**I Data**

**20 Newsgroups**

The 20 Newsgroups data set is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. The data is organized into 20 different newsgroups, each corresponding to a different topic. Some of the newsgroups are very closely related to each other (e.g. comp.sys.ibm.pc.hardware / comp.sys.mac.hardware), while others are highly unrelated (e.g misc.forsale / soc.religion.christian).

Following is a list of 20 Newsgroup, partitioned according to subject matter.

|  |  |
| --- | --- |
| comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x | rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey |
| sci.crypt sci.electronics sci.med sci.space | misc.forsale |
| talk.politics.misc talk.politics.guns talk.politics.mideast | talk.religion.misc alt.atheism soc.religion.christian |

**Yelp Dataset**

Yelp dataset has variety of business on which user gives reviews and start rating. It contains total 736 documents of different 168 business like Accessories, ArtsCrafts, Cafes, DiveBars etc.

For 20 Newsgroup contains following 6 news topics

|  |  |  |  |
| --- | --- | --- | --- |
| Group No. | New topics group | Total number of documents in group. | Unique words |
|  |  |  |  |
| 1. | comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x | 1000 documents per newsgroup.  Therefore total 5000 document. | Graphics,  Window Misc,  Pc Hardware,  Mac Hardware,  Window.X. |
| 2. | rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey | 1000 documents per newsgroup.  Therefore total 4000 document. | Autos,  Motercycles,  Baseball,  Hockey. |
| 3. | sci.crypt sci.electronics sci.med sci.space | 1000 documents per newsgroup.  Therefore total 4000 documents. | Crypt,  Electronics,  Med,  Space. |
| 4. | misc.forsale | Total 1000 documents. | Foresale. |
| 5. | talk.politics.misc talk.politics.guns talk.politics.mideast | 1000 documents per newsgroup.  Therefore total 3000 documents. | Politics Misc,  Politics Guns,  Politics Mideast. |
| 6. | talk.religion.misc alt.atheism soc.religion.christian | 1000 documents per newgroup except soc.religion.christian which contains total 997 document.  Therefore total 2997 documents. | Religion Misc,  Athesim,  Religion Christian |

**II Experiments**

**II.A Data Preprocessing**

**-20Newsgroup Dataset**

Total processing 3000 documents. I am using comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, comp.os.ms-windows.misc newsgroups which are related to hardware and windows group. Because of the same group most of the words are classified into same cluster i.e we get less number of clusters.

**Command:**

*install.packages("tm")*

*install.packages("SnowballC")*

*install.packages("wordcloud")*

*install.packages("RColorBrewer")*

*library("tm")*

*library("SnowballC")*

*library("wordcloud")*

*library("RColorBrewer")*

**Output :**

Install packages tm, Snowballc,wordcloud,RColorBrwer.

**Load Data**

**Command:**

*data2 <- c("E:\\IIT\\Lecture\\CS522 ADM\\HW1\\20\_newsgroups\\comp.sys.ibm.pc.hardware",*

*"E:\\IIT\\Lecture\\CS522 ADM\\HW1\\20\_newsgroups\\comp.sys.mac.hardware",*

*"E:\\IIT\\Lecture\\CS522 ADM\\HW1\\20\_newsgroups\\comp.os.ms-windows.misc")*

**Output:**

Read text file for 3 groups comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, comp.os.ms-windows.misc.

**Command:**

*news <- Corpus(DirSource(data2, recursive=TRUE),readerControl = list(reader=readPlain))*

**Output:**

Load data as corpus

**Text Preprocessing**

**Command:**

*news <- tm\_map(news, removeWords,"Subject")*

*news <- tm\_map(news, removeWords,"Organization")*

*news <- tm\_map(news, removeWords,"writes")*

*news <- tm\_map(news, removeWords,"From")*

*news <- tm\_map(news, removeWords,"lines")*

*news <- tm\_map(news, removeWords," NNTP-Posting-Host")*

*news <- tm\_map(news, removeWords,"article")*

*news <- tm\_map(news, tolower)*

*news <- tm\_map(news, removeWords, stopwords("english"))*

*news <- tm\_map(news, removePunctuation)*

*news <- tm\_map(news, stemDocument)*

*news <- tm\_map(news, removeNumbers)*

*news <- tm\_map(news, stripWhitespace)*

*news <- tm\_map(news , PlainTextDocument)*

**Output:**

the tm\_map() function is used to remove unnecessary white space, number,punctuation,to convert the text to lower case, to remove common stopwords like ‘the’, “we”, to make text stemming.

