Computational Surprise

Knowledge Discovery in Databases

ITIS/ITCS 6162

Group 10:
Harika Katragadda

Naman Gupta

Sonika Rajan

Swati Jha

Divya Kulkarni

Contents

Abstract

In this project, we developed a computational approach to identify “surprising” news from a news corpus related to diabetes. We implemented the framework using topic modeling approach. We have used different techniques to understand the text in the corpus and finally used Topic modeling to decide the final surprise. Our surprise calculation is based on established metric in the text mining field called Mutual Information (MI), which measures how much information several random variables share, or how much the uncertainty (entropy) of one variable is reduced by knowing the other variables. We have also used correlation for finding the words which are highly correlated to diabetes but deviant and would be helpful in discovering a surprise. Usually a surprise might be general to society or personalized. Through our approach we have made a personalized surprise.

Introduction

Surprise is often defined in terms of disconfirmed expectations, whereby the surprisingness of an event is thought to be dependent on the degree to which that event contrasts with a more likely, or expected, outcome. Surprise is more accurately modelled as a manifestation of an ongoing sense-making process. Specifically, the level of surprise experienced depends on the extent to which an event necessitates representational updating.

Surprise is a familiar experience to us all, whether induced by a noise in the dark, or by an unexpected twist in a murder mystery. A review of the literature reveals that the prevailing definition of surprise relates it directly to expectation. Indeed, this view corresponds to people's own naïve understanding of the phenomenon. Theoretically, expectation is formalized in terms of probabilities, where an unexpected outcome is a low probability event, and vice-versa. If we relate this to surprise, then low probability events should lead to a feeling of surprise, while high probability events should not. While there has been some empirical support for this view, the intuitive relationship between probability and surprise does not always hold. Surprises are generally created by low-probability outcomes, yet, as shown by several experiments, not all low-probability outcomes are equally surprising.

However, distinguishing true surprises – from which useful information can be extracted to improve an agent's world model – from environmental noise is a fundamental challenge.

Computational Approach Related Work

Data Preprocessing

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. Data preprocessing prepares raw data for further processing. For the diabetes data set we performed the following data cleaning steps: Removing the white spaces, removing numeric values, filtering out the stop words using English library and we created a list of stop words which are not there in English, removing special characters, removing punctuations etc. After performing the data cleaning, number of words were reduced by a significant amount. We removed the sparse terms as well. The dataset still contained lot of common English words, so we filtered the dataset to contain the words whose length is between 5 and 20.

After the above filtering and cleaning processes, we started with exploratory data analysis to understand the data. Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis that employs a variety of techniques (mostly graphical) to maximize insight into a data set; uncover underlying structure; extract important variables; detect outliers and anomalies. We created histograms, word cloud, dendrograms, clusters, list of most frequent words.

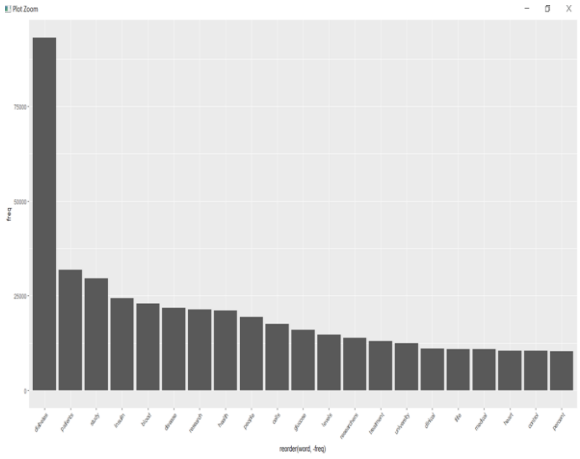

Fig 1. Histogram of most frequent words

Fig 2 Word Cloud

After performing the above EDA processes, we removed the words which are of very high frequency as it has less chances of contributing to finding the surprise element, which is the main task and we also filtered out very low frequency terms as they do not have much significance.

Text Clustering

Clustering is a widely studied data mining technique in the text domains. The technique finds numerous applications in customer segmentation, classification, collaborative filtering, visualization, document organization, and indexing.

Clustering techniques provide a coherent summary of the collection in the form of cluster-digests or word-clusters, which can be used to provide summary insights into the overall content of the underlying corpus. Variants of such methods, especially sentence clustering, can also be used for document summarization, a topic.

