCAN STING & CLIQUE
Models	Boolean Model, Vector Space Model Statistical Language Model	Naïve Bayes Model	Support Vector Machines (SVM) Probabilistic or generative model based	Statistical Model Support Vector Machines (SVM)
Tools	Intelligent Miner Text Analyst	Tropic Tracking Tool	Intelligent Miner	Carrot Rapid Miner

The TCA used for machine learning techniques (fig 7) categorization firstly into vector space model based on algorithms KNN (K-nearest neighbor), Decision-tree, Support Vector Machines (SVM) and Neural Networks. Secondly Probabilistic or generative model based on algorithm Naïve Bayes classifier. The TC also known as DC (Document clustering), It is the process of cluster analysis for text documents. It has applications in automatic document organization, theme extraction, topic extraction, IR (information retrieval) and Information filtering. DC is an unsupervised learning technique that used for grouping of text document into useful and meaningful objects/cluster (group of objects into a single similar/ related group). The clustering process divided into three process data preprocessing, hierarchical clustering, and model based clustering (K-mean algorithm). There are mainly 2 types of text clustering or document clustering hierarchical and partition clustering. The hierarchical text clustering find successive clusters by using the previously available clusters, they are sub divided into Agglomerative and Divisive hierarchical text clustering. The Agglomerative algorithms are belonging to bottom up approach, it is a process that start with each element as a separate cluster & combine them into successively larger clusters. The Divisive algorithms are belonging to top-down approach, it is a process that start with the whole set of text document and proceed to divide it into successively smaller clusters. Whereas the Partitional algorithms specify and determine each and every clusters at once. They include *K*-means and derivatives**,** Fuzzy *c*-means clustering, and QT clustering algorithm. The clustering hierarchy for text clustering or documents clustering we used cluster dendrogram presentation by R implementation. Some people find cluster dendrograms to be fairly clear for reading and understanding purpose. Here, we can make many groups that are identifiable in the dendrogram. Here, I have arbitrarily chosen to look at five clusters, as indicated by the red boxes in Fig. 8. The K-mean clustering algorithm is a process that produce cluster n objects based on attributes into k partitions, where k is smaller than n but k is positive integer number. In other words, it is an algorithm that classify or it used to group the similar objects based on attributes/features into K number of group. This algorithm assume that the object attributes form a vector space. The grouping is achieved by minimizing the sum of squares of distances between raw data and the corresponding cluster centroid. The working of K-mean algorithm is explained in Fig. 11.

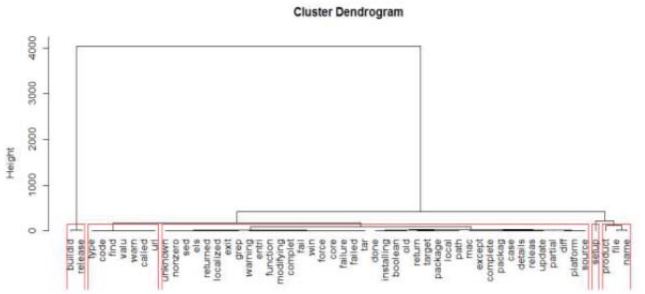

Fig. 8. Hierarchical clustering for cluster Dendrogram.

```
### Clustering by Term Similarity
### Hierarchal Clustering
dtms <- removeSparseTerms(dtm, 0.15)
# This makes a matrix that is only 15% empty space.
library(cluster)
d <- dist(t(dtms), method="euclidian")
# First calculate distance between words
fit <- hclust(d=d, method="ward")
plot.new()
fit
plot(fit, hang=-1)
groups <- cutree(fit, k=5)
# "k=" defines the number of clusters you are using
rect.hclust(fit, k=5, border="red")
# draw dendogram with red borders around the 5 clusters
### K-means clustering
library(fpc)
library(cluster)
dtms <- removeSparseTerms(dtm, 0.15)
# Prepare the data (max 15% empty space)
d <- dist(t(dtms), method="euclidian")
kfit <- kmeans(d, 2)
clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0)
```

Fig. 9. Code for hierarchical and Kmean clustering in R.

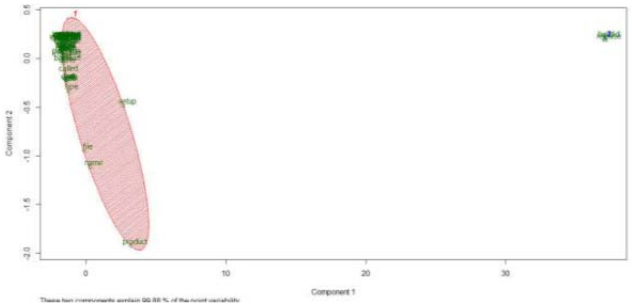

Fig. 10. 2clusters formed for document by K-mean algorithm.

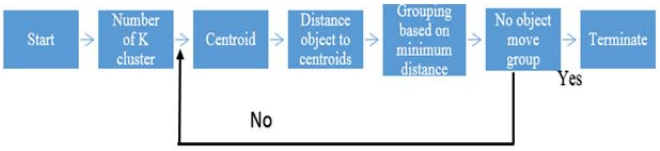

Fig. 11. The flowchart of Process for K-Mean algorithm.

4. TOOLS IN TM

The TDM (Text Data Mining) or TA (Text Analytics) analysis, represented and processed with the help of different kinds of tools available in the market, these tools basically divided into 2 parts commercial text tools and open text tools. There are more than 25 software or tools available for commercial text tools and more than 10 tools for open text tool. The both type of tools have some advantage and disadvantages, but open text tools provide API (application programming interface) feature. In table 2, I have described few list of tools with their feature. The tick mark shows these feature present in tool and blank means this feature do not presented by this tool.

