ROHAN DIGAMBAR GAWADE
A20379951

Text Analysis and Topic Modelling on 20 News Group and Yelp Dataset

I.

NewsGroup:

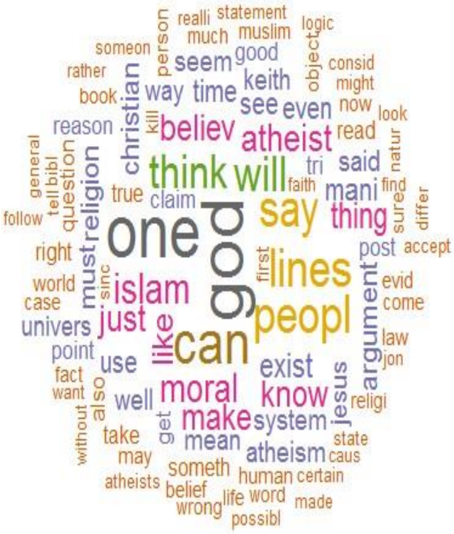
- The 20 newsgroup dataset consist of different topics related to religion, automobiles, sports, computers, science, electronics, politics.
- Each group has around 1000, which contains an article or mail or message related to the topic/group in which the file is present.
- Each file in a group/topics has words that relate us with the topic.
- For example: File 57110 consist of words like – osteoporosis, disease, glucocerebroside, bones, which suggest us that the file can be related to the sci.med topic.
- Each document in a topic has its own importance with respect to the context and for whom the article is written. But on a wider perspective each of the document implies about the topic to which it is belongs
- There are topics that are related to each other for example: ‘talk.religion.misc’, ‘alt.atheism’, ‘soc.religion.christian’ are related as they can be congregated to a single topic “religion.
- There can be more related semantic groups:

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

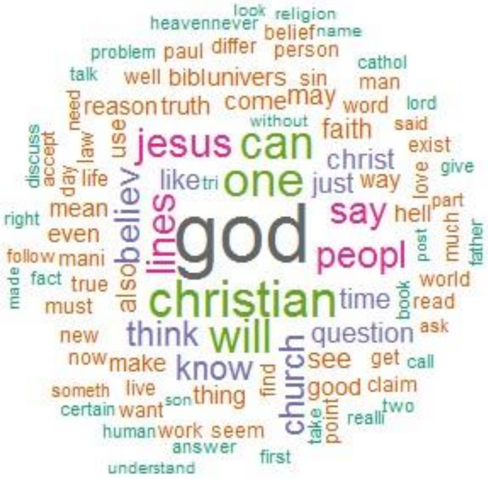
- In all there are 6 semantic groups that can be formed from the 20 NewsGroup Data.

Example :- WordCloud for Semantic Group - Religion

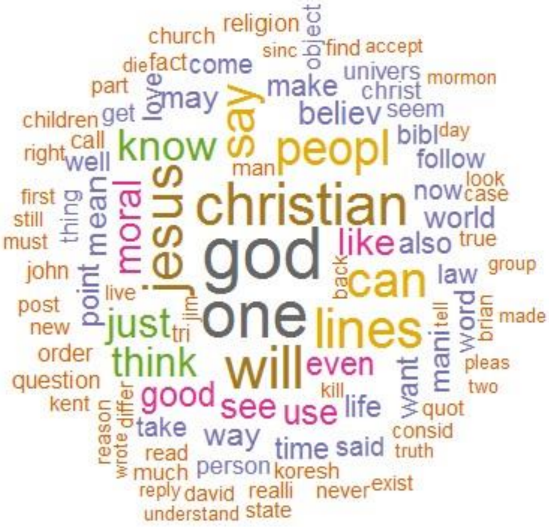
alt.atheism



soc.religion.christian



talk.religion.misc



- As we can see from the word cloud of these groups, words like god, christian, people, moral, believe etc are most frequent words among these groups. Hence we can say that they belong to a semantic group.
- Similarly the other semantic groups have frequent words which are similar.

20 News Group Overview

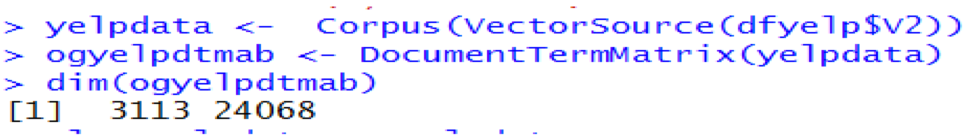
Newsgroup	**Number of Documents**	**Number of Unique Words**
alt.atheism	1000	17470
comp.graphics	1000	19673
comp.os.ms-windows.misc	1000	35733
comp.sys.ibm.pc.hardware	1000	15026
comp.sys.mac.hardware	1000	14404
comp.windows.x	1000	23983
misc.forsale	1000	16142
rec.autos	1000	16142
rec.motorcycles	1000	15320
rec.sport.baseball	1000	14528
rec.sport.hockey	1000	16656
sci.crypt	1000	19162
sci.electronics	1000	16163
sci.med	1000	22328
sci.space	1000	20423
soc.religion.christian	997	18323
talk.politics.guns	1000	19323
talk.politics.mideast	1000	23023
talk.politics.misc	1000	21804
talk.religion.misc	1000	19482

- This experiment is about analyzing how the documents are clustered based on the unique words using kmeans, lda and lsa algorithms.
- The number of unique words that we got here contains unwanted words as well. Stop words, headers, infrequent, very frequent words as well.
- We used tm package to pruned this words. Pruning is done by removing stopwords, removing punctuations, stemming, and taking only important frequent terms.
- Using this pruned unique words we will be clustering the documents and also will execute lda and lsa algorithms to view the document and term representation of this algorithms.

YELP

- The yelp dataset consist of different businesses reviews that customers give on yelp. It has ratings, reviews, votes.
- In this experiment, we need to analyze the reviews given by customers on different businesses.

- The yelp dataset consists of around 4.5GB of data.
- The yelp data is provided in JSON format, we need to convert the two json files – ‘yelp_academic_dataset_review.json’ and ‘yelp_academic_dataset_business.json’ into csv format and use the business-id and reviews that we get after merging these files for our experiment.
- The business.csv file has two columns
  1. business id
  2. categories
- Reviews can be grouped into different thematic groups. There are different groups like Restaurants, Cusines, Hotels etc. Few reviews correspond to these themes while few reviews are obscure to the group.
- I filtered the dataset using location = ‘Concord’
- I got around 3113 rows, which consisted of 181 categories,
- There are in all 24068 words.



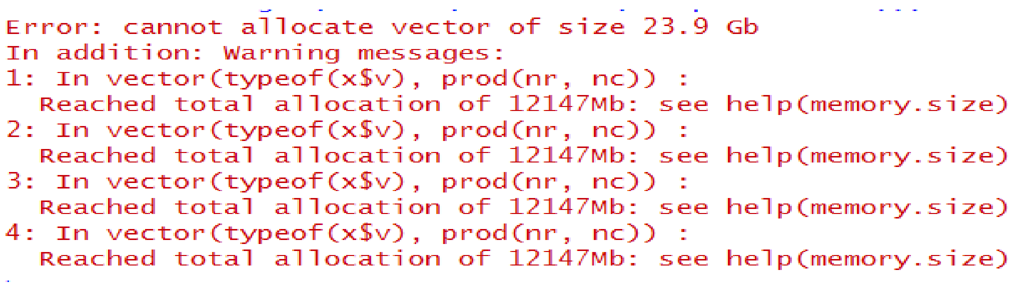
```
> yelpdata <-  Corpus(VectorSource(dfyelp$V2))
> ogyelpdtmab <- DocumentTermMatrix(yelpdata)
> dim(ogyelpdtmab)
[1]  3113 24068
```

II. A. DATA PREPROCESSING

NEWSGROUP

- Executed the preprocessing steps for all 20 newsgroups, but was not able to process this many files and words for clustering algorithm like kmeans and nbclust() due to size limitation.

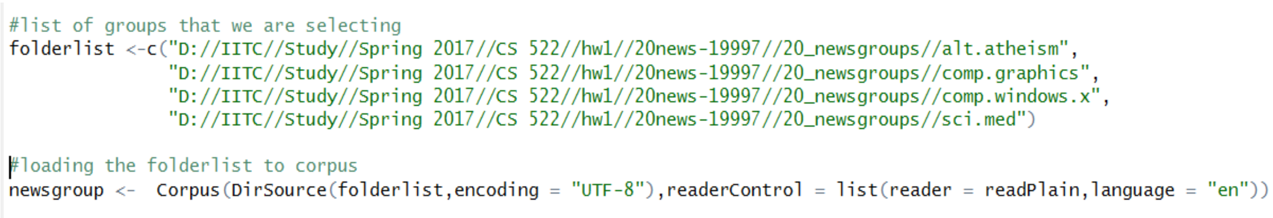
- Output Error



```
Error: cannot allocate vector of size 23.9 Gb
In addition: Warning messages:
1: In vector(typeof(x$v), prod(nr, nc)) :
  Reached total allocation of 12147Mb: see help(memory.size)
2: In vector(typeof(x$v), prod(nr, nc)) :
  Reached total allocation of 12147Mb: see help(memory.size)
3: In vector(typeof(x$v), prod(nr, nc)) :
  Reached total allocation of 12147Mb: see help(memory.size)
4: In vector(typeof(x$v), prod(nr, nc)) :
  Reached total allocation of 12147Mb: see help(memory.size)
```

- Hence, performed the analysis on four Newsgroup due to memory size limitation.
- The four folders that I selected for performing this experiment are:
  1. alt.atheism
  2. comp.graphics
  3. comp.windows.x
  4. sci.med

Code Snippet

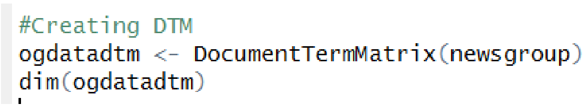


```
#list of groups that we are selecting
folderlist <-c("D://IITC//Study//Spring 2017//CS 522//hw1//20news-19997//20_newsgroups//alt.atheism",
               "D://IITC//Study//Spring 2017//CS 522//hw1//20news-19997//20_newsgroups//comp.graphics",
               "D://IITC//Study//Spring 2017//CS 522//hw1//20news-19997//20_newsgroups//comp.windows.x",
               "D://IITC//Study//Spring 2017//CS 522//hw1//20news-19997//20_newsgroups//sci.med")

#loading the folderlist to corpus
newsgroup <-  Corpus(DirSource(folderlist,encoding = "UTF-8"),readerControl = list(reader = readPlain,language = "en"))
```

SIMILARITY/DISSIMILARITY OF TOPICS PICKED

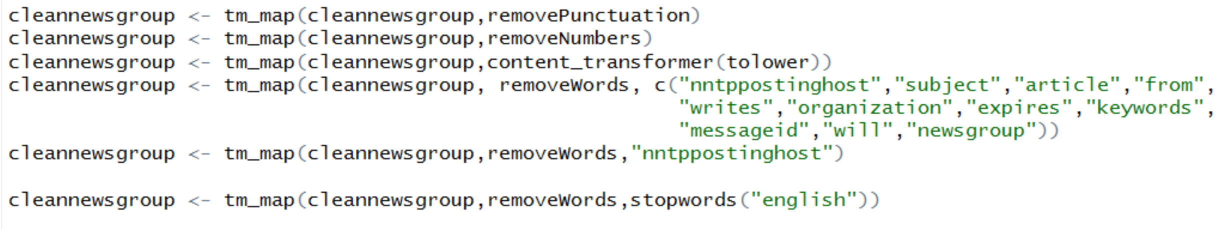
- The above four topics were picked for analysis.
- The topics ‘comp.graphics’ and ‘comp.windows.x’ seems similar as they belong to semantic group “Computers”.
- The other groups “alt.atheism”, ”sci.med” seems dissimilar.
- Using this groups ideally should yield us with the formation of 3 clusters through kmeans clustering algorithm. One cluster should form for the two computer topics, while the other two topics should be in separate clusters.
- LSA and LDA will perform better than kmeans clustering because documents will be related to the topics based on the words that we get using Document Term Matrix.