**Build Document Term Matrix**

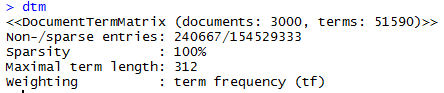
**Command:**

*dtm <- DocumentTermMatrix(news,control=list(wordLengths=c(4,Inf)))*

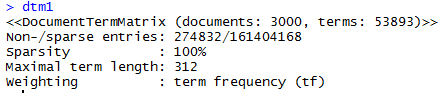
*dtm1 <- DocumentTermMatrix(news)*

**Output:**

Create document term matrix which contain list of word greater than 4 in length and stored in dtm. There are total 51590 terms.



Create document term matrix which contains any length of words and stored in tdm1. There are total 53893 terms.



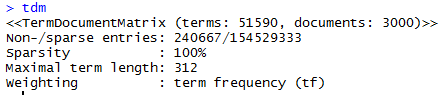
**Command:**

*tdm <- TermDocumentMatrix(news, control=list(wordLengths=c(4,Inf)))*

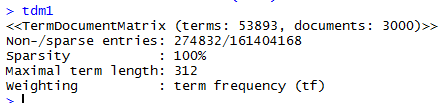
*tdm 1<- TermDocumentMatrix(news)*

**Output:**

Create term document matrix which contains list of words greater than 4 in length and stored in tdm variable. There are total 51590 terms.



Create term document matrix which contains any length of words and stored in tdm1 variable. There are total 53893 terms.



**Verify Frequent Term**

**Command:**

*m <- as.matrix(tdm)*

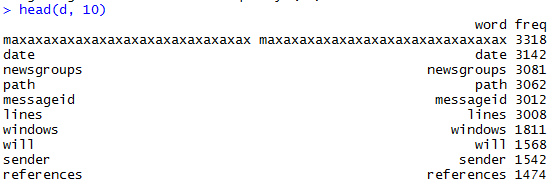
*v <- sort(rowSums(m), decreasing=TRUE)*

*d <- data.frame(word = names(v),freq=v)*

*head(d, 10)*

**Output:**

It displays frequency of each words/terms in document. There are total 51590 words/terms.



**Wordcloud**

**Command:**

*dtms <- removeSparseTerms(dtm, 0.15)*

*freq <- colSums(as.matrix(dtm))*

*dark2 <- brewer.pal(6, "Dark2")*

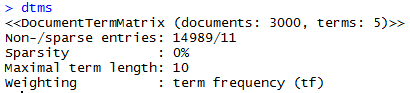
*wordcloud(names(freq), freq, max.words=100, rot.per=0.2,colors=dark2)*

**Output:**

*-removeSparseTerms* function used to remove sparse term from document term matrix.

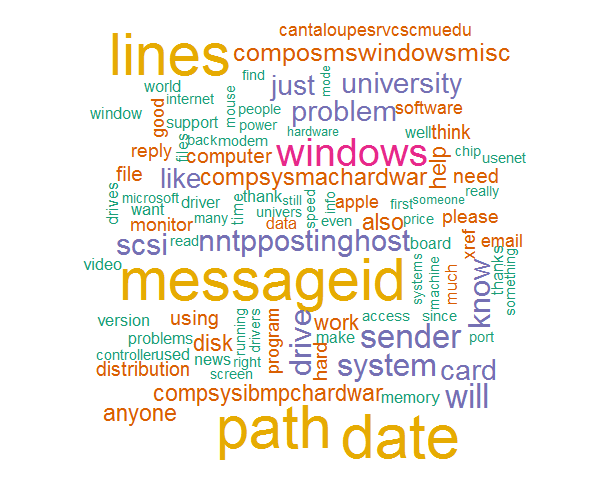
-Number of terms before removing sparse, terms are 51590, maximum term length is 312, sparsity is 100%.

-After removing sparse, terms are 5, maximum term length is 10, sparsity is 0%.



-Freq find the words frequency of document term matrix.

-Wordcloud usd to plot the word cloud.



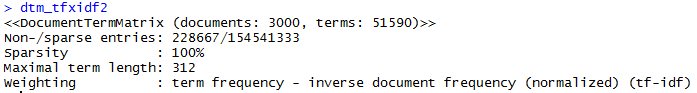
**InverseDocument Frequency**

**Command:**

*dtm\_tfxidf2<- weightTfIdf(dtm)*

**Output:**

*weightTfIdf* function weight a document-term matrix by term frequency - inverse document frequency.