K means clustering is one of many techniques within unsupervised learning that can be used for text analysis. *Unsupervised* refers to the fact that we're trying to understand the structure of our underlying data, rather than trying to optimize for a specific, pre-labeled criterion (such as creating a predictive model for conversion). K means clustering groups similar observations in clusters in

order to be able to extract insights from vast amounts of unstructured data. The basic idea of K Means clustering is to form K seeds first, and then group observations in K clusters based on distance with each of K seeds. The observation will be included in the nth seed/cluster if the distance between the observation and the nth seed is minimum when compared to other seeds.

Below is a brief overview of the methodology involved in performing a K Means Clustering Analysis.

The Process of building K clusters on text data:

The first step is data cleansing. This is the most important part as data do not have any specific format.

Remove punctuations, numbers, stop words (R has specific stop word library, but you can also create your own list of stop words). Also, remove duplicate rows or URLs from the social media mentions.

The next step is to create corpus vector of all the words. Once you have created the corpus vector of words, the next step is to create a document term matrix. To enable an effective clustering process, the word frequencies need to be normalized in terms of their relative frequency of presence in the document and over the entire collection. In general, a common representation used for text processing is the vector-space based TF-IDF representation. In the TF-IDF representation, the term frequency for each word is normalized by the inverse document frequency, or IDF. The inverse document frequency normalization reduces the weight of terms which occur more frequently in the collection. This reduces the importance of common terms in the collection, ensuring that the matching of documents be more influenced by that of more discriminative words which have relatively low frequencies in the collection.

One approach would be to use the Elbow method to choose the optimal number of clusters. We have randomly taken the ‘k’ value and conducted the clustering. Below are the results of the text clustering. We were not able to find any words which are deviant from the words related to diabetes so we proceeded to Topic modelling.