- **Create Document Term Matrix.**
  - Document Term Matrix describes the term frequency with respect to the documents.
  - Rows represents the collection of documents and columns represents terms.
  - The below code snippet takes the Corpus newsgroup and converts it into Document Term Matrix and stores it in ogdatadtm variable.



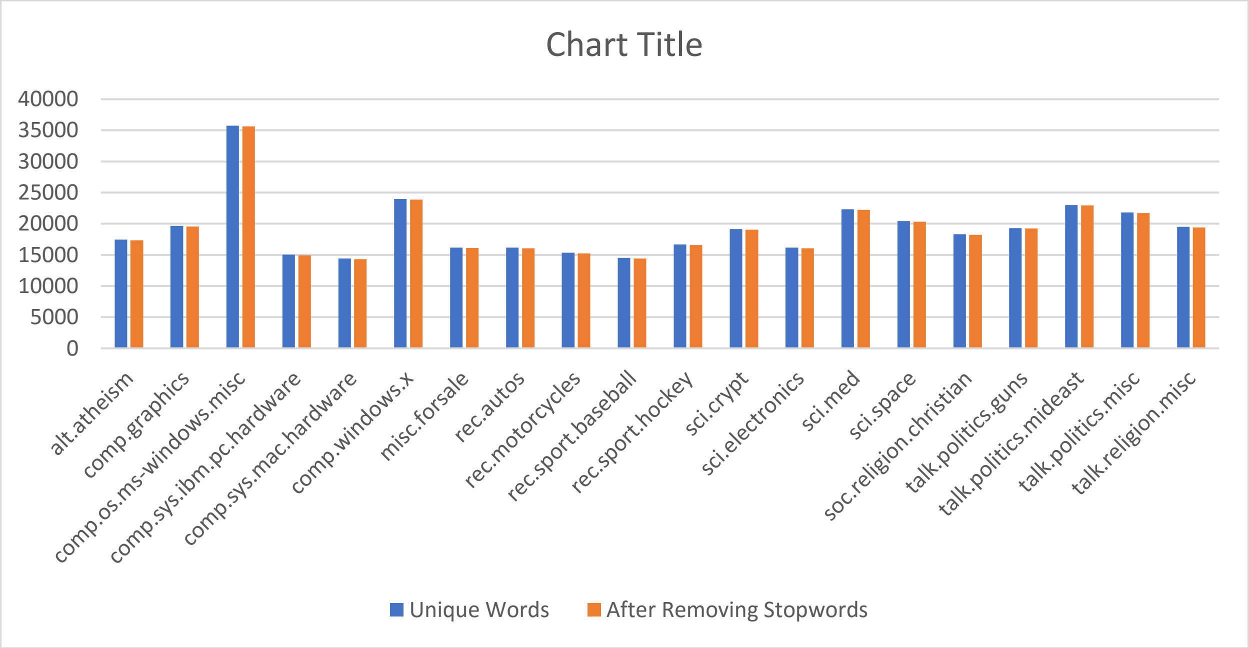
```
#Creating DTM
ogdatadtm <- DocumentTermMatrix(newsgroup)
dim(ogdatadtm)
```

- **Remove Stop words**
  - Stop words are words which even though very frequent, don’t have significance.
  - These words needs to be removed or filtered out so that only important words can be processed.
  - The stopwords can be removed using ‘tm’ package.



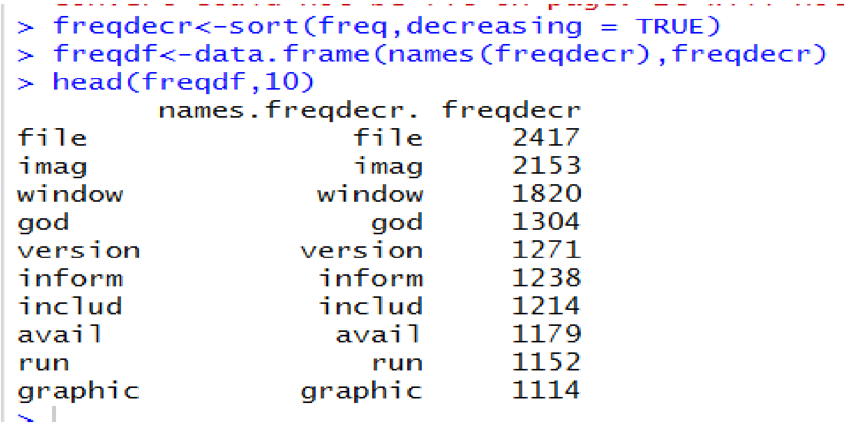
```
cleannewsgroup <- tm_map(cleannewsgroup,removePunctuation)
cleannewsgroup <- tm_map(cleannewsgroup,removeNumbers)
cleannewsgroup <- tm_map(cleannewsgroup,content_transformer(tolower))
cleannewsgroup <- tm_map(cleannewsgroup, removeWords, c("nntppostinghost","subject","article","from",
                                                         "writes","organization","expires","keywords",
                                                         "messageid","will","newsgroup"))
cleannewsgroup <- tm_map(cleannewsgroup,removeWords,"nntppostinghost")

cleannewsgroup <- tm_map(cleannewsgroup,removeWords,stopwords("english"))
```

  - The above commands removes and cleans the newsgroup word list and also removes stop words like – the, that, is, am, etc.
  - Hardly 0.5% of the original unique words got reduced.



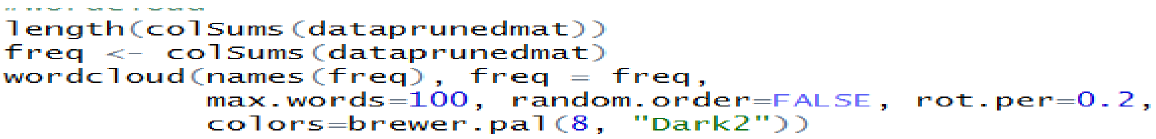
- **Count of word occurence**
  - Word occurrence count can be calculated using Document Term Matrix or Term Document Matrix.
  - If we are using Document Term Matrix we will be using colSums(dtm) as terms are spread across columns in DTM.

- ○ If we are using Term Document Matrix we will be using rowSums(tdm) as terms are spread across rows in TDM.
- ○ The count of word occurrence will tell us how many times the word appeared in the corpus.
- ○ The snippet to calculate the word count using DTM is:



```
> freqdecr<-sort(freq,decreasing = TRUE)
> freqdf<-data.frame(names(freqdecr),freqdecr)
> head(freqdf,10)
        names.freqdecr. freqdecr
file               file     2417
imag               imag     2153
window           window     1820
god                 god     1304
version         version     1271
inform           inform     1238
includ           includ     1214
avail             avail     1179
run                 run     1152
graphic         graphic     1114
>
```

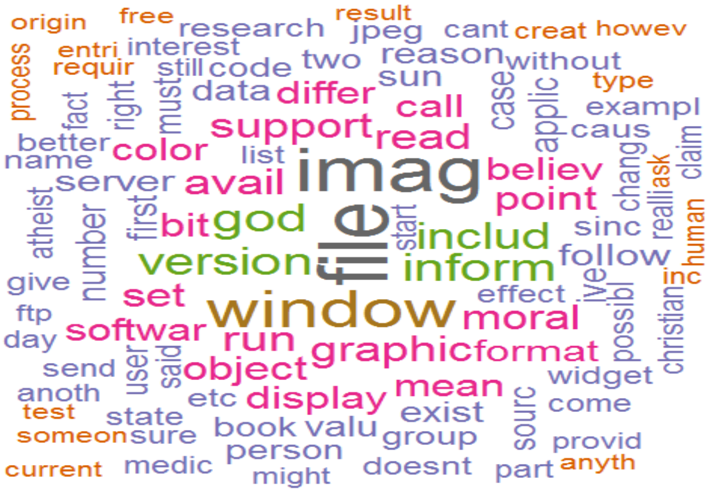
- ○ The frequency freq is computed to generate the wordcloud.



```
length(colSums(dataprunedmat))
freq <- colSums(dataprunedmat)
wordcloud(names(freq), freq = freq,
          max.words=100, random.order=FALSE, rot.per=0.2,
          colors=brewer.pal(8, "Dark2"))
```

.

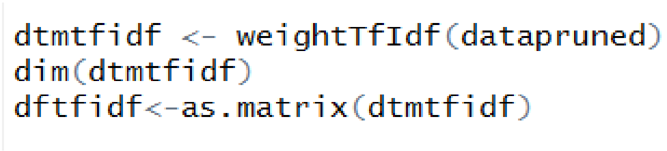
Word cloud generated



As we can see from the word cloud and the frequent terms displayed, both have frequent words.

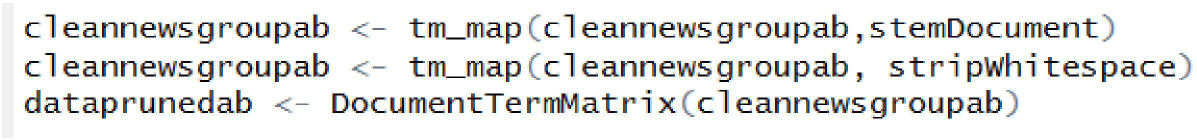
- **Using tf-idf**
- ○ The tf-idf is term frequency - inverse document frequency, shows how significant the word is to the document.

- The term frequency is a frequency of a term in a document, the inverse document frequency helps in diminishing the weight of the terms that occur very frequently and increases the weight of the significant terms that has less occurrence.

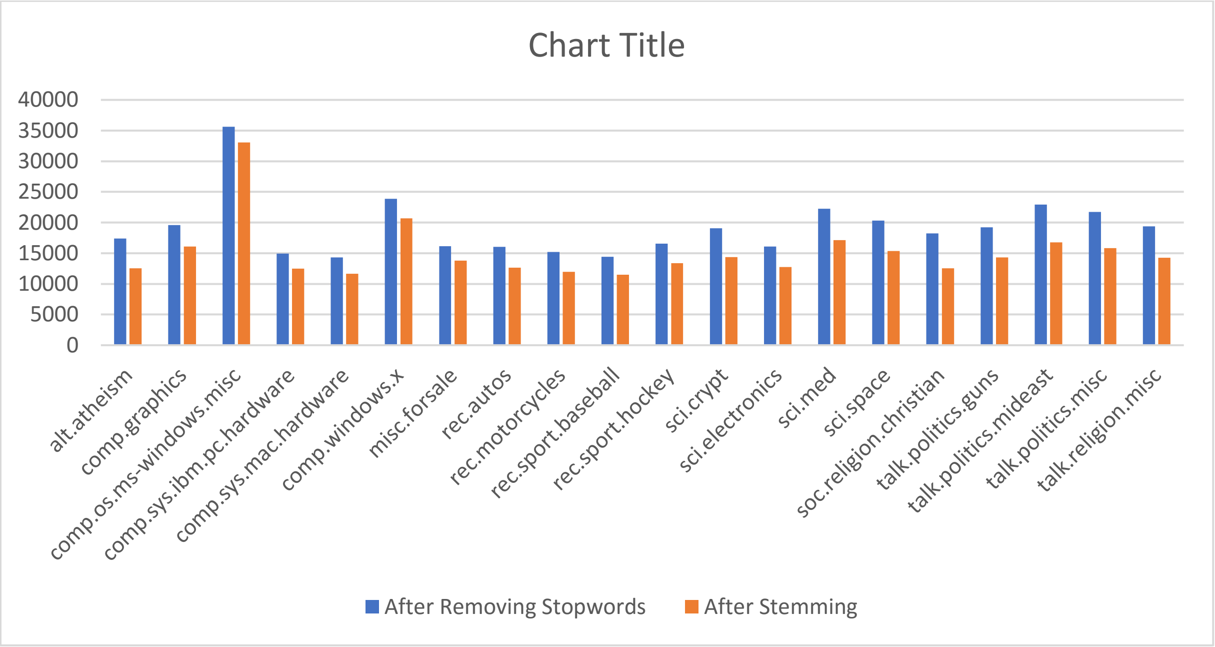


```
dtmtfidf <- weightTfIdf(datapruned)
dim(dtmtfidf)
dftfidf<-as.matrix(dtmtfidf)
```

- **Stemming**
  - Stemming is used to make the words which has same root into a root word.
  - The words like stemming, stemmed will be stem. So these words will be semantically mean one word, which will improve LDA, LSA and clustering process
  - Stemming can be done using stemDocument parameter to tm_map().