**Yelp Dataset**

I process 736 documents of different 168 business. Because documents are from different business, we will get more numbers of clusters.

**Command:**

*data2 <- c("E:\\IIT\\Lecture\\CS522 ADM\\HW1\\reviewdata\\")*

*yelp <- Corpus(DirSource(data2, recursive=TRUE),readerControl = list(reader=readPlain))*

**Output:**

Load all yelp review data as corpus. Total 736 documents are loaded.

**Command:**

*yelp <- tm\_map(yelp, tolower)*

*yelp <- tm\_map(yelp, removeWords, stopwords("english"))*

*yelp <- tm\_map(yelp, removePunctuation)*

*yelp <- tm\_map(yelp, stemDocument)*

*yelp <- tm\_map(yelp, removeNumbers)*

*yelp <- tm\_map(yelp, stripWhitespace)*

*yelp <- tm\_map(yelp , PlainTextDocument)*

**Output:**

the tm\_map() function is used to remove unnecessary white space, number,punctuation,to convert the text to lower case, to remove common stopwords like ‘the’, “we”, to make text stemming.

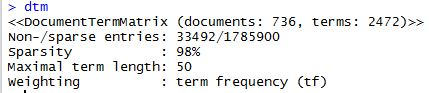
**Command:**

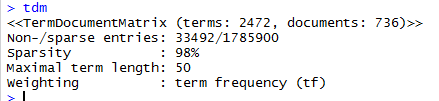
*dtm <- DocumentTermMatrix(yelp,control=list(wordLengths=c(4,Inf)))*

*tdm <- TermDocumentMatrix(yelp, control=list(wordLengths=c(4,Inf)))*

**Output:**

Create document term matrix using *DocumentTermMatrix* function and term document matrix using*TermDocumentMatrix.*





**Verify Frequent Terms**

**Command:**

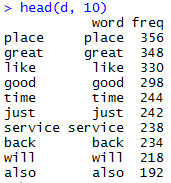
*m <- as.matrix(tdm)*

*v <- sort(rowSums(m), decreasing=TRUE)*

*d <- data.frame(word = names(v),freq=v)*

*head(d, 10)*

**Output:**



**Wordcloud**

**Command:**

*dtms <- removeSparseTerms(dtm, 0.15)*

*freq <- colSums(as.matrix(dtm))*

*dark2 <- brewer.pal(6, "Dark2")*

*wordcloud(names(freq), freq, max.words=100, rot.per=0.2,colors=dark2)*

**Output:**

Using wordcloud we plot word cloud which represent more frequent terms in documents.



**Frequent Words:**

**Command:**

*ord <- order(freq,decreasing=TRUE)*

*freq[head(ord)]*

**Output:**

It will display more frequent words with their frequency.



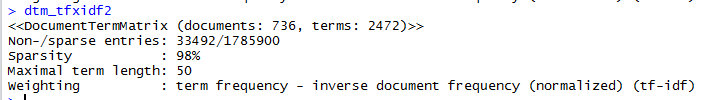
**InverseDocument Frequency**

**Command:**

*dtm\_tfxidf2<- weightTfIdf(dtm)*

**Output:**

*weightTfIdf* function weight a document-term matrix by term frequency - inverse document frequency.