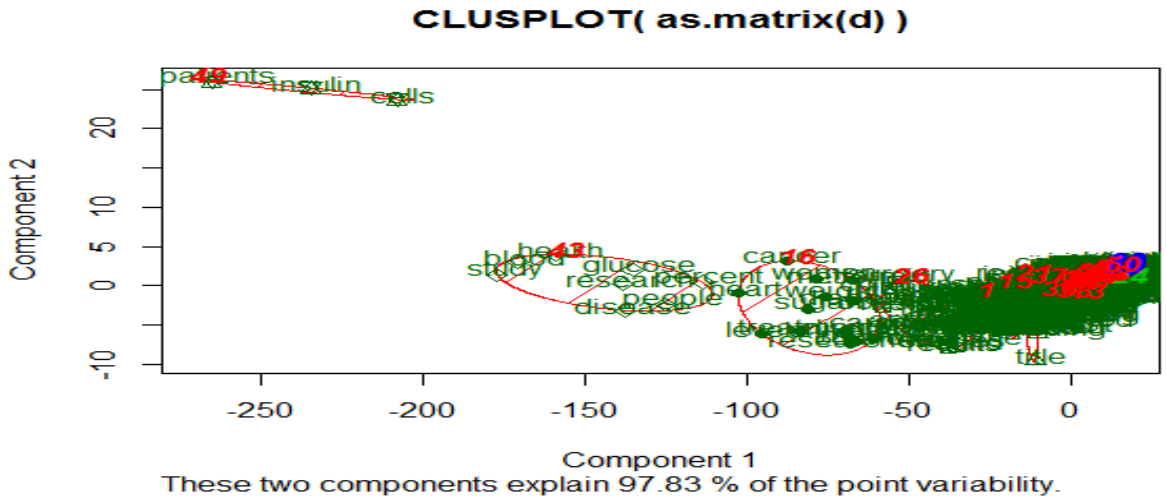

Fig 3 Cluster plot 1

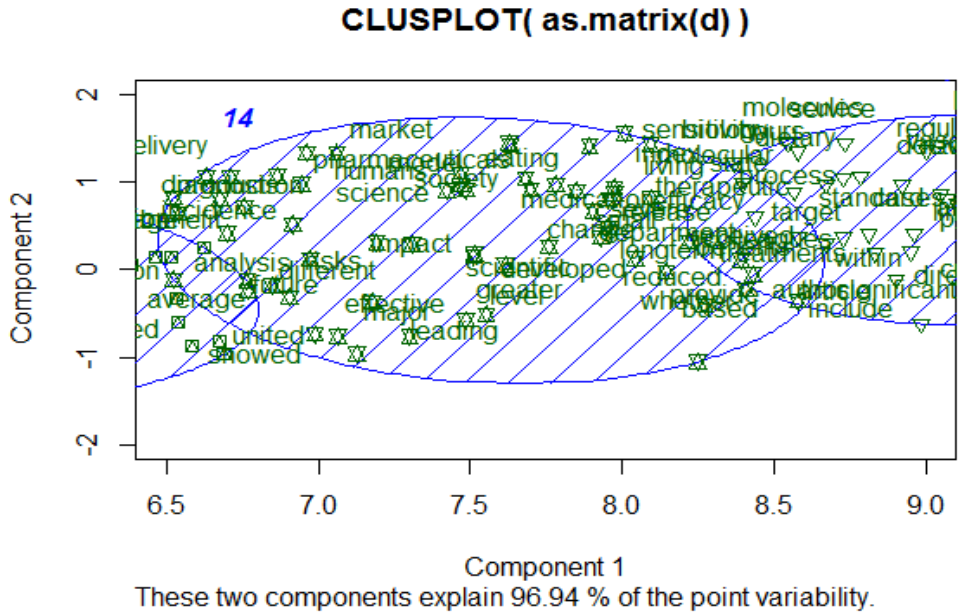

Fig 4 Cluster plot 2 with different values of ‘k’

Clustering - SAS:

We used SAS Miner to get more clear view of the text clustering. We randomly selected the 'k' value until we were able to interpret the clusters properly. The Clusters table contains an ID for each cluster, the descriptive terms that make up that cluster, and statistics for each cluster.

Descriptive Terms		Frequency	Percentage
diabetes health +people care +help +support information +service national +improve	...	1341	13%
+study health +risk +increase +level +high +people +disease +factor +researcher	...	3597	34%
+study +cell +researcher +type +research +find +show +cause +work +body	...	2433	23%
+result clinical +company +product +end +drug +release +market +announce +development	...	577	6%
+cell diabetic clinical +drug +patient +insulin +trial +research +treatment +development	...	1703	16%
+control clinical safety +end +effect +drug +dose +trial information +market	...	803	8%

Fig 5 Clustering results in SAS

Topic Modeling

Topic Modeling provides a way to quickly analyze large volumes of raw text and identify the latent topics. Topic models are statistical models that examine words from a set of documents, determine the themes over the text, and discover how the themes are associated or change over time. Topic modeling helps to uncover the hidden patterns within a corpus. A topic can be viewed as a cluster of words with related meanings, and each word has a corresponding weight inside this topic. Also, the same word can reside in multiple topics with different weights.

The first step is to set a value for the number of topics to be retrieved. LDA (Latent Dirichlet allocation) is one of the most common algorithms for topic modeling. This method helps to find the mixture of words that is associated with each topic and also helps to determine the mixture of topics that describes each document. We have used the LDA() function from the topic models package to set k=5 i.e., to set the number of topics to 5. This function returns an object containing the full details of the model fit, such as how words are associated with topics and how topics are associated with documents.

The next step is to determine the word-topic probabilities. The tidytext package provides the method tidy () for extracting the per-topic-per-word probabilities, called the beta, from the model. has turned the model into a one-topic-per-term-per-row format. For each combination, the model computes the probability of that term being generated from that topic.

We then used the function top_n() provided by package dplyr to find the 30 terms that are most common within each of the 5 topics. The method ggplot() was used to visualize the topic modeling results as shown in Figure 6 below.