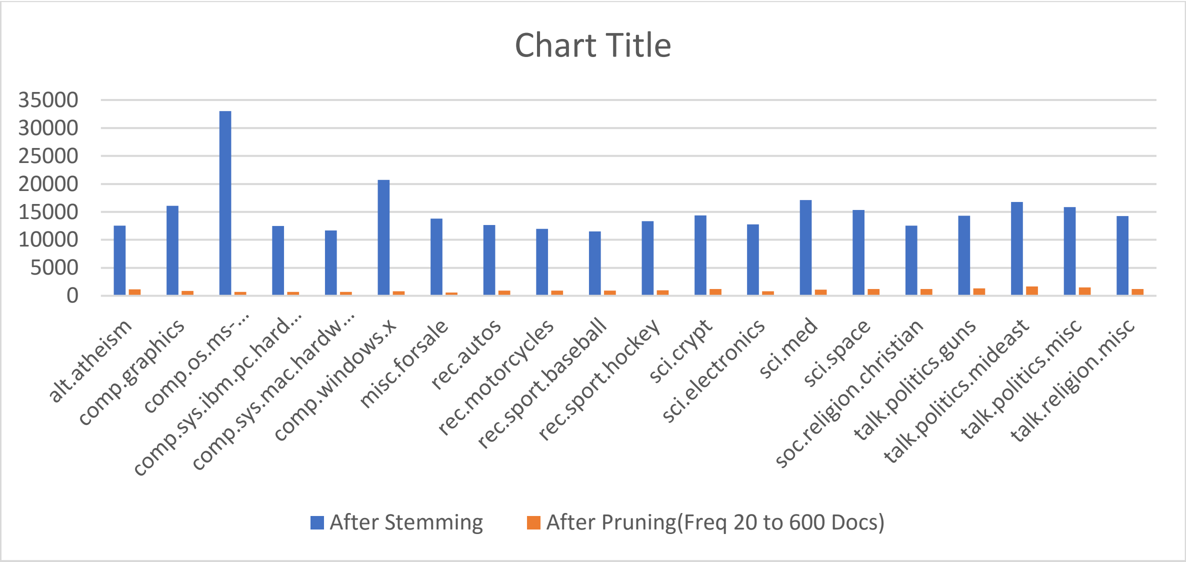
```
cleannewsgroupab <- tm_map(cleannewsgroupab,stemDocument)
cleannewsgroupab <- tm_map(cleannewsgroupab, stripWhitespace)
dataprunedab <- DocumentTermMatrix(cleannewsgroupab)
```



As we can see from the graph, the word count of the words after stemming discreased by approximately 20%.

- **Pruned words**
  - The words can be pruned using the Document Term Matrix function.

- We can pass lengths of the word, the frequencies of how much time the word is present in the document.
- It is also useful as it reduces the size of the document term matrix.
- The vocabulary size after pruning reduces immensely as we pruned based on frequency of the terms in the Document Term Matrix.

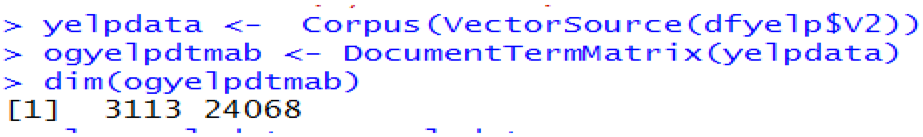


YELP

- Yelp dataset is very huge to process. Hence I filtered it using location = "Concord".
- The Yelp Dataset has similar groups which consist of restaurants and cuisines like American Traditional, Italian, Indian, Seafood, etc. It also consists of different catefories like Beauty and Spa, Event Planning & Services, Water Parks, Hotels & Travel, Hotels, Resorts, Amusement Parks, Active Life, etc.
- The performance of clustering on this dataset wouldn't be good, as though the dataset has wide range of categories, the reviews are mostly similar for them like a positive review, negative review. The reviews which perfectly describes the category are less compare to the generalized reviews.
- Even for LDA and LSA the terms will overlap with the topics. i.e many words can belong to many topics.

Document Term Matrix

- For Yelp Dataset is created similar to that for 20 Newsgroup. Since we are loading the corpus using csv file we use VectorSource().

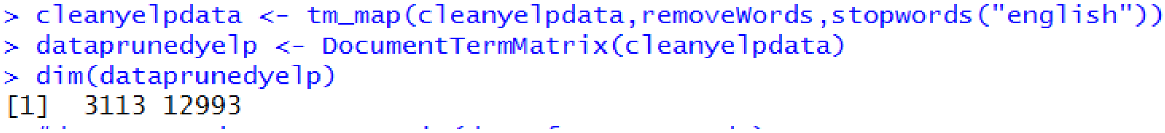


```
> yelpdata <-  Corpus(VectorSource(dfyelp$V2))
> ogyelpdtmab <- DocumentTermMatrix(yelpdata)
> dim(ogyelpdtmab)
[1]  3113 24068
```

- The above output displays the number of categories which are there for yelp dataset and the number of words in the data set.
- There are in all 3113 categories of which 181 are unique and there are around 24068 words

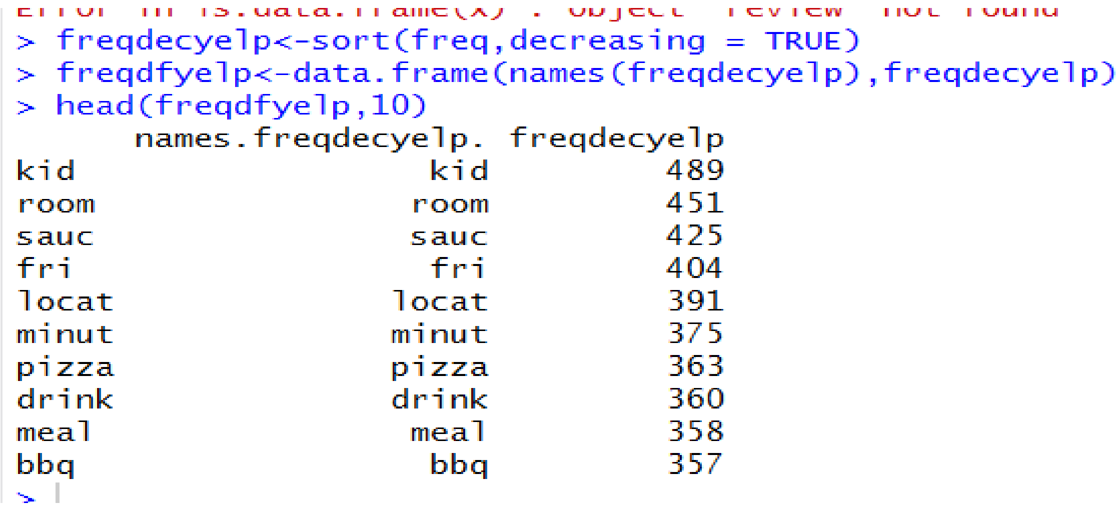
STOP WORDS

- After removing the stop words, the word count decreased by 49%.



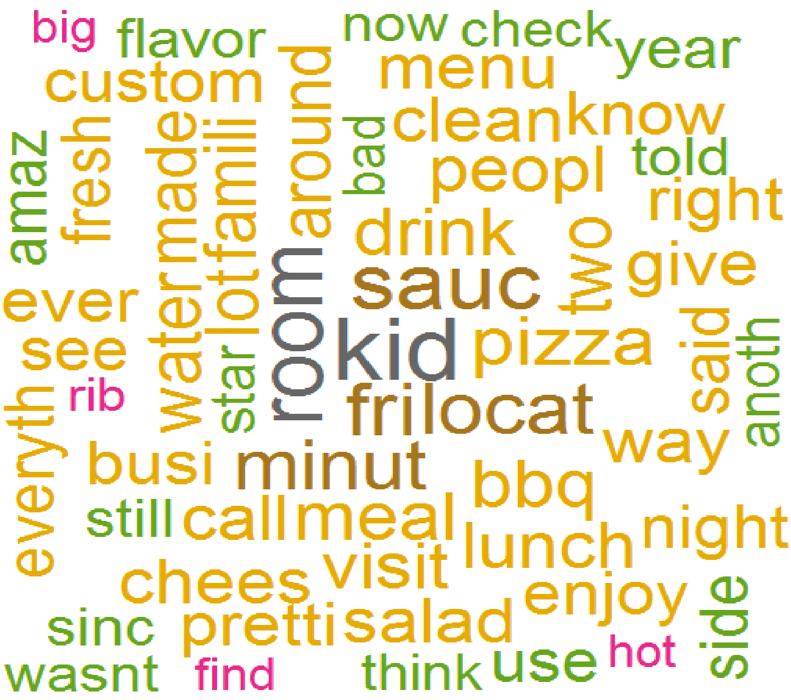
```
> cleanyelpdata <- tm_map(cleanyelpdata,removeWords,stopwords("english"))
> dataprunedyelp <- DocumentTermMatrix(cleanyelpdata)
> dim(dataprunedyelp)
[1]  3113 12993
```

COUNT OF WORD OCCURENCES



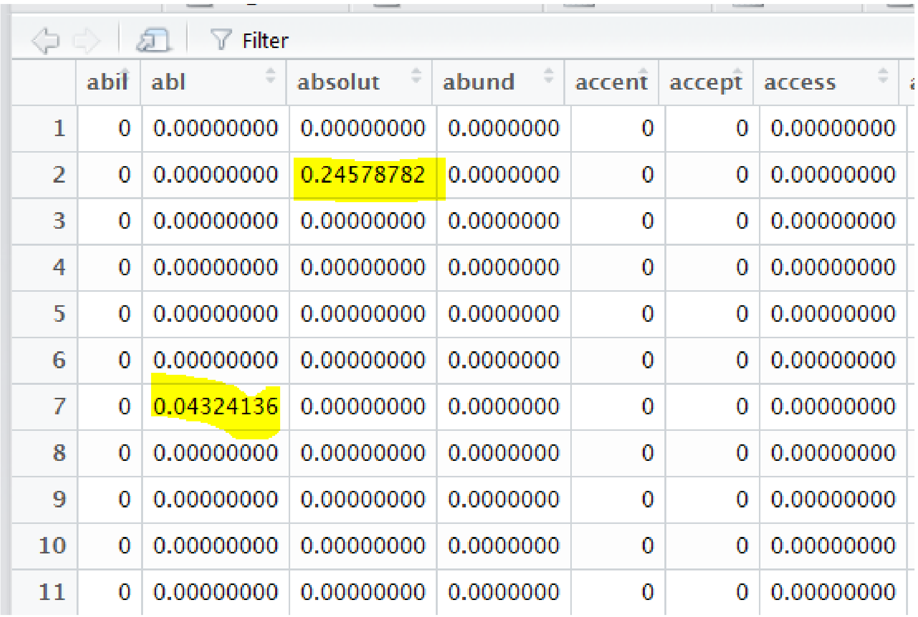
```
Error in is.data.frame(x) : object 'review' not found
> freqdecyelp<-sort(freq,decreasing = TRUE)
> freqdfyelp<-data.frame(names(freqdecyelp),freqdecyelp)
> head(freqdfyelp,10)
      names.freqdecyelp. freqdecyelp
kid                  kid         489
room                room         451
sauc                sauc         425
fri                  fri         404
locat              locat         391
minut              minut         375
pizza              pizza         363
drink              drink         360
meal                meal         358
bbq                  bbq         357
> 
```