**II.B Clustering Experiments.**

**20Newsgroup dataset**

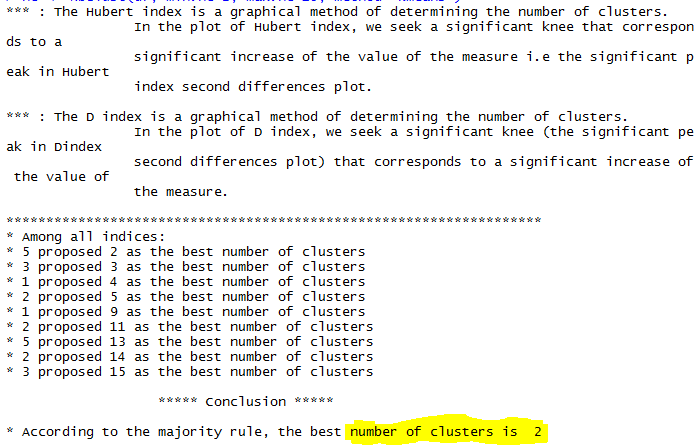
**1.**

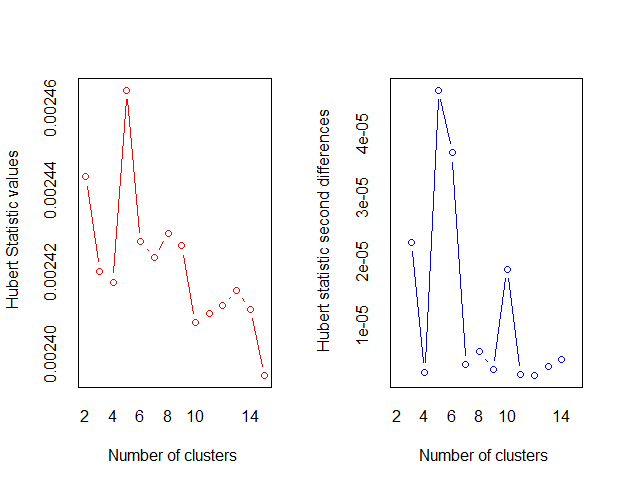
**Command:**

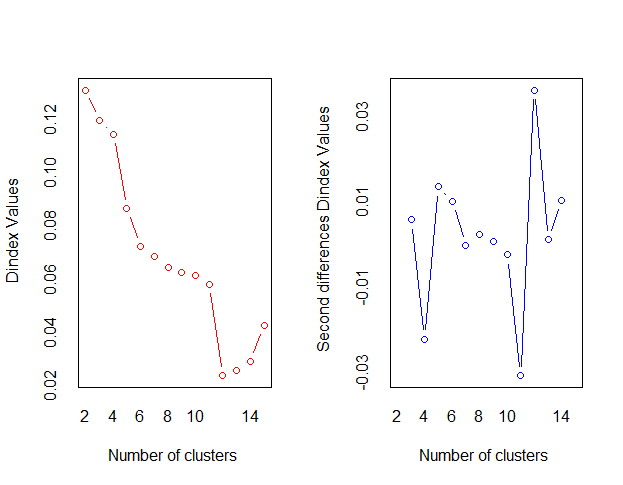
> df = as.matrix(dtms)

> nc <- NbClust(df, min.nc=2, max.nc=15, method="kmeans")

**Output:**







We can set parameter of k using NbClust. Using NbClust we can find best number of cluster as 2.

