

1. **ABSTRACT**

With the increase in popularity of online review sites comes a corresponding need for tools capable of extracting the information most important to the user from the plain text data.

Sentiment Analysis is a widely addressed Natural Language Processing task wherein the semantic orientation of a text unit is adjudged. However, a major challenge in Sentiment Analysis is the identiﬁcation of entities towards which the opinion is expressed. Aspect based Sentiment Analysis consists of two tasks. The ﬁrst part involves the extraction of the aspect term from a sentence and secondly the polarity of the opinion corresponding to that aspect is adjudged.

This project is the implementation for Unsupervised and Supervised Methods for Aspect-Based Sentiment Analysis [1] of restaurant reviews dataset. Some Unsupervised and supervised methods are proposed, implemented and evaluated. We focus on determining the aspect terms existing in each sentence, finding out their polarities, detecting the categories of the sentence and the polarity of each category.

**2. INTRODUCTION**

**2.1 General Introduction to the Topic**

Sentiment analysis is increasingly viewed as a vital task both from an academic and a commercial standpoint. The majority of current approaches, however, attempt to detect the overall polarity of a sentence, paragraph, or text span, regardless of the entities mentioned (e.g. restaurants) and their aspects (e.g. food, service).

By contrast, this task is concerned with aspect based sentiment analysis (ABSA), where the goal is to identify the aspects of given target entities and the sentiment expressed towards each aspect.

Eg. The food was good, but the ambiance was poor.

Here, we have 2 aspects “food”, and “ambiance”. Also, polarities associated with both aspects are “positive”, and “negative” respectively.

We will use 2 labelled datasets for this problem, and will find aspects, corresponding polarities, and visualize them.

**2.2 Organization**

**Impetus**is a software solutions, products and services company headquartered in Los Gatos, USA with development centers in NOIDA, Indore, Gurgaon, and Bengaluru, India.

With more than 1600 employees globally, Impetus is focused on creating new ways of analyzing data for businesses—helping them gain key business insights across the enterprise.

 They bring together a unique mix of Data Science capabilities and technology expertise across the Big Data ecosystem including Hadoop, NoSQL, NewSQL, MPP databases, machine learning, and innovative visualization.

**2.3 Area of Computer Science**

Aspect Based Sentiment Analysis (ABSA) falls under the Machine Learning Branch of Computer Science Applications.

Machine learning is the science of getting computers to act without being explicitly programmed. In the past decade, machine learning has given us self-driving cars, practical speech recognition, effective web search, and a vastly improved understanding of the human genome. Machine learning is so pervasive today that you probably use it dozens of times a day without knowing it.

Many researchers also think it is the best way to make progress towards human-level AI. In this class, you will learn about the most effective machine learning techniques, and gain practice implementing them and getting them to work for yourself.

**2.4 Hardware and Software Requirements**

* R studio
* TensorFlow
* CoreNLP

**3. PROBLEM DEFINITION**

2.1 Dataset

The data set provided by SemEval is a subset of Ganu et at(2009). It is in the XML format, and has separate ﬁles for Laptop and Restaurant reviews. The training data contains about 500 reviews ,which is 1606 sentences for Restaurants. The image attached below shows a part the XML ﬁle of the Restaurant data.



Fig.1: Snipptet of the Restaurant dataset XML ﬁle

For reach sentence, we have a target attribute which lists the aspect term, and a corresponding polarity attribute. The distribution of positive and negative sentiments in both the dataset is provided below:

|  |  |  |
| --- | --- | --- |
| Domain | Positive | Negative |
| Restaurant | 1198 | 408 |

**4.  OBJECTIVE(S)**

Datasets consisting of customer reviews with human-authored annotations identifying the mentioned aspects of the target entities and the sentiment polarity of each aspect are given.

The goal is to identify the aspects of given target entities and the sentiment expressed towards each aspect. Eg. The food was good, but the ambiance was poor.

5. **BACKGROUND**

Various approaches have been adopted to identify aspects from sentences. Bing Lui et al. used frequency of noun phrases, followed by a redundancy pruning to identity the feature corresponding to a review[6].Yejin Choi et al. performed semantic tagging using conditional random ﬁelds with features based on Capitalization, syntactic chunking to extract sources of opinions from texts[3].

The best performing one uses a Conditional Random Field with features extracted using named entity recognition, POS tagging and parsing.

We try to augment this approach by using features not only based upon text processing techniques, but also on vector embeddings of words and sentences. The motivation behind this being that the number of candidate aspect words of restaurant domain is rather restrictive. The task of polarity detection was addressed using various classiﬁcation techniques like Naive Bayes, SVM etc.

**6. METHODOLOGY**

In particular, the task consists of the following subtasks:

**6.1 Aspect Term Extraction**

Given a set of sentences with pre-identified entities (e.g., restaurants), identify the aspect terms present in the sentence and return a list containing all the distinct aspect terms. An aspect term names a particular aspect of the target entity.