Word cloud



TF-IDF

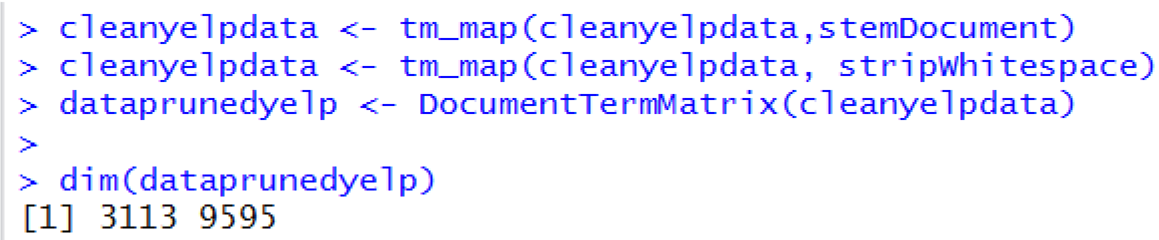
```
> dtmtfidfyelp <- weightTfIdf(dataprunedyelp)
> View(as.matrix(dtmtfidfyelp))
```



	abil	abl	absolut	abund	accent	accept	access
1	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000
2	0	0.00000000	0.24578782	0.0000000	0	0	0.00000000
3	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000
4	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000
5	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000
6	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000
7	0	0.04324136	0.00000000	0.0000000	0	0	0.00000000
8	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000
9	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000
10	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000
11	0	0.00000000	0.00000000	0.0000000	0	0	0.00000000

STEMMING

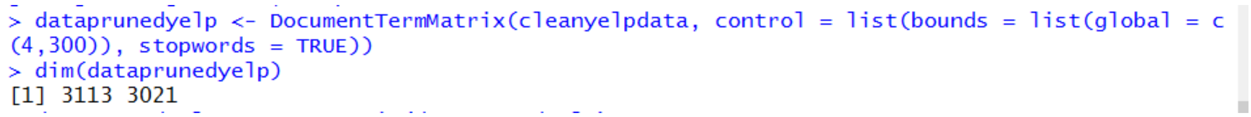
- After stemming the count of the words reduced from 12993 to 9596. i.e 26% reduction after stemming



```
> cleanyelpdata <- tm_map(cleanyelpdata,stemDocument)
> cleanyelpdata <- tm_map(cleanyelpdata, stripWhitespace)
> dataprunedyelp <- DocumentTermMatrix(cleanyelpdata)
>
> dim(dataprunedyelp)
[1] 3113 9595
```

PRUNNING

After pruning based on frequency of the words the word count reduced to 3021.



```
> dataprunedyelp <- DocumentTermMatrix(cleanyelpdata, control = list(bounds = list(global = c
(4,300)), stopwords = TRUE))
> dim(dataprunedyelp)
[1] 3113 3021
```

Analysis:

- The size of the vocabulary decreases at each step of data preprocessing.
- The reduction in vocabulary due to pruning gives us the most significant words.
- These pruned words improves the output of LDA and LSA algorithms.
- Even clustering of the words will be better as probability of the words belonging to only one cluster will increase.

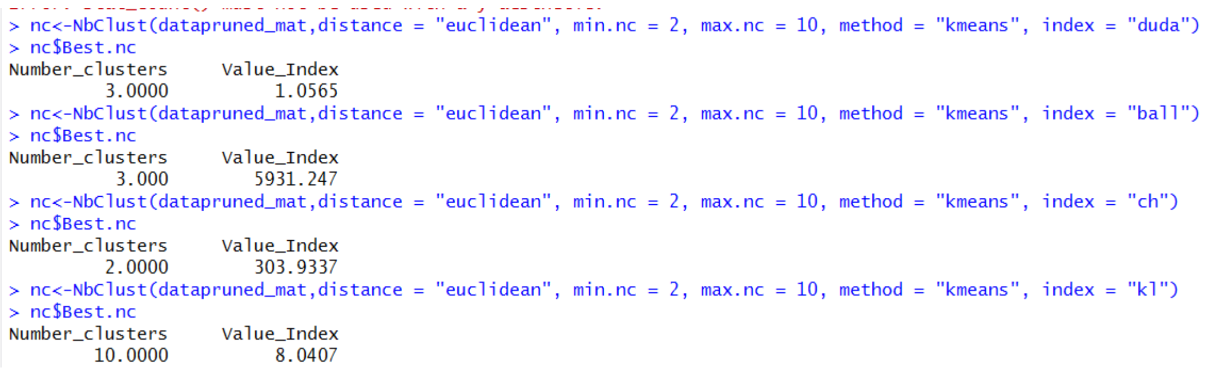
II. B

1.

CLUSTERING On NewsGroup

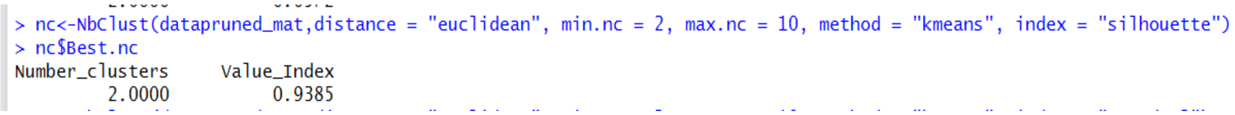
- The NbClust method is used to determine the number of clusters ‘k’ that we can use for kmeans algorithm.
- As we can see from the below snaps. 3 indexes-> duda, ball, cindex proposed 3 as best cluster.
- While 2 indexes -> ch, silhouette proposed 2 as best cluster.
- Few indexes were skipped due to the space required for compution and also taking very long time to compute.
- Hence I came up to the conclusion that k will be 2.

DUDA Index, BALL Index, CH Index, KL , Index



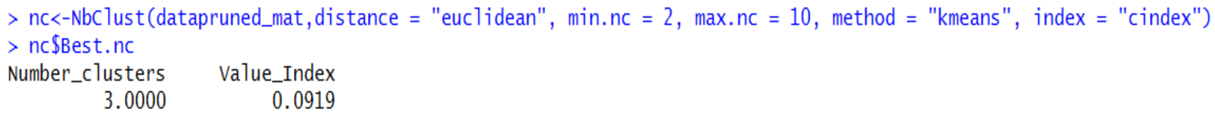
```
> nc<-NbClust(datapruned_mat,distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans", index = "duda")
> nc$Best.nc
Number_clusters     Value_Index
         3.0000          1.0565
> nc<-NbClust(datapruned_mat,distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans", index = "ball")
> nc$Best.nc
Number_clusters     Value_Index
          3.000        5931.247
> nc<-NbClust(datapruned_mat,distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans", index = "ch")
> nc$Best.nc
Number_clusters     Value_Index
         2.0000        303.9337
> nc<-NbClust(datapruned_mat,distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans", index = "kl")
> nc$Best.nc
Number_clusters     Value_Index
        10.0000          8.0407
```

SILHOUETTE Index



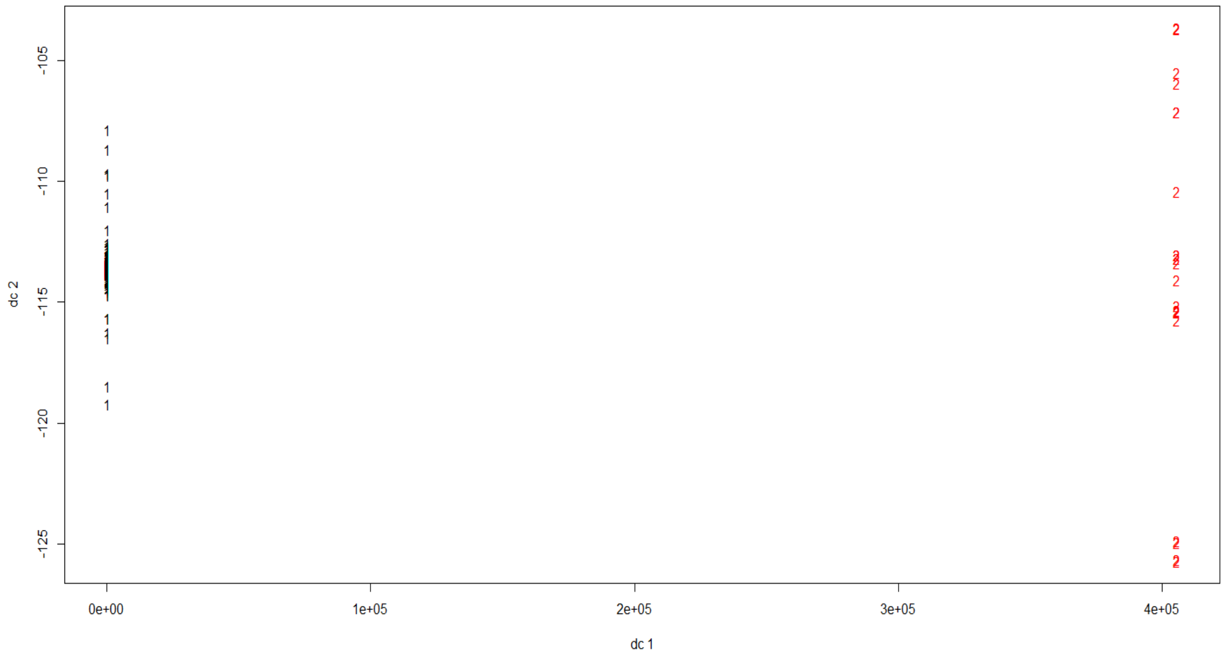
```
> nc<-NbClust(datapruned_mat,distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans", index = "silhouette")
> nc$Best.nc
Number_clusters     Value_Index
         2.0000          0.9385
```

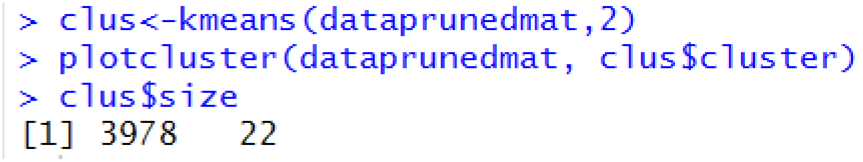
C Index



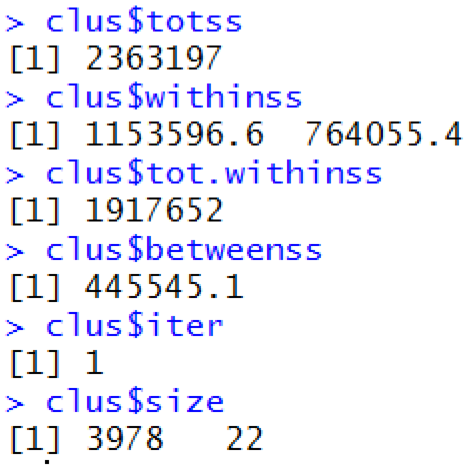
```
> nc<-NbClust(datapruned_mat,distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans", index = "cindex")
> nc$Best.nc
Number_clusters     Value_Index
         3.0000          0.0919
```

Cluster output





```
> clus<-kmeans(dataprunedmat,2)
> plotcluster(dataprunedmat, clus$cluster)
> clus$size
[1] 3978   22
```



```
> clus$totss
[1] 2363197
> clus$withinss
[1] 1153596.6  764055.4
> clus$tot.withinss
[1] 1917652
> clus$betweenss
[1] 445545.1
> clus$iter
[1] 1
> clus$size
[1] 3978   22
```