**Yelp Dataset**

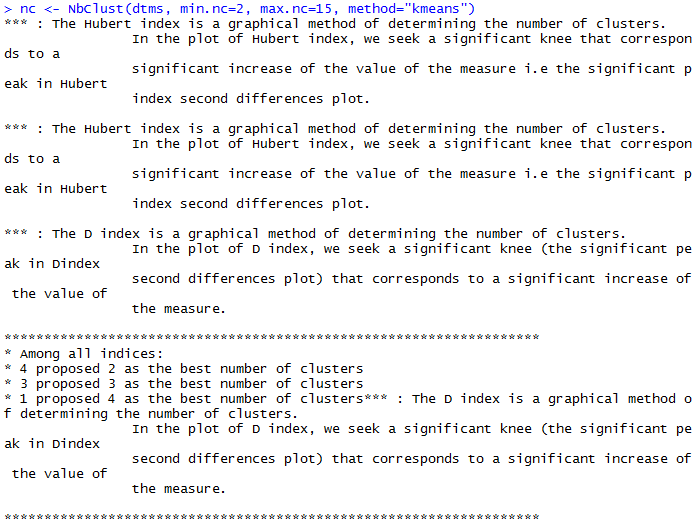
**1.**

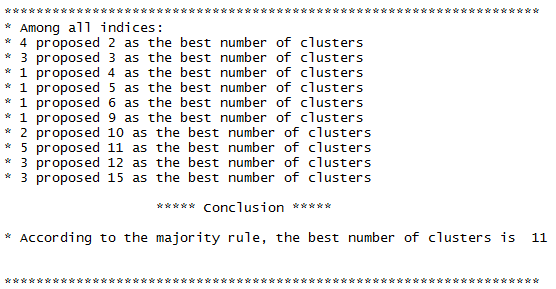
**Command:**

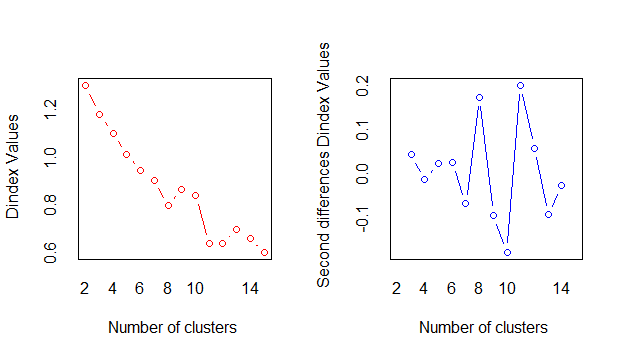
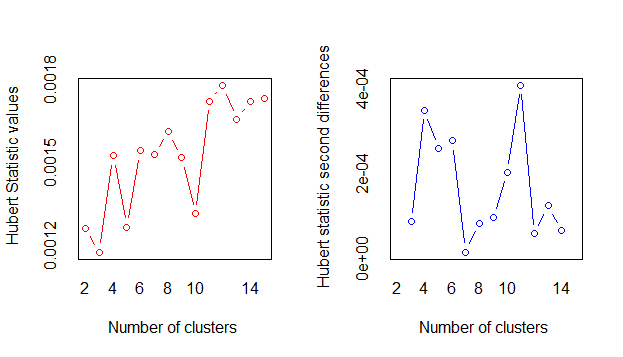
*nc <- NbClust(dtms, min.nc=2, max.nc=15, method="kmeans")*

**Output:**

As a result, we observed that best number of cluster is 11.







Using above graph, we can conclude number of clusters are 11.

**2. Compute the LSA representation, cluster LSA document vectors**

-**Compute the SVD of the document-term matrix**

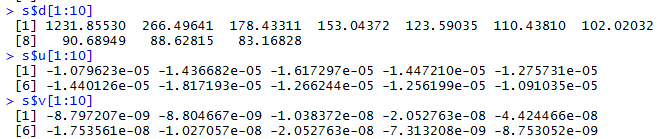
**20Newsgroup Dataset**

**Command:**

*X = as.matrix(dtm)*

*s = svd(X)*

**Output:**



**Yelp Dataset**

**Command:**

*X = as.matrix(dtm)*

*s = svd(X)*

**Command:**







**20Newsdataset**

**Clustering for dtm data using k value 2.**

**Command:**

*cl <- kmeans(X, 2)*

**Output:**

Above command performs kmeans algorithm for document term matrix with k value 2.







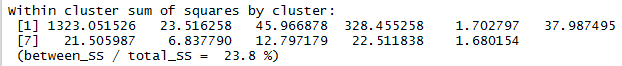
As shown above all documents are divided into 2 clusters and percentage of within cluster sum of square by cluster is 0.6%. Top 10 words of cluster 1 and2 are display above.

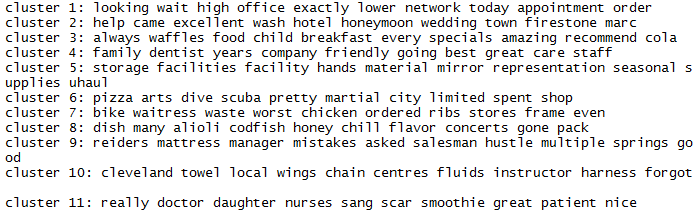
**Yelp Dataset**

**Clustering using no. of cluster 11**









As a result, we find top 10 words for different cluster as shown above.

-**LSA document vectors and LSA word vectors from the SVD**

From svd we can find the value of d,u,v where u and v are orthonormal i.e. orthogonal unit vector which can be represented as LSA document vector and LSA word vectors respectively and d is diagonal matrix. The columns of u and v are eigenvector of XXT and XTX respectively. The entries in d are the square roots of of the non-zero eigenvalues of XXT and XTX.

**Compute the d=50, 100, 200 dimensional representation for the term-document matrix**

**Command:**

*X = as.matrix(tdm)*

*s = svd(X)*

*D <- diag(s$d)*

*# d = 50*

*Uk<-(s$u[, 1:50])*

*Vk<-(s$v[, 1:50])*

*# d = 100*

*Uk<-(s$u[, 1:100])*

*Vk<-(s$v[, 1:100])*

*# d = 200*

*Uk<-(s$u[, 1:200])*

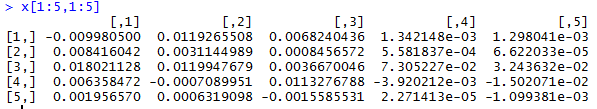
*Vk<-(s$v[, 1:200])*

*x = Uk %\*% D %\*% t(Vk) # X = U D V'*

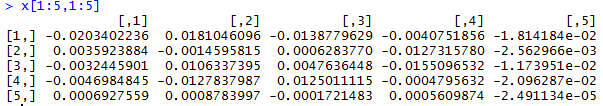
*X1 = t(Uk) %\*% X %\*% Vk # D = U' X V*