For example, "I liked the service and the staff, but not the food”, “The food was nothing much, but I loved the staff”. Multi-word aspect terms (e.g., “hard disk”) should be treated as single terms (e.g., in “The hard disk is very noisy” the only aspect term is “hard disk”).

**6.2 Aspect Term Polarity**

For a given set of aspect terms within a sentence, determine whether the polarity of each aspect term is positive, negative, neutral or conflict (i.e., both positive and negative).

For example:

“I loved their fajitas” → {fajitas: positive}

“I hated their fajitas, but their salads were great” → {fajitas: negative, salads: positive}

“The fajitas are their first plate” → {fajitas: neutral}

“The fajitas were great to taste, but not to see” → {fajitas: conflict}

**6.3 Aspect Category Detection**

Given a predefined set of aspect categories (e.g., price, food), identify the aspect categories discussed in a given sentence. Aspect categories are typically coarser than the aspect terms of Subtask 1, and they do not necessarily occur as terms in the given sentence.

For example, given the set of aspect categories {food, service, price, ambience, anecdotes/miscellaneous}:

“The restaurant was expensive, but the menu was great” → {price, food}

**6.4 Aspect Category Polarity**

Given a set of pre-identified aspect categories (e.g., {food, price}), determine the polarity (positive, negative, neutral or conflict) of each aspect category.

For example:

“The restaurant was expensive, but the menu was great” → {price: negative, food: positive}

**7. IMPLEMENTATION DETAILS**

**7.1 Unsupervised Approach**

We load the XML file into our R studio environment and extract all the text from it.

docxml=ldply(xmlToList("Restaurants\_Train.xml"), data.frame)

sentence <- as.data.frame(unique(docxml$text))

for(i in 1:3038)

txt <- paste (txt, sentence$Text[i])

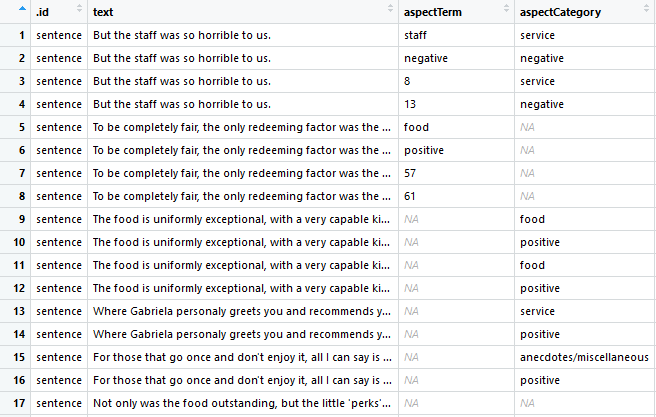


Fig.2:Docxml Dataframe Snapshot

We now need to extract terms from our sentences . We need to build a custom built function for extraction and cleansing of the data.

#POS tagging

tagPOS <- function(x, ...) {

s <- as.String(x)

word\_token\_annotator <- Maxent\_Word\_Token\_Annotator()

a2 <- Annotation(1L, "sentence", 1L, nchar(s))

a2 <- annotate(s, word\_token\_annotator, a2)

a3 <- annotate(s, Maxent\_POS\_Tag\_Annotator(), a2)

a3w <- a3[a3$type == "word"]

POStags <- unlist(lapply(a3w$features, `[[`, "POS"))

POStagged <- paste(sprintf("%s/%s", s[a3w], POStags), collapse = " ")

list(POStagged = POStagged, POStags = POStags)}

tagged\_str <- tagPOS(txt)

#filter nouns

grep('NN', tagged\_str$POStags)## used this to make vector v1

for(i in 1:9820)##used this to put those words in noun df

noun <-paste(noun, pos\_words[v1[i],])

noun <- as.data.frame(strsplit(noun," "))

#to remove /NN... from our df

noun$word<- as.character(noun$word)

for(i in 1:nrow(noun))

noun$word[i]<- gsub("/+[[:alpha:]]\*","",noun$word[i])

#to remove punct

for (i in 1:nrow(noun))

noun$word[i]<- gsub("[[:punct:]]", "", noun$word[i])

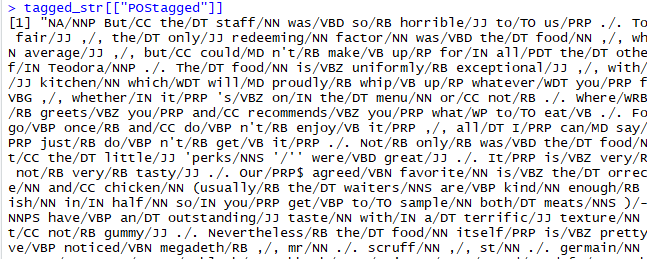


Fig.3: POS Tagged result

In the tagged\_str result set, all the aspects (nouns) are tagged by /NN.