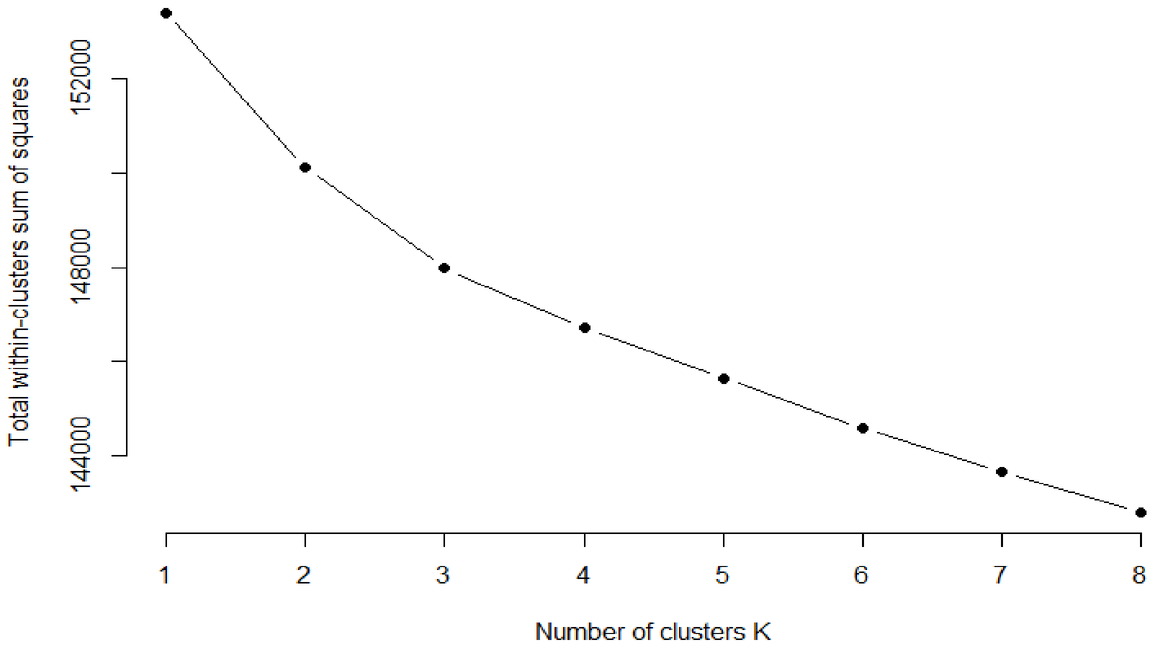
Analysis:

- As we can see the data is more biased towards cluster 1 than cluster 2.
- Ideally there should have been atleast 3 clusters. Two comp groups should be joined to form one cluster, while the rest should be two different clusters.
- But here since we are forming two clusters(using the best number of clusters via nbclust()).
- The data points should have been in such a way that atleast the three folders comp.graphics, comp.windows.x and also sci.med due to the technical terms in these folders while altheism forms a separate group. But since clustering is not done based on terms belonging to a topic, we get unbalanced cluster size.

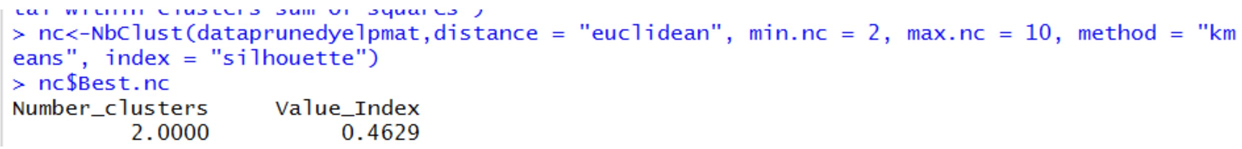
Clustering On YELP

- The NbClust method is used to determine the number of clusters 'k' that we can use for kmeans algorithm.
- As we can see from the below snaps. 3 indexes-> wss, ball, kl proposed 3 as best cluster.
- While 1 indexes -> silhouette proposed 2 as best cluster.
- Few indexes were skipped due to the space required for compution and also taking very long time to compute.
- Hence, I came up to the conclusion that k will be 3.

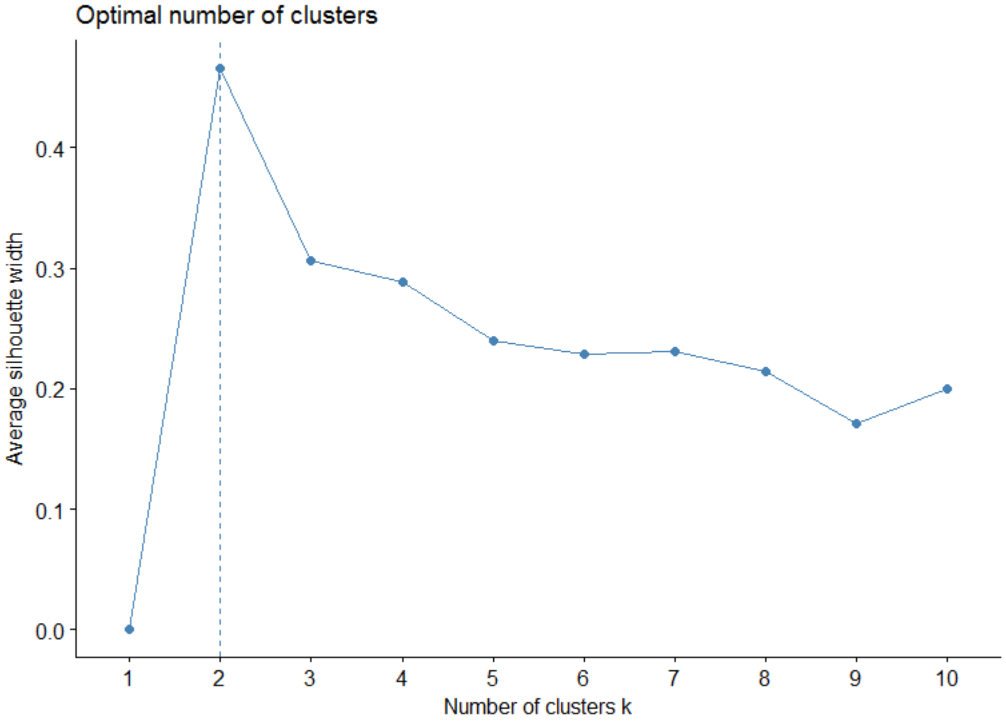
WSS Index



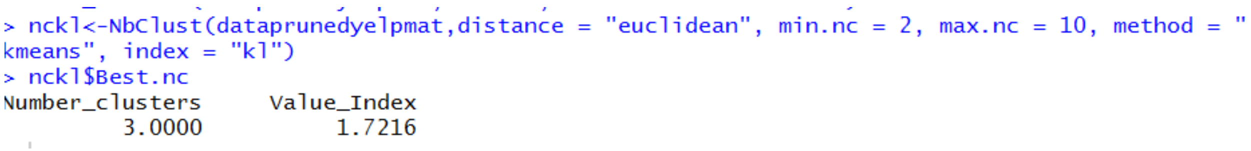
SILHOUETTE Index



```
tal within clusters sum of squares )
> nc<-NbClust(dataprunedyelpmat,distance = "euclidean", min.nc = 2, max.nc = 10, method = "km
eans", index = "silhouette")
> nc$Best.nc
Number_clusters     Value_Index
         2.0000          0.4629
```

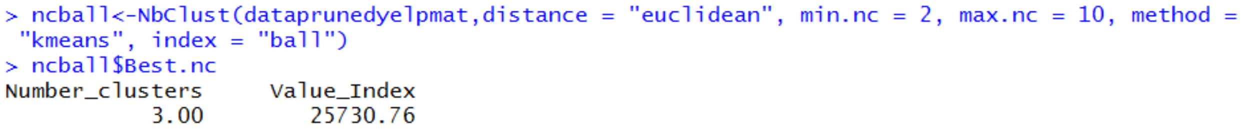


KL Index



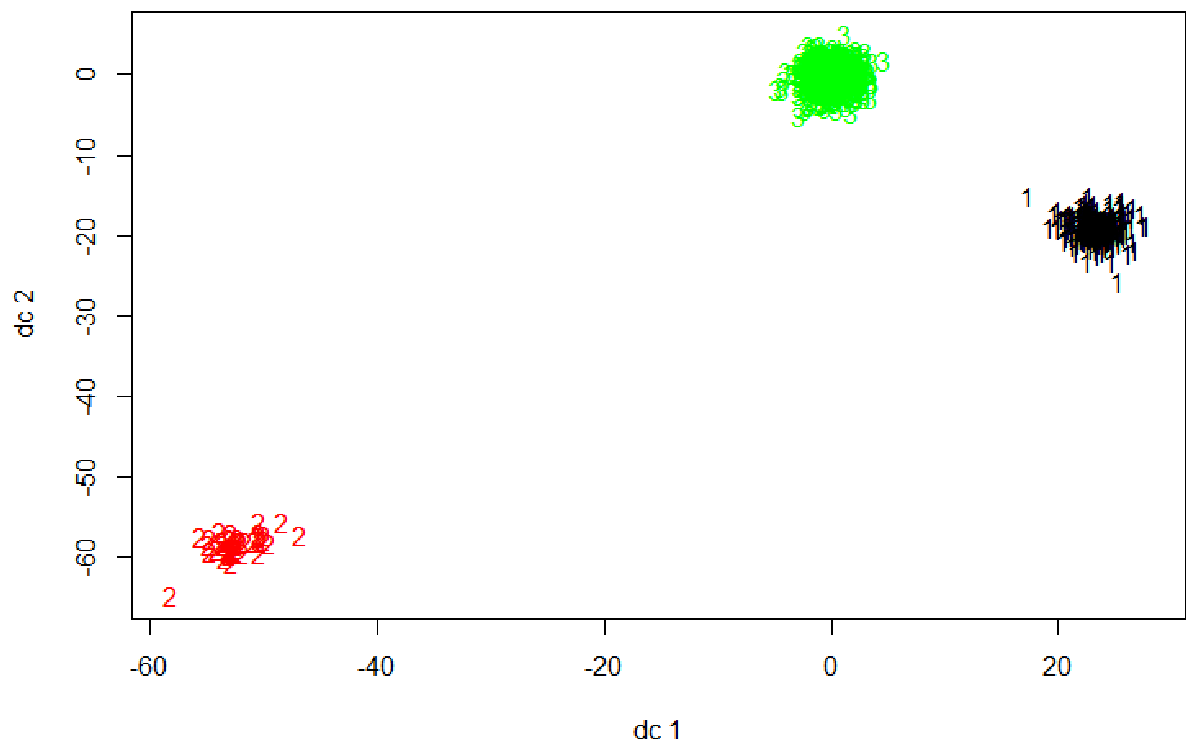
```
> nckl<-NbClust(dataprunedyelpmat,distance = "euclidean", min.nc = 2, max.nc = 10, method = "
kmeans", index = "kl")
> nckl$Best.nc
Number_clusters     Value_Index
         3.0000          1.7216
```

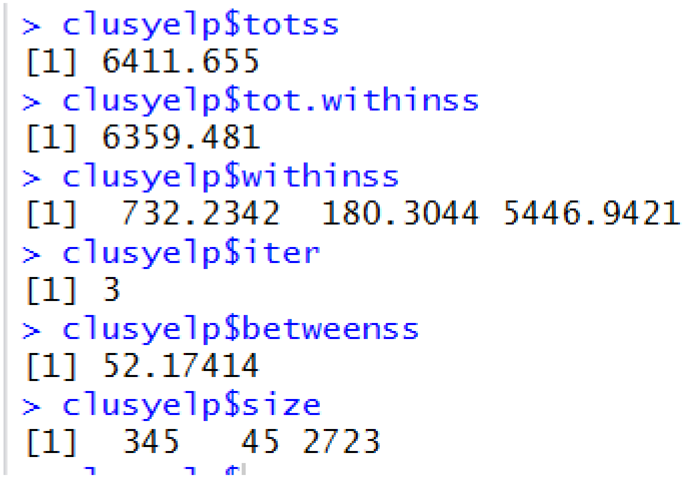
BALL Index



```
> ncball<-NbClust(dataprunedyelpmat,distance = "euclidean", min.nc = 2, max.nc = 10, method =
 "kmeans", index = "ball")
> ncball$Best.nc
Number_clusters     Value_Index
           3.00        25730.76
```

```
> clusyelp<-kmeans(dataprunedyelpmat,3)
> plotcluster(dataprunedyelpmat, clusyelp$cluster)
```

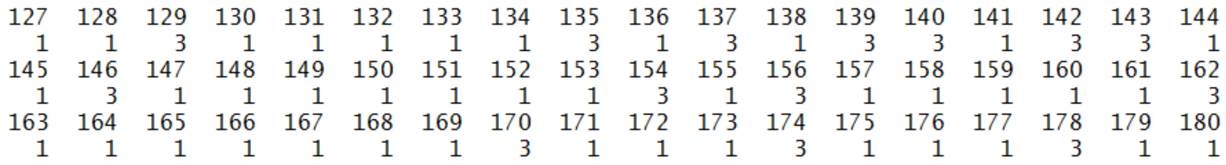




```
> clusyelp$totss
[1] 6411.655
> clusyelp$tot.withinss
[1] 6359.481
> clusyelp$withinss
[1]  732.2342  180.3044 5446.9421
> clusyelp$iter
[1] 3
> clusyelp$betweenss
[1] 52.17414
> clusyelp$size
[1]  345   45 2723
```