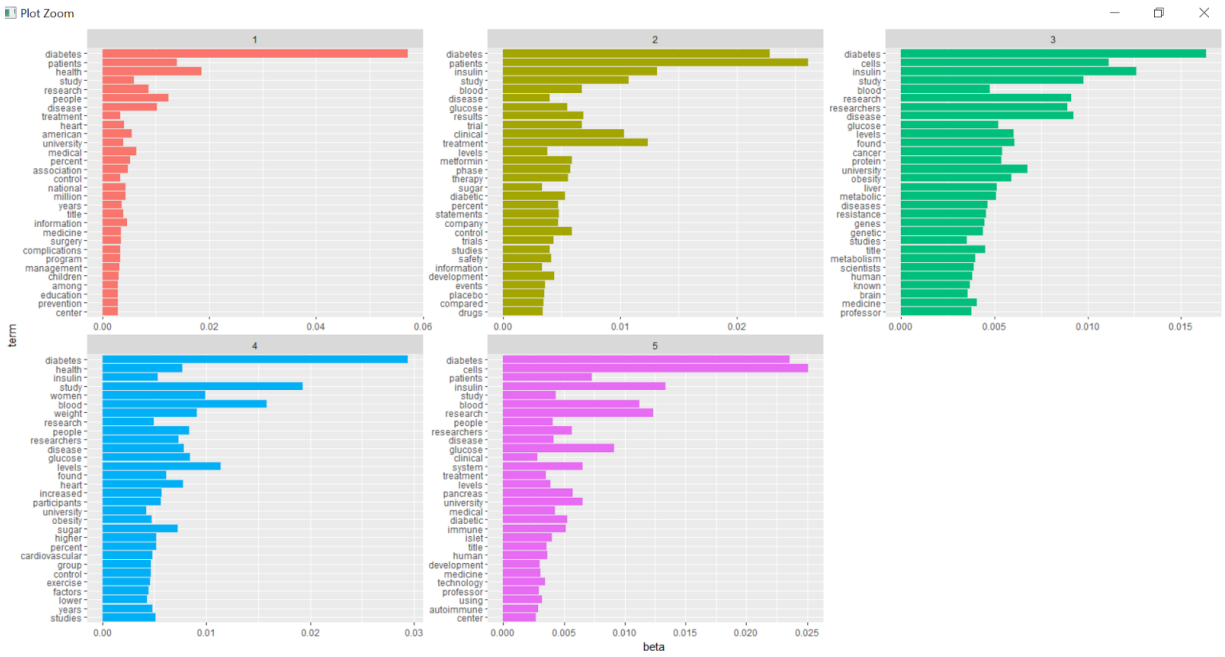

Figure 6: The terms that are most common within each topic

The visualization lets us understand the five topics that were extracted from the corpus. The most common words in Topic 1 include diabetes, patients, health, study, research, people, disease and so on. Similarly, by viewing and analyzing the graph, we can determine what each topic is likely to be talking about. By analyzing the 5 topics and the words appearing in each topic, we randomly picked few words like brain, coffee, sleep and children to understand the relationship that occurred between the terms and the word diabetes. The reason behind selecting the above mentioned 4 words is because we found it surprising that to realize that diabetes has an effect on the selected

terms. So, to learn in depth about the relationship that existed between the terms and word diabetes we performed further analysis.

Using SAS-Miner we did the topic modelling. In the results of the Topics table we can notice that single-term topics have been created along with the multi-term topics. The Multi-term indicates which terms make up each topic.

The + sign at the beginning of certain terms says that there are others term that appeared frequently with center term. The image below will explain the following phenomena