**Output:**

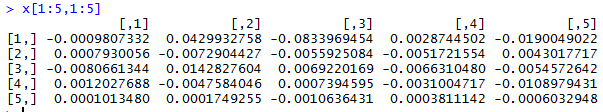
**20Newsgroup Dataset / d=50**



**20Newsgroup Dataset / d = 100**

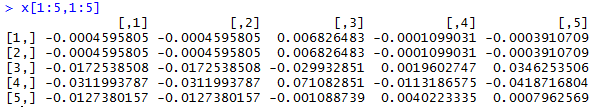


**20Newsgroup Dataset / d=200**

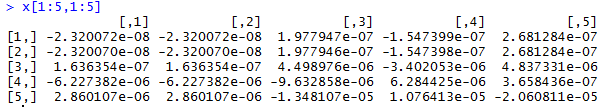


**Output:**

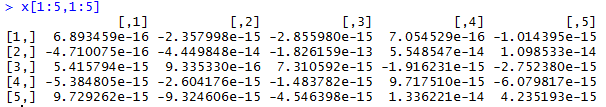
**Yelp Dataset / d = 50**



**Yelp Dataset / d = 100**



**Yelp Dataset / d = 200**



**Cluster d=50**

**Command:**

*cl <- kmeans(Uk %\*% D %\*% t(Vk), 2)*

**Output:**

**20Newsgroup Dataset / d=50**

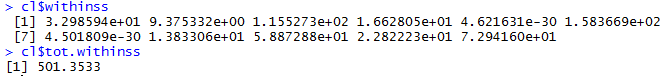
Above command performs kmeans algorithm for document term matrix with k value 2.

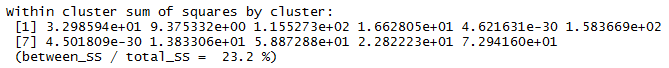




As shown above all documents are divided into 2 clusters and percentage of within cluster sum of square by cluster is 7.4%.

**Yelp Dataset / d=50**





As shown above all documents are divided into 11 clusters and percentage of within cluster sum of square by cluster is 23.2%.

**Cluster d=100**

**Command:**

*cl <- kmeans(Uk %\*% D %\*% t(Vk), 2)*

**Output:**

**20Newsgroup Dataset / d=100**

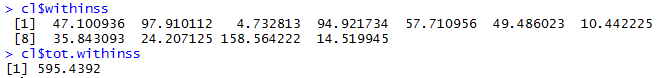
Above command performs kmeans algorithm for document term matrix with k value 2.

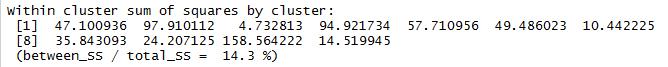




As shown above all documents are divided into 2 clusters and percentage of within cluster sum of square by cluster is 4.8%.

**Yelp Dataset / d=100**





As shown above all documents are divided into 11 clusters and percentage of within cluster sum of square by cluster is 14.3%.

**Cluster d=200**

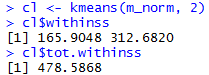
**Command:**

*cl <- kmeans(Uk %\*% D %\*% t(Vk), 2)*

**Output:**

Above command performs kmeans algorithm for document term matrix with k value 2.

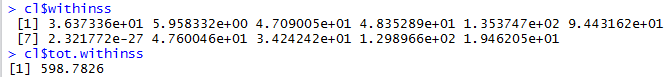
**20NewsGroup Dataset / d=200**

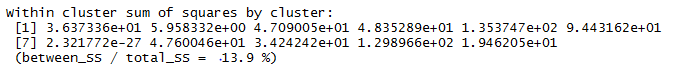




As shown above all documents are divided into 2 clusters and percentage of within cluster sum of square by cluster is 3.1%.

**Yelp Dataset / d=200**





As shown above all documents are divided into 11 clusters and percentage of within cluster sum of square by cluster is 13.9%.