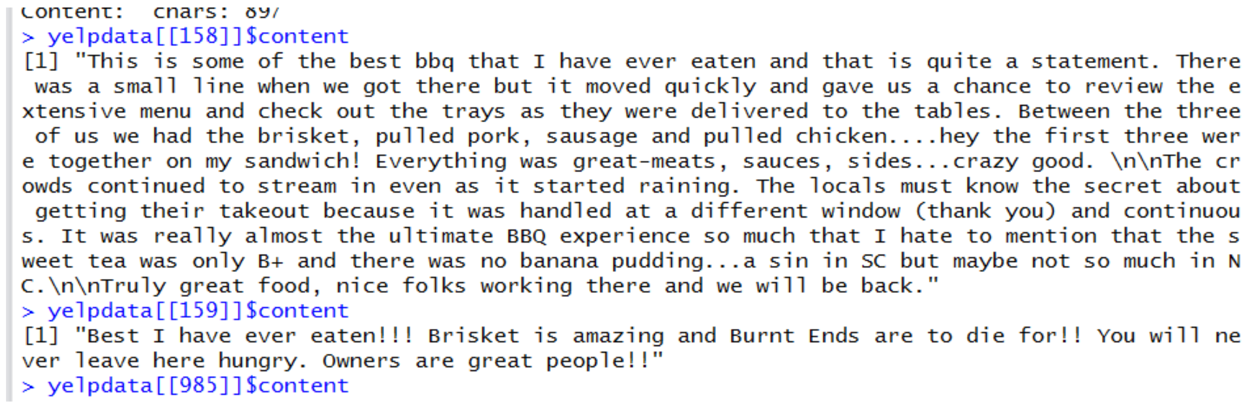
Analysis

- We selected 3 clusters based on nbclust method and different indexes given to Nbclust.
- Ideally we should have more than 100 clusters with respect to Yelp businesses but since the words are more generic.
- For the kmeans clustering, the words are grouped into three categories – positive reviews, negative reviews and neutral reviews
- The cluster of reviews for cluster 1 :



```
 127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144 
   1    1    3    1    1    1    1    1    3    1    3    1    3    3    1    3    3    1 
 145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162 
   1    3    1    1    1    1    1    1    1    3    1    3    1    1    1    1    1    3 
 163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180 
   1    1    1    1    1    1    1    3    1    1    1    3    1    1    1    3    1    1 
```

- So, if we compare the reviews of 158 and 159:



```
Content:  chars: 897
> yelpdata[[158]]$content
[1] "This is some of the best bbq that I have ever eaten and that is quite a statement. There
 was a small line when we got there but it moved quickly and gave us a chance to review the e
xtensive menu and check out the trays as they were delivered to the tables. Between the three
 of us we had the brisket, pulled pork, sausage and pulled chicken....hey the first three wer
e together on my sandwich! Everything was great-meats, sauces, sides...crazy good. \n\nThe cr
owds continued to stream in even as it started raining. The locals must know the secret about
 getting their takeout because it was handled at a different window (thank you) and continuou
s. It was really almost the ultimate BBQ experience so much that I hate to mention that the s
weet tea was only B+ and there was no banana pudding...a sin in SC but maybe not so much in N
C.\n\nTruly great food, nice folks working there and we will be back."
> yelpdata[[159]]$content
[1] "Best I have ever eaten!!! Brisket is amazing and Burnt Ends are to die for!! You will ne
ver leave here hungry. Owners are great people!!"
> yelpdata[[985]]$content
```

- We can see that the two reviews are positive, hence the reviews that belongs to cluster 1 are positive reviews.

II. B. 2

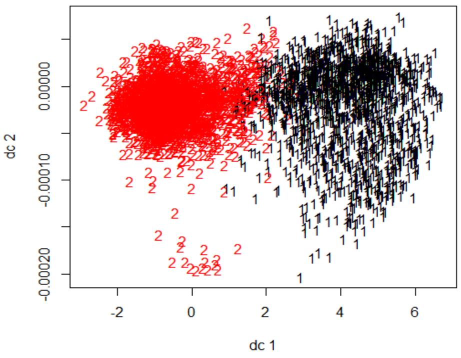
LSA

- **Compute SVD**
  svd_dtm<-svd(dtmprune)

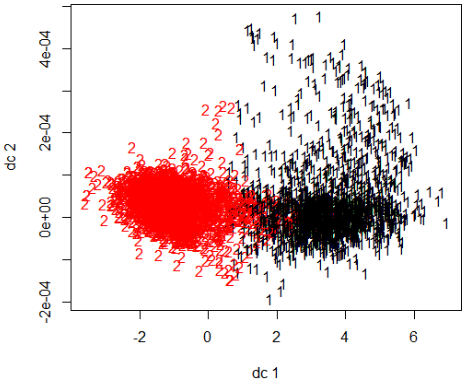
- SVD is used in LSA as reduced order matrix where rows corresponds to words and columns corresponds to documents
- The singular vector and values that the SVD computes maps the words and documents in the Latent Semantic space.
- The below plot shows that when we use svd, the hidden data is plotted as we add first few vector from svd to the input raw data.

CLUSTERING d dimension of SVD for NEWSGROUP

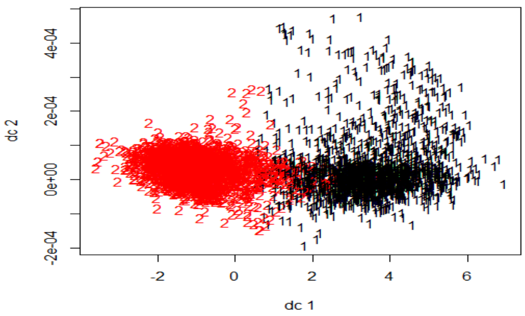
Dimension d = 50

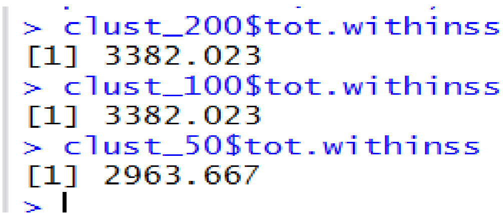


Dimension d = 100



Dimension d = 200





```
> clust_200$tot.withinss
[1] 3382.023
> clust_100$tot.withinss
[1] 3382.023
> clust_50$tot.withinss
[1] 2963.667
> 
```

CLUSTERING d dimension of SVD for Yelp

Analysis:

Topics in the Corpus	Cluster		SSE
alt.atheism comp.graphics comp.windows.x sci.med	2	50	2963.667
	2	100	3382.023
	2	200	3382.023

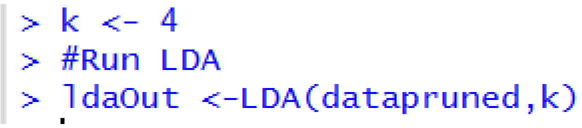
- As the dimensions increased the SSE value also increased for the two clusters
- As we can see that the SSE value of the dimension 50 is low hence we will choose dimension 50 for clustering.
- Also if we compare the plot of Kmeans clustering with the LSA we can see that SVD enhances the cluster plot.

II B

3. LDA

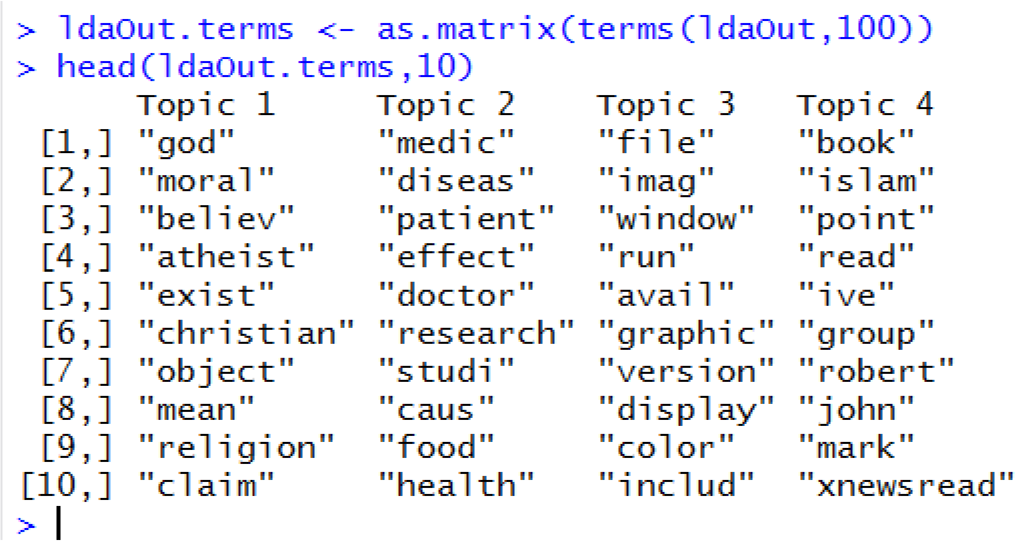
- LDA represents documents as mixtures over topics using the words/terms that belongs to that topic.
- LDA will classify words according to topics.

LDA On NewsGroups



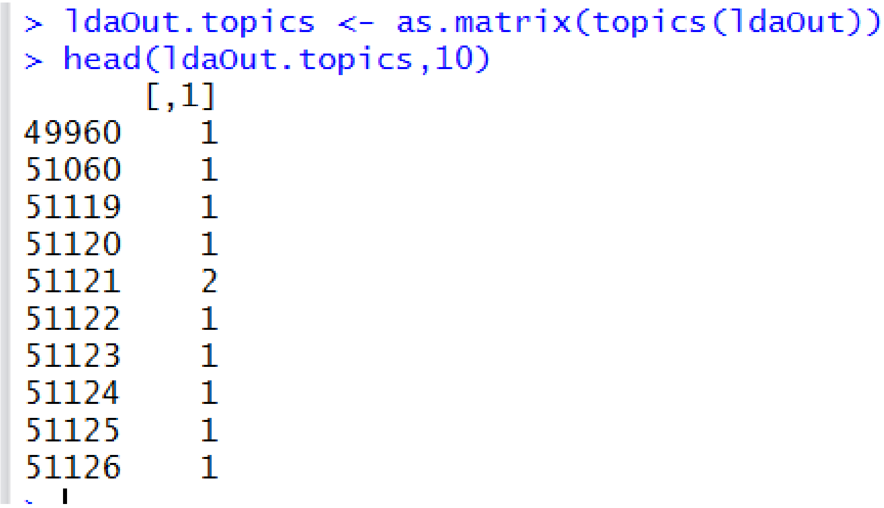
```
> k <- 4
> #Run LDA
> ldaOut <-LDA(datapruned,k)
```