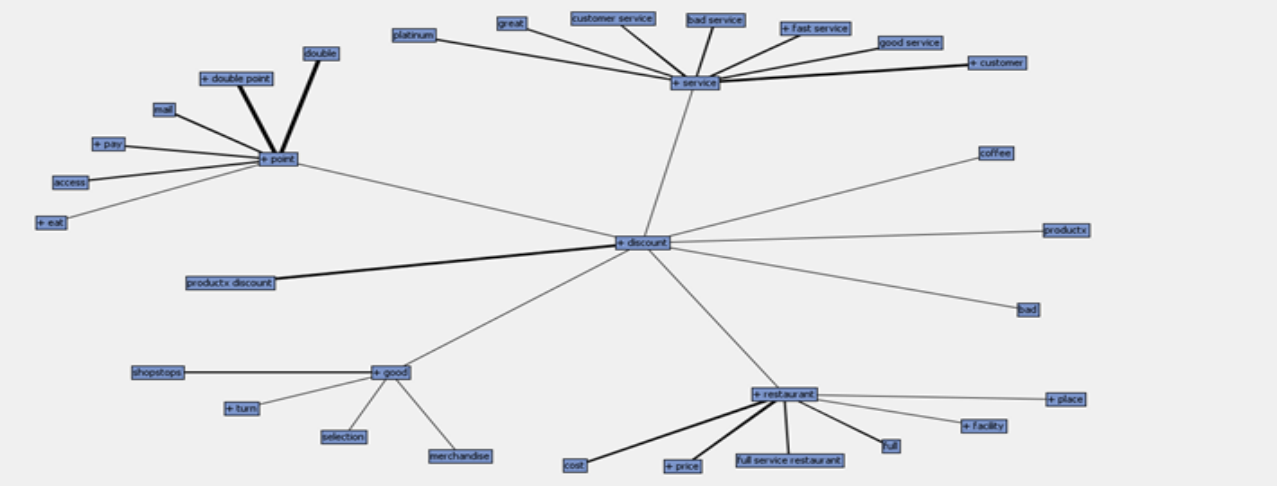

Fig 7 + sign in Topic modelling in SAS

Topic	Number of Terms	# Docs ▼
+number	1	2927
+control	1	2922
+lead	1	2905
+state	1	2844
+affect	1	2808
obesity	1	2751
+team	1	2721
+identify	1	2713
sleep,alzheimer,+brain,cognitive,+insulin	2233	1499
+heart,cardiovascular,+heart attack,+death,+cholesterol	1929	1410
+fat,+fat,+mouse,+resistance,insulin resistance	1633	1409
care,+patient,health,+physician,+practice	2505	1371
+weight,physical,+exercise,+activity,+physical activity	2338	1270
+protein,+brain,alzheimer,+molecule,+cancer	2525	1248
uk,nhs,+people,+cent,type	1989	1236
association,american,+american,ada,+community	2104	1174
immune,autoimmune,+immune system,+autoimmune disease,+system	1859	1082
+device,+glucose,+monitor,+technology,+test	2337	1037
+cell,+stem,+islet,beta,+beta cell	1724	1031
+diet,consumption,+food,+consume,+intake	1895	962
googletag,joslin,+center,research,cmd	2485	940
metformin,mg,+placebo,sitagliptin,+inhibitor	2171	915
+gene,genetic,+variant,+dna,+genome	1939	897
novo,nordisk,+insulin,liraglutide,+injection	2454	828
actos,takeda,avandia,rosiglitazone,pioglitazone	1772	666
+statement,forward-looking,+forward-looking statement,+uncertainty,+company	1459	634
+country,global,+cancer,+world,international	2601	632
+foot,+ulcer,+wound,+heal,+amputation	1826	569
+pregnancy,gestational,+woman,gestational diabetes,pregnant	1056	560
byetta,lilly,amylin,exenatide,glp-1	1227	560
+surgery,bariatric,gastric,+bariatric surgery,+bypass	1392	539
+eye,+retinopathy,+vision,+diabetic retinopathy,+retina	1321	406
googletag,cmd,mnt,+push,+display	2143	133

Fig 8 Topic Modelling results in SAS

Evaluation

Correlation Analysis:

Factors in relationships between two variables

- The **strength** of the relationship:
  - is indicated by the correlation coefficient: **r**
- The **significance** of the relationship
  - is expressed in probability levels: **p** (e.g., significant at p =.05)
  - This tells how **unlikely** a given correlation coefficient, **r**, will occur given **no** relationship
    - The **smaller** the p-level, the more **significant** the relationship
    - The **larger** the correlation, the **stronger** the relationship

Correlation coefficient can be computed using the functions **cor()** or **cor.test()**:

- **cor()** computes the **correlation coefficient**
- **cor.test()** test for association/correlation between paired samples. It returns both the **correlation coefficient** and the **significance level**(or p-value) of the correlation .

It is comprised between -1 and 1.

- **-1** indicates a strong **negative correlation** : this means that every time **x increases**, **y decreases**.
- **0** means that there is no **association** between the two variables (x and y)
- **1** indicates a strong **positive correlation** : this means that **y increases** with **x**

There are different methods to perform correlation analysis, but we used the Pearson correlation. (It measures a linear dependence between two variables (x and y)).

cor(df,method="pearson")

```
> cor(brain_df,diabetes_df , method = "pearson")
                [,1]
cancer   -0.019231127
brain    -0.044769006
sleep    -0.021256075
children  0.155193740
coffee    0.006932675
disease   0.196942102
```

Figure 8: Correlation coefficient using pearson method

cor.test(dfx,dfy,method="pearson")

```
> diabetes_df<-df["diabetes"]
> brain_df<-df[,c("cancer")]
> cor.test(brain_df,diabetes_df , method = "pearson")$p.value
[1] 0.04926124
> brain_df<-df[,c("brain")]
> cor.test(brain_df,diabetes_df , method = "pearson")$p.value
[1] 4.663879e-06
> brain_df<-df[,c("coffee")]
> cor.test(brain_df,diabetes_df , method = "pearson")$p.value
[1] 0.4784562
> brain_df<-df[,c("sleep")]
> cor.test(brain_df,diabetes_df , method = "pearson")$p.value
[1] 0.02974926
> brain_df<-df[,c("children")]
> cor.test(brain_df,diabetes_df , method = "pearson")$p.value
[1] 2.297867e-57
> brain_df<-df[,c("disease")]
> cor.test(brain_df,diabetes_df , method = "pearson")$p.value
[1] 6.392683e-92
```