3. **Compute the LDA representation for the documents**

**20Newsgroup**

**Command:**

*ldaOut <-LDA(dtm,k)*

*k=10*

*ldaOut <-LDA(dtm,k)*

*ldaOut.topics <- as.matrix(topics(ldaOut))*

*ldaOut.terms <- as.matrix(terms(ldaOut,10))*

*topicProbabilities <- as.data.frame(ldaOut@gamma)*

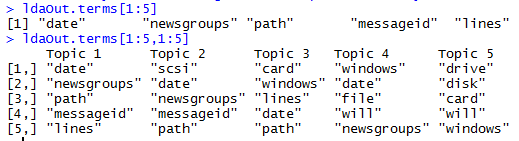
*topic1ToTopic2 <- lapply(1:nrow(dtm),function(x)sort(topicProbabilities[x,])[k]/sort(topicProbabilities[x,])[k-1])*

*topic2ToTopic3 <- lapply(1:nrow(dtm),function(x)sort(topicProbabilities[x,])[k-1]/sort(topicProbabilities[x,])[k-2])*

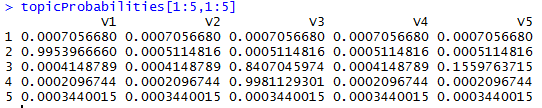
**Output:**

Perform LDA on document term matrix.

ldaout.terms find top 10 ten words in each topics.



*topicProbabilities calculate* probabilities associated with each topic assignment.

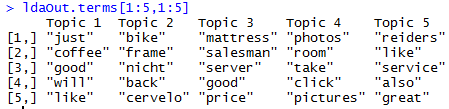


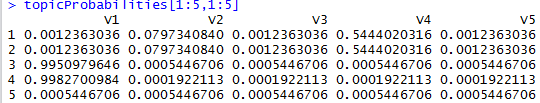
**Yelp Dataset**

**Output:**

Perform LDA on document term matrix.

ldaout.terms find top 10 ten words in each topics.





**4.**

**Compare clustering results**

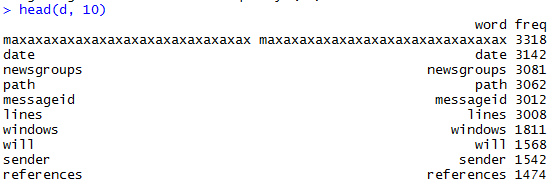
|  |  |  |
| --- | --- | --- |
|  | **20Newsgroup Dataset**  Within cluster sum of square by cluster | **Yelp Dataset**  Within cluster sum of square by cluster |
|  |  |  |
| **Tf-idf** | 0.6 | 23.8 |
| **d=50** | 7.4 | 23.2 |
| **d=100** | 4.8 | 14.3 |
| **d=200** | 3.1 | 13.9 |

As a result, we observed that for Tf-idf we get less SSE. For LSA SSE value change with respect to value of d change. If value of d increase, SSE value decease. For 20Newsgroup dataset we get 0.6% as SSE while for d=50,100,200 we get 7.4%,4.8%,3.1% SSE respectively. Similarly, for Yelp dataset we 23.8% as SSE while for d=50,100,200 we get 23.2%,14.3%,13.9% SSE respectively. Therefore, SSE decrease when value of d increase.

**II.C Results Summary**

**20Newsgroup Dataset**

For this dataset, we get more frequent words in process documents are date, newsgroup, path, messageid, lines, windows, sender etc. Following fig. represent most frequent word in process document with their frequency. (Total process 3000 documents)

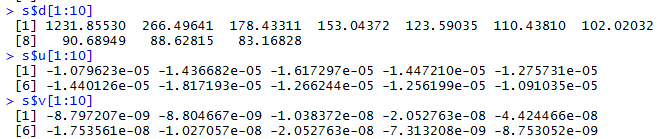


Using NbClust we find best number of cluster which is 2 for this dataset.

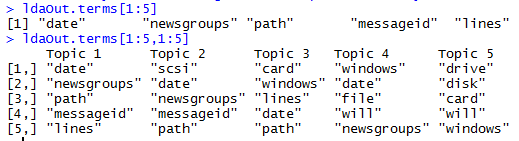
We perform kmeans for this dataset. We get words like card, windows, monitor, video etc. in clusetr1 while words like scsi, drive, hard disk, tape, backup etc. in cluster2. Following fig. represent classification of words in 2 clusters. (Total process 600 documents)



After performing SVD on document terms matrix we get the following result in which u and v are orthonormal i.e. orthogonal unit vector which can be represented as LSA document vector and LSA word vectors respectively and d is diagonal matrix which is square roots of the non-zero eigenvalues (u,v).

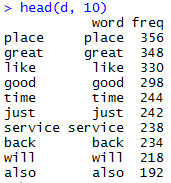


After performing LDA on this dataset we get following terms in different topics. We observed that some terms are common in some topics. Word like newsgroup find in topic 1,2 and 4.