Terms to Topics



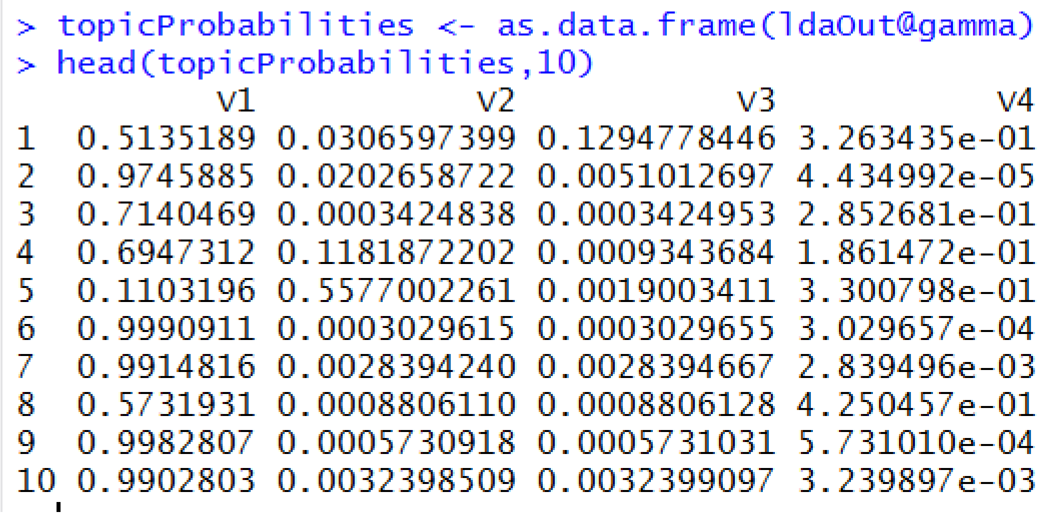
```
> ldaOut.terms <- as.matrix(terms(ldaOut,100))
> head(ldaOut.terms,10)
      Topic 1     Topic 2    Topic 3   Topic 4
 [1,] "god"       "medic"    "file"    "book"
 [2,] "moral"     "diseas"   "imag"    "islam"
 [3,] "believ"    "patient"  "window"  "point"
 [4,] "atheist"   "effect"   "run"     "read"
 [5,] "exist"     "doctor"   "avail"   "ive"
 [6,] "christian" "research" "graphic" "group"
 [7,] "object"    "studi"    "version" "robert"
 [8,] "mean"      "caus"     "display" "john"
 [9,] "religion"  "food"     "color"   "mark"
[10,] "claim"     "health"   "includ"  "xnewsread"
>
```

Documents To Topics



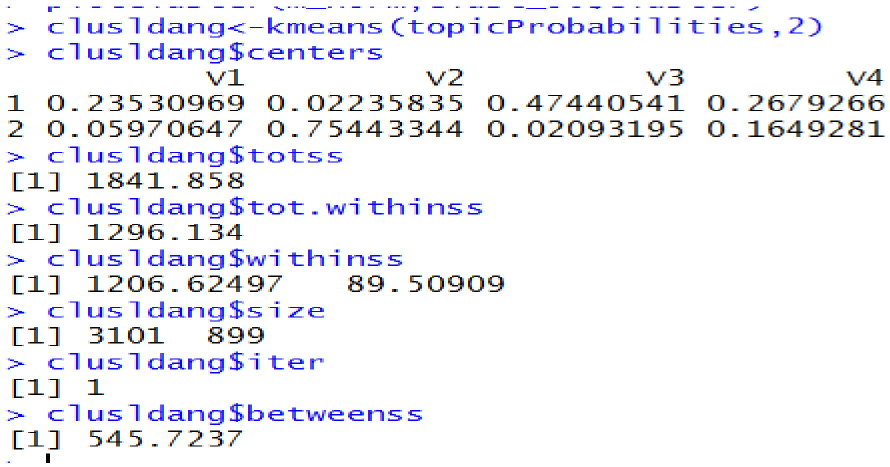
```
> ldaOut.topics <- as.matrix(topics(ldaOut))
> head(ldaOut.topics,10)
      [,1]
49960    1
51060    1
51119    1
51120    1
51121    2
51122    1
51123    1
51124    1
51125    1
51126    1
```

Topic Probabilities

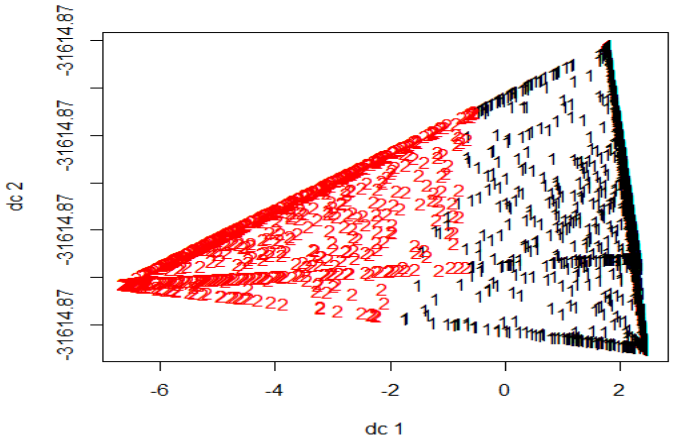


```
> topicProbabilities <- as.data.frame(ldaOut@gamma)
> head(topicProbabilities,10)
          V1           V2           V3           V4
1  0.5135189 0.0306597399 0.1294778446 3.263435e-01
2  0.9745885 0.0202658722 0.0051012697 4.434992e-05
3  0.7140469 0.0003424838 0.0003424953 2.852681e-01
4  0.6947312 0.1181872202 0.0009343684 1.861472e-01
5  0.1103196 0.5577002261 0.0019003411 3.300798e-01
6  0.9990911 0.0003029615 0.0003029655 3.029657e-04
7  0.9914816 0.0028394240 0.0028394667 2.839496e-03
8  0.5731931 0.0008806110 0.0008806128 4.250457e-01
9  0.9982807 0.0005730918 0.0005731031 5.731010e-04
10 0.9902803 0.0032398509 0.0032399097 3.239897e-03
```

Clustering on Topic Probability for Newsgroup

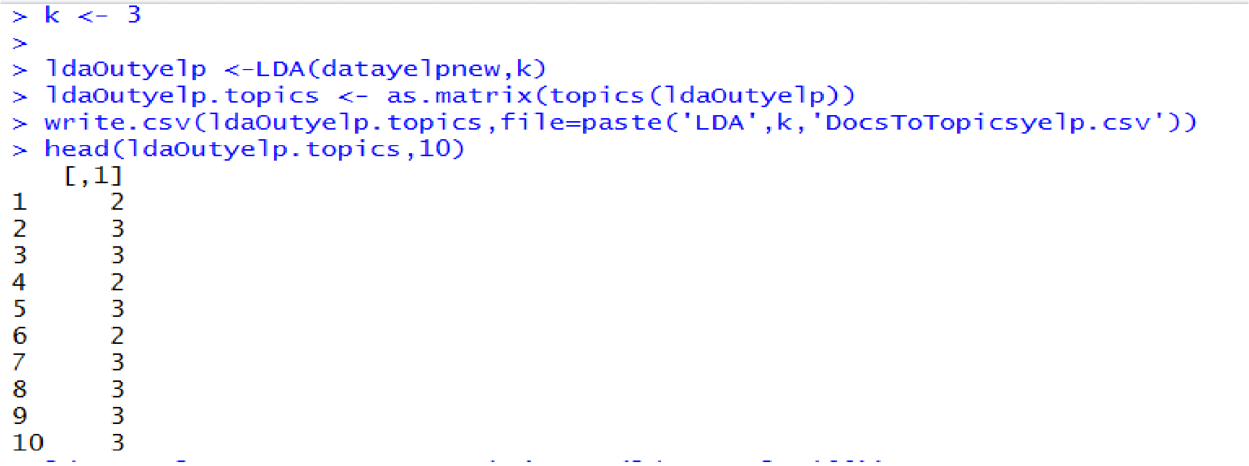


```
> clusldang<-kmeans(topicProbabilities,2)
> clusldang$centers
          V1         V2         V3        V4
1 0.23530969 0.02235835 0.47440541 0.2679266
2 0.05970647 0.75443344 0.02093195 0.1649281
> clusldang$totss
[1] 1841.858
> clusldang$tot.withinss
[1] 1296.134
> clusldang$withinss
[1] 1206.62497   89.50909
> clusldang$size
[1] 3101  899
> clusldang$iter
[1] 1
> clusldang$betweenss
[1] 545.7237
```



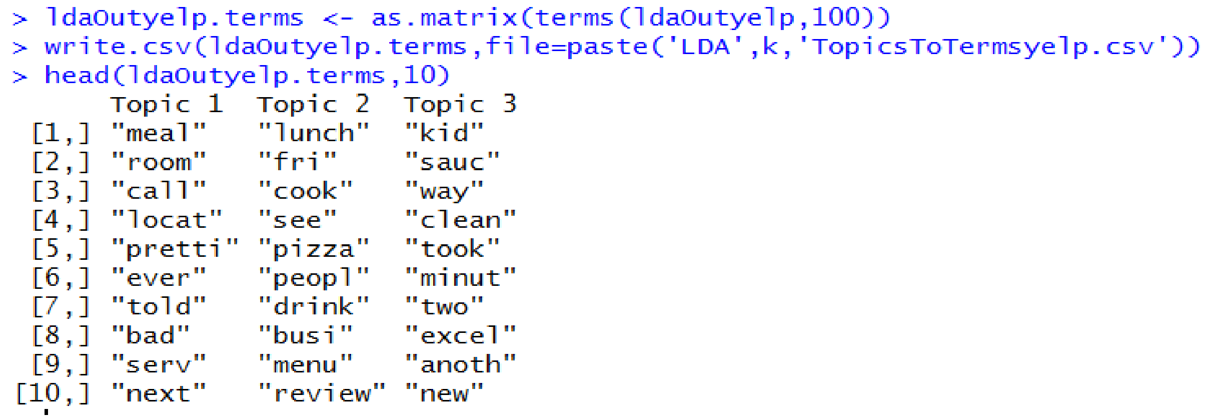
LDA On Yelp

Document To Topics



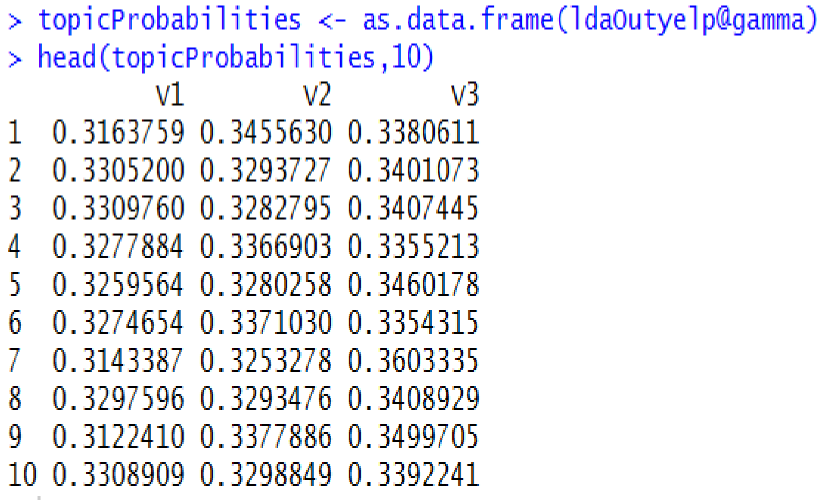
```
> k <- 3
>
> ldaOutyelp <-LDA(datayelpnew,k)
> ldaOutyelp.topics <- as.matrix(topics(ldaOutyelp))
> write.csv(ldaOutyelp.topics,file=paste('LDA',k,'DocsToTopicsyelp.csv'))
> head(ldaOutyelp.topics,10)
   [,1]
1     2
2     3
3     3
4     2
5     3
6     2
7     3
8     3
9     3
10    3
```

Terms To Topics



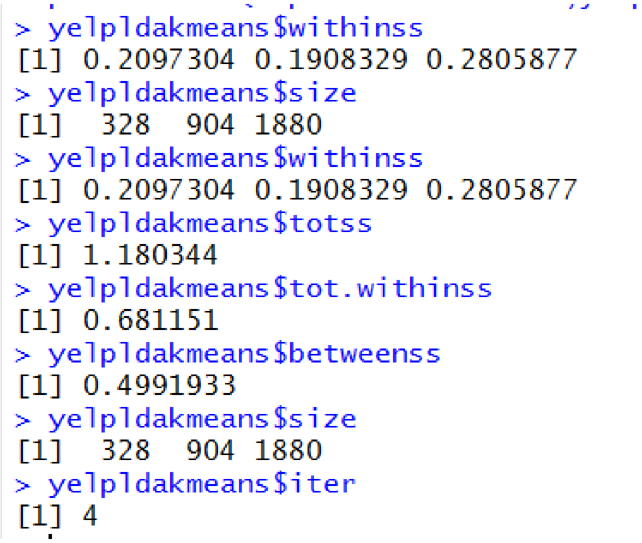
```
> ldaOutyelp.terms <- as.matrix(terms(ldaOutyelp,100))
> write.csv(ldaOutyelp.terms,file=paste('LDA',k,'TopicsToTermsyelp.csv'))
> head(ldaOutyelp.terms,10)
      Topic 1  Topic 2  Topic 3
 [1,] "meal"   "lunch"  "kid"
 [2,] "room"   "fri"    "sauc"
 [3,] "call"   "cook"   "way"
 [4,] "locat"  "see"    "clean"
 [5,] "pretti" "pizza"  "took"
 [6,] "ever"   "peopl"  "minut"
 [7,] "told"   "drink"  "two"
 [8,] "bad"    "busi"   "excel"
 [9,] "serv"   "menu"   "anoth"
[10,] "next"   "review" "new"
```