Figure 9: Significance values for various rare words(p)

Conclusion

We developed analytical approaches for evaluating the personalized surprise and applied our computational models to a data-set of documents related to Diabetes. We came across high and low levels of surprise in our study, based on each of our approaches to modeling surprise. Our interpretation of these results is that topic co-occurrence likelihood serves as a good predictor of the level of surprise in documents related to health. Combination of topics that are rare are more likely to contain surprising content, such as the co-occurrence example of brain and cancer, like brain and diabetes, children and diabetes etc.

Our general belief is that people with diabetes are at significantly higher risk for many forms of cancer lung/bronchus, colon/rectum, stomach, and liver cancer. But above results illuminate surprising relationship between Brain Cancer and diabetes.

When we look for correlation we found negative correlation between diabetes and brain cancer. It surprised us that diabetes has negative impact on brain cancer. Which implies Brain cancer are less common among those with elevated blood sugar and diabetes.

This really forces us to think, “Why is the association between blood glucose levels and brain cancer the opposite of that for several other cancerous tumors”?

Personalized surprise is just for a person based on a person’s background knowledge. Personalized surprise is not necessarily surprising to society.

Based on the result we understood it’s a personalized surprise as we are not aware of this fact.

Appendix

The project R code is given below:

```
#get input data
setwd("C:/College/KDD/project/")
library(tm)
docs<-Corpus(DirSource("diabetes"))
writeLines(as.character(docs[[1]]))

#data                                                    pre-processing
docs              <-              tm_map(docs,              stripWhitespace)
docs          <-          tm_map(docs,          content_transformer(tolower))

docs      <-      tm_map(docs,      removeWords,      stopwords("english"))
for               (j               in               seq(docs))              {
 docs[[j]]         <-         gsub("/",         "         ",         docs[[j]])
 docs[[j]]         <-         gsub("\"",         "         ",         docs[[j]])
 docs[[j]]         <-         gsub("@",         "         ",         docs[[j]])
 docs[[j]]         <-         gsub("~",         "         ",         docs[[j]])
 docs[[j]]         <-         gsub("\\|",         "         ",         docs[[j]])
 docs[[j]]        <-        gsub("\u2028",        "        ",        docs[[j]])
 docs[[j]]        <-        gsub("\u0090",        "        ",        docs[[j]])
 docs[[j]] <- gsub("Ãâ???zÂ", " ", docs[[j]])# This is an ascii character
that    did    not    translate,    so    it    had    to    be    removed.
}

docs              <-              tm_map(docs,              removePunctuation)
docs               <-               tm_map(docs,               removeNumbers)

removeNumPunct     <-     function(x)     gsub("[^[:alpha:][:space:]]*","",x)
docs       <-       tm_map(docs,       content_transformer(removeNumPunct))


myStopwords                              <-                              c("can",
"say","one","way","use","also","howev","tell","will",
                "much","need","take","tend","even","like","particular","ra
ther","said",
                "get","well","make","ask","come","end","first","two","help
","often","may",
                "might","see","someth","thing","point","post","look","righ
t","now","think",
                "'ve                                                    ","'re
","remains","suggesting","rapidly","principal","involving",
                "millions","publication","supporting","opportunity","dedic
ated","fully","continues","avoid","assessed","basis","plays","defined","ca
using","investigate","released","founded","examine","majority","allows","c
oncluded","presence","respond","options","amounts","michael","continued","
differ","reach","ensure","toward","april","demonstrate","studys","actually
","getting","though","indicated","contains","course","variety","monitor","
learn","instead","revealed","option","forms","larger","seven","estimates",
"create","march","suffer","properly","chair","necessary","comprehensive","
generally","success","improves","performed","essential","activities","eval
```

```
uate","notes","completed","explain","effectiveness","industry","officer","
resulting","ongoing","raise","signs","normally","underlying","fewer","redu
ces","investigation","influence","burden","agents","short","promising","no
ted","clinicians","actual","twice","analyzed","existing","evaluated","regu
lation","processes","methods","helping","advance","progress","managing","w
hole","allow","great","controls","combined","protect","executive","difficu
lt","leader","environment","controlling","approaches","prior","suggested",