Table 2. Comparison of tools based on Text mining features.

	P	A	CL	SU	CA	SE	PA	NLP
Commercial Text Tools								
Angoss	√	√	√	√	√	√	√	
Basis	√							√
Clarabridge			√		√	√		√
Cogito					√	√		√
IBM SPSS	√	√		√	√	√	√	√
Inxight(SAP)	√	√	√	√	√		√	√
Lexalytics	√	√				√		√
Megaputer	√		√	√	√		√	√
SAS Miner	√	√	√	√	√	√		√
Autonomy			√		√			
Netowl	√					√	√	
DT search	√	√		√				
Mathematica			√			√	√	
Open Text Tools								
Gate	√	√	√	√	√			
Rapid Miner	√	√	√	√	√			
Open NLP	√					√		√
Carrot2			√				√	
NLTK	√						√	√
R	√	√	√	√	√			
Weka	√	√	√	√	√			
Gensim	√					√		√
Datumbox	√					√		√

The features that described in table for TA are P, A, CL, SU, CA, SE, PA, and NLP. The P stands for preprocessing that explained in section 2, it consist Theme extraction, Name entity extraction, tokenization, POS (part of speech), removal of Stopwords and stemming. The A stands for association analysis in documents. The CL (text clustering), SU (text summarization), and CA (text categorization) theses are explained in table 1 with tools, methods, algorithm and main concept of these three methods for TA. The SE **(**sentiment analysis), PA (pattern analysis), and NLP (natural language processing) features are used in machine learning, business intelligence, TA, statistics and artificial intelligence.

The list of commercial text tool, Angoss (entity and theme extraction, topic categorization, sentiment analysis and document summarization, text-based analysis with predictive models and association analysis), Attensity (text mining, NLP, customer relationship management, research, e-discovery, and intelligence analysis), Autonomy (text mining, clustering and categorization), Averbis (text analytics, clustering & categorization), Clarabridge (text analytics, NLP, machine learning, clustering, categorization, & sentiment analytics), Complete Discovery Source (data discovery and data analytics), Inxight (text analytics, search, & unstructured visualization technologies), Luminoso (NLP, machine learning and artificial intelligence), Mathematica (text alignment, pattern matching, clustering and semantic analysis), Megaputer Intelligence (NLP, machine learning, sentiment analysis, entity extraction, clustering, and categorization), NetOwl (entity extraction, link and event extraction, sentiment analysis, name translation, name matching, and identity resolution), SAS Text Miner (text analytics, NLP, & Information Management), Sysomos (media analytics, text analytics and sentiment analysis), DiscoverText (cloud-based text analytics), dtSearch (indexing, searching, & retrieving), Entrieva (indexes, categorizes & organizes unstructured text), IBM SPSS (Predictive Analytics), Intellexer (NLP, & summarization), KBSPortal(NLP), Linguamatics (NLP), MeaningCloud (text analytics), Megaputer Text Analyst (semantic analysis, summarization, clustering, navigation, & NLP), Monarch (data access & analysis), MonkeyLearn (Text Mining tool, Machine Learning applications), NetOwl (sentiment analysis), NewsFeed Researcher (summarization tool), Plagiarism Software (online check plagiarism), Recommind MindServer (Semantic Analysis), SAS Text Miner (text processing and analysis tools), SIFT (customer feedback analysis and reporting, NLP, & machine learning), TeSSI®, (semantic indexing, searching, coding and IE), Textalyser (online text analysis), TextPipe Pro (text conversion, extraction and manipulation), TextQuest, (text analysis software), Treparel KMX (Text Analytics), Readware (Information Processor), VantagePoint (interactive graphical views and analysis tools), VisualText (text analytics development environment, NLP), VP Student Edition (TM and visualization tool), Xanalys Indexer (IE and DM library) and Wordstat (analysis module).

The list of open source text tools, Carrot2 (text and search results clustering), GATE (General Architecture for Text Engineering, NLP), Gensim (Topic modelling and extraction of semantic information), OpenNLP (NLP), Natural Language Toolkit (NLTK - NLP), Orange (text mining), Stanbol (semantic content management), The programming language R provides a framework for text mining applications in the

package *tm* and NLP task view contains *tm* and other text mining library packages, KNIME (Text Processing extension), And PLOS (Text Mining).

5. CONCLUSION

The preprocessing process is understood by flowchart diagram. The R language tool providing text clustering, text categorization, text summarization, association and preprocessing for text data mining with representation of data in word cloud, word frequency and frequency graph for text analysis. There are different features provided by different kinds of algorithm that execute for text mining of document. The K-means algorithm is text clustering algorithm that useful for undirected KD (knowledge discovery) and it is simple. There are 2 types of tools studied in this paper with their features.

REFERENCES

[1] Mrs. Sayantani Ghosh1, Mr. Sudipta Roy2, and Prof. Samir K. Bandyopadhyay, "A tutorial review on Text Mining Algorithms", *International Journal of Advanced Research in Computer and Communication Engineering*, IJARCCE, *Vol. 1, Issue 4, June 2012*,

[2] Yu Zhang; Mengdong Chen; Lianzhong Liu, "A review on text mining", *International Conference on Software Engineering and Service Science (ICSESS)*, 2015, Pp. 681 – 685.

[3] D. Sánchez, M. J. Martín, I. Blanco; C. Justicia de la Torre, "Text Knowledge Mining: An Alternative to Text Data Mining", *IEEE International Conference on Data Mining Workshops*, 2008, Pp. 664 – 672.

[4] K.L.Sumathy, M.Chidambaram, "Text Mining: Concepts, Applications, Tools and Issues – An Overview", *International Journal of Computer Applications (0975 – 8887)* Volume 80 – No.4, October 2013.