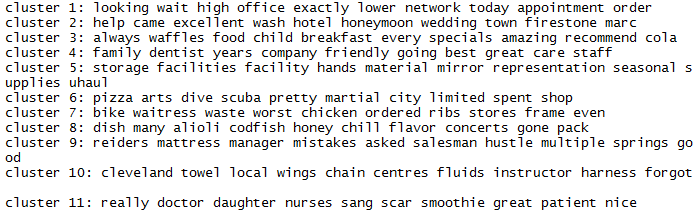
**Yelp Dataset**

For this dataset, we get more frequent words in process documents are great, like, good, service etc. Following fig. represent most frequent word in process document with their frequency. (Total process 736 documents)



Using NbClust we find best number of cluster which is 11 for this dataset.

We perform kmeans for this dataset. We get words like wait, lower, high etc. in clusetr1 while words like help, excellent, wedding etc. in cluster2. Following fig. represent classification of words in 11 clusters. (Total process 736 documents)



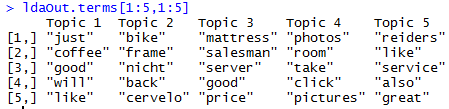
After performing SVD on document terms matrix we get the following result in which u and v are orthonormal i.e. orthogonal unit vector which can be represented as LSA document vector and LSA word vectors respectively and d is diagonal matrix which is square roots of the non-zero eigenvalues (u,v).







After performing LDA on this dataset we get following terms in different topics.



**III Analysis**

In Latent Semantic Analysis, we create the words by documents matrix. Before generating this matrix, we first remove unnecessary white space, number, punctuation, convert the text to lower case, remove common stop words like ‘the’, “we”. In this matrix, each row represents words while each column represent document. Each cell contains the number of times that word occurs in that title. Matrix build during LSA tend to be very large, but also very sparse i.e. most cells contain 0. As a result, document contains only a small number of all the possible words.

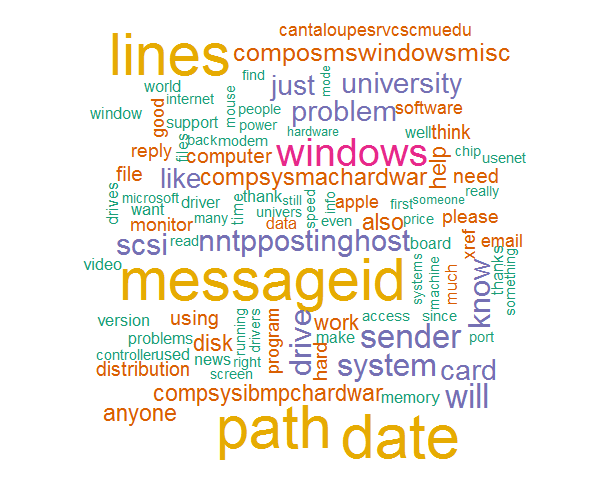
Once all documents are parsed, all the words that are in more than 1 documents are extracted and sorted, and a matrix is built with the number of rows equal to number of words(keys), and the number of columns equal to the document count.

Using TF-IDF, weight a matrix by term frequency - inverse document frequency.

From SVD we can find the value of d,u,v where u and v are orthonormal i.e. orthogonal unit vector which can be represented as LSA document vector and LSA word vectors respectively and d is diagonal matrix. The columns of u and v are eigenvector of XXT and XTX respectively. The entries in d are the square roots of the non-zero eigenvalues of XXT and XTX.

**20Neswgroup Dataset**

For this dataset, we process 3 newsgroups which contains total 3000 documents and we get following word cloud which represent most frequent words in all process documents.



This all words are related to all processing documents.

Using NbClust we find best number of cluster 2 and we perform kmeans algorithm using k value 2. Following words classification, we get for 2 clusters.



We observed that most of the words are common in result of word cloud and in cluster classification.

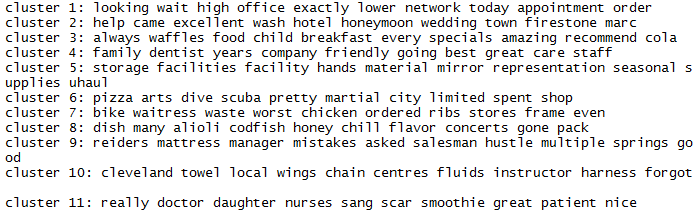
**Yelp Dataset**

For this dataset, we process 736 documents and we get following word cloud which represent most frequent words in all process documents.



This all words are related to all processing documents.

Using NbClust we find best number of cluster 11 and we perform kmeans algorithm using k value 11. Following words classification, we get for 11 clusters.



We observed that most of the words are common in result of word cloud and in cluster classification.