"profile","thought","assess","effectively","indicate","maintain","resource
s","complete","network","regulate","status","highly","worlds","country","r
equire","implications","promote","pathways","looked","environmental","thir
d","simple","along","nations","successful","central","strong","explains","
highest","responsible","stage","plans","lowering","occur","member","david"
,"appear","design","direct","experienced","strategy","regarding","field","
attacks","researcher","relationship","commonly","latest","usually","please
","light","method","appropriate","tolerance","little","works","specificall
y","moderate","beneficial","versus","explained","impaired","initial","samp
les","offer","physician","forward","makes","experts","genetics","grant","o
ffers","improvements","unique","canada","centre","achieved","recommendatio
ns","companies","written","various","clear","deaths","knowledge","boston",
"directly","persons","innovative","means","natural","issues","despite","aw
areness","independent","whose","excess","nature","approval","clinic","plac
e","collaboration","molecule","associations","examined","efforts","start",
"address","followed","looking","secretion","leads","chemical","providing")
#remove                          custom                              stopwords
docs        <-        tm_map(docs,        removeWords,        myStopwords)

#dtm                      <-                      DocumentTermMatrix(docs)
#dtm             <-             removeSparseTerms(dtm,             0.95)
dtm3   <-DocumentTermMatrix(docs,   control=list(wordLengths=c(5,   20)))

dtm3

freq <- sort(colSums(as.matrix(removeSparseTerms(dtmss, 0.95))),decreasing
=                                                                   TRUE)
length(freq)

head(table(freq),                                                     20)

#ways            to            view            term            frequency
freq     <-     sort(colSums(as.matrix(dtm4)),     decreasing=TRUE)
head(freq,                                                          14)
tail(freq,                                                          14)

findFreqTerms(dtm4,                                          lowfreq=50)
wf       <-       data.frame(word=names(freq),       freq=freq)
head(wf)


#word                                                                cloud
library("SnowballC")
library("wordcloud")
library("RColorBrewer")
set.seed(1234)
wordcloud(words   =   wf$word,   freq   =   wf$freq,   min.freq   =   2000,
```

```
        max.words=200,          random.order=FALSE,          rot.per=0.35,
        colors=brewer.pal(8,                                     "Dark2"))

df_dis           <-           wf[wf$word           ==           "sleep",]
df_dis
write.csv(wf,               file                =               "df.csv")

dtm4         <-         DocumentTermMatrix(docs,         control         =
list(dictionary=c("sleep","brain","coffee","cancer","children","diabetes")
))
#plotting                              the                              words
install.packages("ggplot2")
library(ggplot2)

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

p

#relationships                          between                          words

findAssocs(dtm3,     c("brain"     ,     "diabetes"),     corlimit=0.05)

findAssocs(dtm,                  "university",                  corlimit=1)


#clustering
install.packages("cluster")
library(cluster)
dtmss             <-             removeSparseTerms(dtm3,             0.95)
dtmss
d         <-         dist(t(dtmss),         method="euclidian")
fit  <-  hclust(d=d,  method="ward.D")    #  for  a  different  look  try
substituting:                                          method="ward.D"
fit

plot(fit,                                                        hang=-1)

#creating                                                         borders

plot.new()
plot(fit,                                                        hang=-1)
groups <- cutree(fit, k=6)   # "k=" defines the number of clusters you are
using
rect.hclust(fit,                    k=2,                    border="red")



library(fpc)
d         <-         dist(t(dtm3),         method="euclidian")
kfit              <-              kmeans(d,              15)
clusplot(as.matrix(d),  kfit$cluster,  color=T,  shade=T,  labels=2,  lines=0,
xlim         =         c(6.5,9),         ylim         =         c(-2,2))
```

```
install.packages("infotheo")
library(infotheo)

entropy                  <-                  function(p)                  {
 #        Assumes:        p        is        a        numeric        vector
 if               (sum(p)               ==               0)               {
   return(0)
 }
 p     <-     p/sum(p)     #     Normalize     so     it     sums     to     1
 p  <-  p[p  >  0]  #  Discard  zero  entries  (because  0  log  0  =  0)
 H                         =                         -sum(p*log(p,base=2))
 return(H)
}

m3<-as.matrix(dtmss)