Topic Probabilities

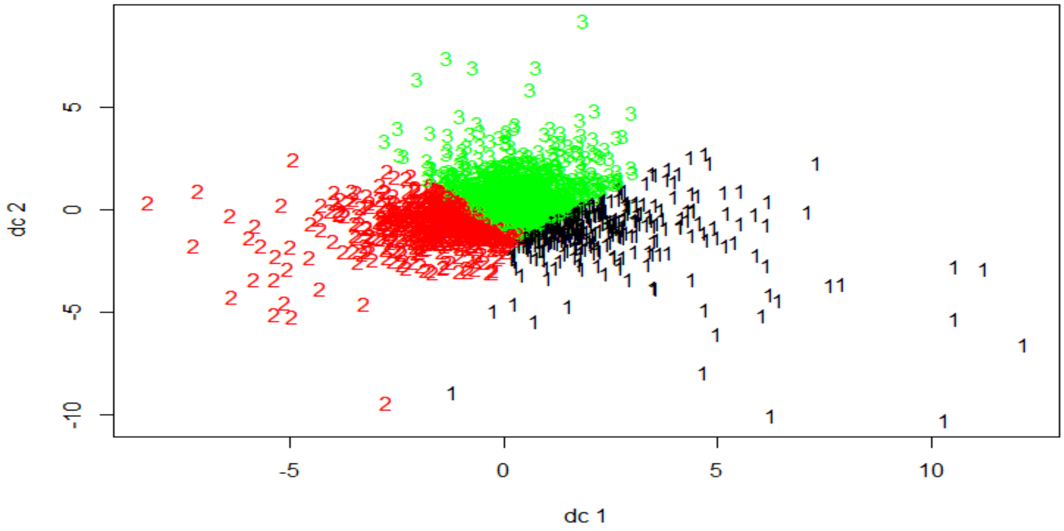


```
> topicProbabilities <- as.data.frame(ldaOutyelp@gamma)
> head(topicProbabilities,10)
          V1        V2        V3
1  0.3163759 0.3455630 0.3380611
2  0.3305200 0.3293727 0.3401073
3  0.3309760 0.3282795 0.3407445
4  0.3277884 0.3366903 0.3355213
5  0.3259564 0.3280258 0.3460178
6  0.3274654 0.3371030 0.3354315
7  0.3143387 0.3253278 0.3603335
8  0.3297596 0.3293476 0.3408929
9  0.3122410 0.3377886 0.3499705
10 0.3308909 0.3298849 0.3392241
```

LDA Clustering using topic probabilities for yelp



```
> yelpldakmeans$withinss
[1] 0.2097304 0.1908329 0.2805877
> yelpldakmeans$size
[1]  328  904 1880
> yelpldakmeans$withinss
[1] 0.2097304 0.1908329 0.2805877
> yelpldakmeans$totss
[1] 1.180344
> yelpldakmeans$tot.withinss
[1] 0.681151
> yelpldakmeans$betweenss
[1] 0.4991933
> yelpldakmeans$size
[1]  328  904 1880
> yelpldakmeans$iter
[1] 4
```



Analysis

- LDA provides a Document to topic representation as well as words to topic representation of the data set.
- Using LDA we can also find relative importance between two topics.
- The more generic the words are the clusters centers are more closer to each other as in case of Yelp.

4.

SSE Comparison

NewsGroup

Algorithms	**Clusters**	**SSE**
Kmeans	2	1917652
LSA (SVD d =50)	2	2963.667
LDA	2	1296.134

Yelp

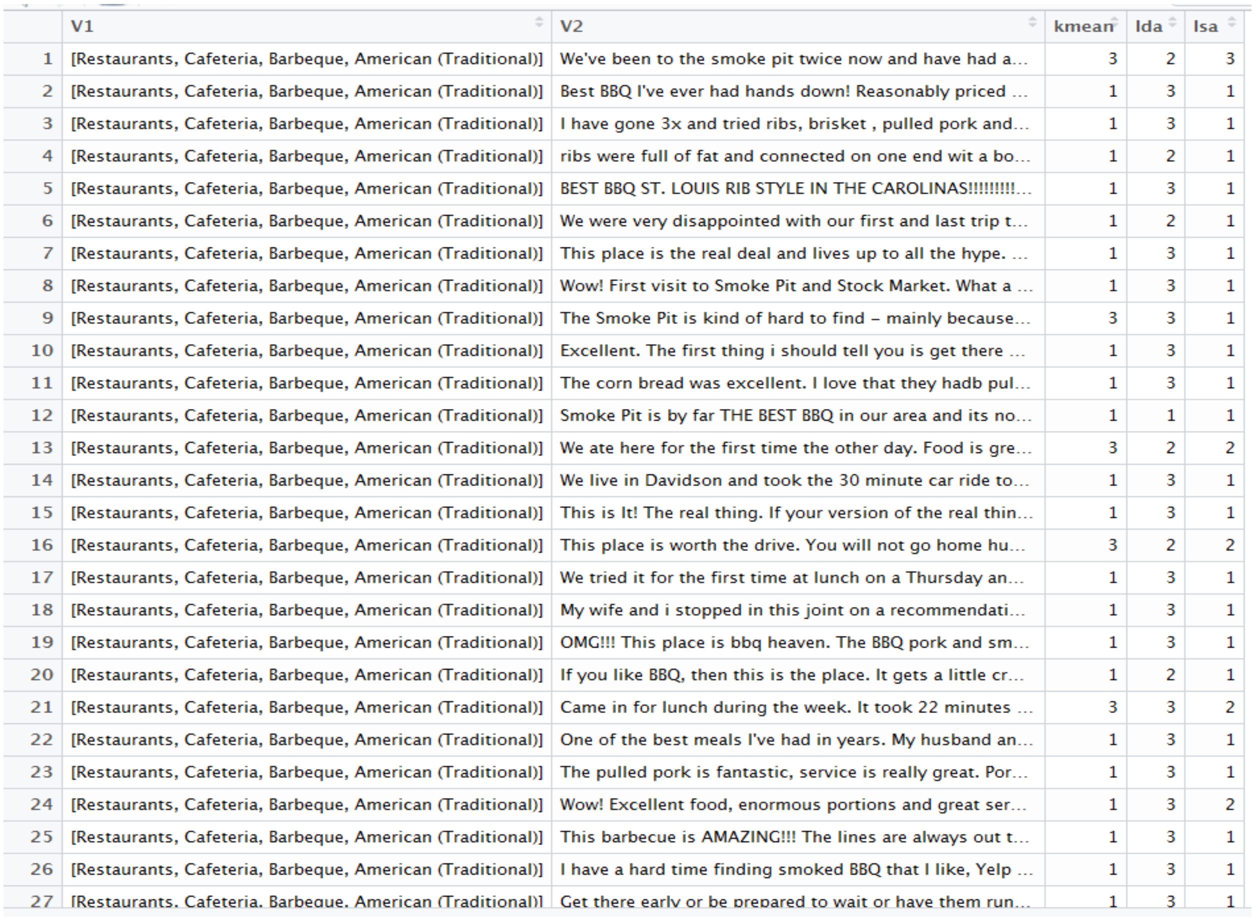
Algorithms	**Clusters**	**SSE**
Kmeans	3	6359.481
LSA (SVD d =50)	3	2310.73
LDA	3	0.681151

Analysis

From the above tables for both Newsgroups and Yelp we can see that LDA performs way better than kmeans and LSA for clustering the documents according to the words and topics.

II. C

Evaluation of Yelp Dataset



	V1	V2	kmean	lda	lsa
1	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	We've been to the smoke pit twice now and have had a…	3	2	3
2	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	Best BBQ I've ever had hands down! Reasonably priced …	1	3	1
3	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	I have gone 3x and tried ribs, brisket , pulled pork and…	1	3	1
4	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	ribs were full of fat and connected on one end wit a bo…	1	2	1
5	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	BEST BBQ ST. LOUIS RIB STYLE IN THE CAROLINAS!!!!!!!!!…	1	3	1
6	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	We were very disappointed with our first and last trip t…	1	2	1
7	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	This place is the real deal and lives up to all the hype. …	1	3	1
8	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	Wow! First visit to Smoke Pit and Stock Market. What a …	1	3	1
9	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	The Smoke Pit is kind of hard to find – mainly because…	3	3	1
10	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	Excellent. The first thing i should tell you is get there …	1	3	1
11	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	The corn bread was excellent. I love that they hadb pul…	1	3	1
12	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	Smoke Pit is by far THE BEST BBQ in our area and its no…	1	1	1
13	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	We ate here for the first time the other day. Food is gre…	3	2	2
14	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	We live in Davidson and took the 30 minute car ride to…	1	3	1
15	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	This is It! The real thing. If your version of the real thin…	1	3	1
16	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	This place is worth the drive. You will not go home hu…	3	2	2
17	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	We tried it for the first time at lunch on a Thursday an…	1	3	1
18	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	My wife and i stopped in this joint on a recommendati…	1	3	1
19	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	OMG!!! This place is bbq heaven. The BBQ pork and sm…	1	3	1
20	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	If you like BBQ, then this is the place. It gets a little cr…	1	2	1
21	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	Came in for lunch during the week. It took 22 minutes …	3	3	2
22	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	One of the best meals I've had in years. My husband an…	1	3	1
23	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	The pulled pork is fantastic, service is really great. Por…	1	3	1
24	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	Wow! Excellent food, enormous portions and great ser…	1	3	2
25	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	This barbecue is AMAZING!!! The lines are always out t…	1	3	1
26	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	I have a hard time finding smoked BBQ that I like, Yelp …	1	3	1
27	[Restaurants, Cafeteria, Barbeque, American (Traditional)]	Get there early or be prepared to wait or have them run…	1	3	1

Algorithms	Accuracy
LDA	64.91003
LSA	82.13368
Kmeans	92.12725

YELP

Algorithms	Clusters	SSE
Kmeans	3	6359.481
LSA (SVD d =50)	3	2310.73
LDA	3	0.681151

- The above calculations are done using CrossTable().
- Each category is represented according to the cluster number we get from kmeans, lda, and lsa algorithms.
- The accuracy is calculated using
  Accuracy = Correctly classified/Total predictions.
- LDA gave less accuracy because the documents are not allotted to the topics in same group. But it using SSE we get LDA as better algorithm than LSA and Kmeans.
- Similar analysis can be done for 20 News Group

III. ANALYSIS

- In this experiment we were given with two datasets
  20 NewsGroup
  Yelp Data set
- I learned about the text mining package(tm), the steps for **text preprocessing** using tm package, how the stop words need to be removed, how to generate a WORD CLOUD, and pruning of infrequent as well as very frequent words.
- The Document Term Matrix, Term Document Matrix, TF-IDF are the matrix we used as inputs for kmeans, LSA and LDA algorithms
- For **Kmeans** we first used nbclust() to determine the k value using different indexes like Duda, KL, CH, Silhouette, Cindex etc.
- Then we clustered the words according to the Document Term frequency.
- Then we came across **LSA** algorithms where we calculated the SVD of TFIDF matrix and thus we explored the hidden data in 20 Newsgroup dataset.
- **LDA** algorithm provided us with Documents to Topics relations and also Terms to Topics Relations. It can also be used to find the relative difference between two topics.
- Using LDA and LSA we can get to know about the semantics of the dataset and about the semantic groups that can be formed from this dataset.
- Also this process of text mining can be used for sentimental analysis as we can analyse whether a review is positive or negative or neutral.