grep('dis',m3,                                                 value=TRUE)
df3<-as.data.frame(m3)
entropy_dis<-df3[,"european"]

entropy(entropy_dis)

dtm_tfxidf                        <-                       weightTfIdf(dtm3)
m1                          <-                          as.matrix(dtm_tfxidf)
m<-t(m1)
memory.limit(size=56000)

norm_eucl <- function(m) m/apply(m, MARGIN=1, FUN=function(x) sum(x^2)^.5)
m_norm                             <-                             norm_eucl(m)
num_cluster<-10
cl               <-               kmeans(t(m1),               num_cluster)

cl

plot(m,col=cl$cluster)
points(cl$centers,    col    =    1:2,    pch    =    8,    cex    =    2)


install.packages("infotheo")
library(infotheo)


entropy_dis1<-df3[,"diabetes"]
entropy_dis2<-df3[,"sleep"]

mutinformation(entropy_dis1,entropy_dis2,method="emp")

#calculate                                                      correlation
m<-as.matrix(dtm3)
df<-as.data.frame(m)

diabetes_df<-df[,"diabetes"]
brain_df<-df[,"cancer"]
```

```
brain_df<-df[,c("disease")]
cor.test(brain_df,diabetes_df     ,     method     =     "pearson")$p.value
brain_df
,"brain","sleep","children","coffee","disease"
cor.test()
#topic                                                          modeling****
install.packages("topicmodels")
library(topicmodels)


memory.limit(size=56000)
rowTotals          <-          apply(dtm3          ,          1,          sum)
dtm.new           <-         dtm3[rowTotals>         0,          ]


ap_lda   <-   LDA(dtm.new,   k   =   5,   control   =   list(seed   =   2345))
ap_lda

install.packages("tidytext")
library(tidytext)

install.packages("magrittr")
library(magrittr)

#                          Word-Topic                              probabilities
ap_topics       <-       tidy(ap_lda,       matrix       =       "beta")
ap_topics

install.packages("ggplot2")
library(ggplot2)

install.packages("dplyr")
library(dplyr)

#    Visualization    of    the    top    10    words    for    each    topic
ap_top_terms                <-                ap_topics                %>%
 group_by(topic)                                                         %>%
 top_n(30,                           beta)                               %>%
 ungroup()                                                               %>%
 arrange(topic,                                                        -beta)

ap_top_terms                                                             %>%
 mutate(term          =          reorder(term,          beta))          %>%
 ggplot(aes(term,      beta,      fill      =      factor(topic)))      +
 geom_col(show.legend               =               FALSE)               +
 facet_wrap(~        topic,        scales        =        "free")        +
 coord_flip()

topics(ap_lda)

terms(ap_lda)
#The words with the greatest differences among the three topics are
visualized in Figure. We filter out those rare words (<1/1000).
```

```
install.packages("tidyr")
library(tidyr)

beta_spread                <-                ap_topics                %>%
 mutate(topic         =         paste0("topic",         topic))         %>%
 spread(topic,                          beta)                          %>%
 filter(topic1 > .001 | topic2 > .001| topic3 > .001| topic4 > .001| topic5
>                                .001)                                  %>%
 mutate(entropy         =         -topic1*log2(topic1)-topic2*log2(topic2)-
topic3*log2(topic3)-topic4*log2(topic4)-topic5*log2(topic5))

beta_spread

beta_spread                                                             %>%
 top_n(-10,                            entropy)                         %>%
 mutate(term         =         reorder(term,         entropy))         %>%
 ggplot(aes(term,                          entropy))                       +
 geom_col()                                                              +
 labs(y  =  "entropy  of  terms  over  the  five  topics  distribution")  +
 coord_flip()

#Document-topic                                                probabilities
library(tidyr)
install.packages("tidytext")
library(tidytext)
ap_documents       <-       tidy(ap_lda,       matrix       =       "gamma")
ap_documents

gamma_spread               <-               ap_documents               %>%
 mutate(topic         =         paste0("topic",         topic))         %>%
 spread(topic, gamma)
```

References

- Surprise Me If You Can: Serendipity in Health Information, Xi Niu,Fakhri Abbas,Mary Lou Maher
- Data-intensive evaluation of design creativity using novelty, value, and surprise.Kazjon Grace, Mary Lou Maher, Douglas Fisher & Katherine Brady
- Finding contradictions in text, Marie-Catherine de Marneffe
- Topic Modeling For Associated Press Articles Using Latent Dirichlet Allocation [LDA]
- Use of Topic Modeling for Recommending Relevant Education Material to Diabetic Patients. Sasikiran Kandula, Dorothy Curtis, Brent Hill, and Qing Zeng-Treitler