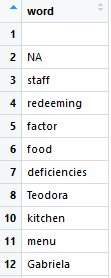


Fig.4: Aspects Dataframe Snippet

We now find the aspects in each sentence so as to link them later to make governor-Dependent pairs.

j<- 3

#Assomatrix contruction #use try and catch while contructing loops

for( i in 1: nrow(Assomatrix))

{ k=2

while(isTRUE(grep(noun$word[j],Assomatrix$Text[i])==1))

{

Assomatrix[i,k]<- noun$word[j]

j=j+1

k=k+1

}

}

# Remove NA from Assomatrix so no error later

Assomatrix[is.na(Assomatrix)] <- ""

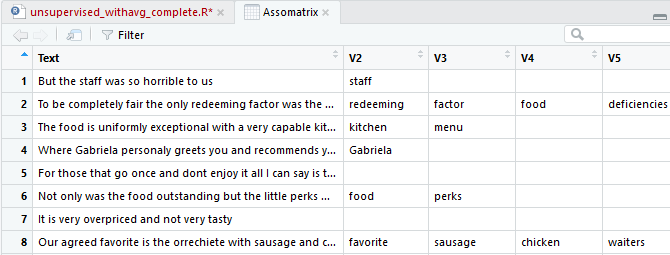


Fig.5: Sentence – Aspect Matrix

We now need to perform stemming and lemmatization on both

aspects<- as.data.frame(unique(noun))

aspects<- as.data.frame(lemmatize\_words(stem\_words(aspects$word)))

transaction<- as.data.frame(t(aspects))

for(i in 1:nrow(Assomatrix))

Assomatrix[i,]<- stem\_strings(Assomatrix$Text[i])

#to change col names of transaction

for(i in 1:nrow(aspects))

names(transaction)[i]<-paste(aspects$`lemmatize\_words(stem\_words(copy))`[i])

A Term – Document matrix is then constructed to create formal transactions.

# to put 1,0 in sentence-aspect matrix

for(i in 1:nrow(tran))

for(j in 1:ncol(tran))

{

if(isTRUE(grep(aspects$`lemmatize\_words(stem\_words(copy))`[j],lemmatize\_strings(stem\_strings(Assomatrix$Text)[i]))==1))

{

tran[i,j]<- 1

}

else

tran[i,j]<-0

}

# create transact 0 filled data frame to make a new matrix

for(i in 1:ncol(tran))

{ k=2

for(j in 1:nrow(tran))

{

if(tran[j,i]==1)

{

transact[j,k]<- j

k=k+1

}

}

}

### make transpose of transact

tran\_trim<- transact

transact<- t(transact)

transact<-data.frame(transact)

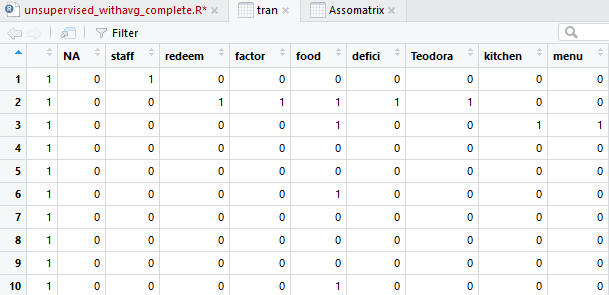


Fig.6: Document-Term Matrix

We now used apriori algorithm to find the most frequent item sets

trans<- as(newmatrix,"transactions")

itemsets <- apriori(trans, parameter = list(target = "frequent",

supp=0.03, minlen = 2, maxlen=4))

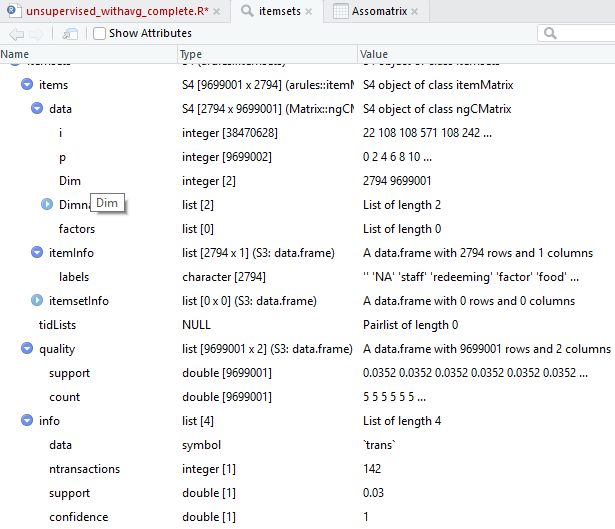


Fig.7: Apriori Algorithm

frequent<- as.data.frame(inspect(head(sort(itemsets), n=nrow(transact)))

frequent$items<-as.character(frequent$items)

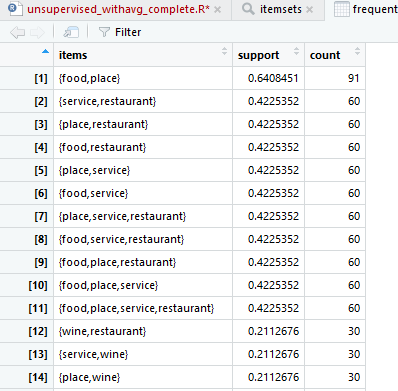


Fig.8: Most Frequent Itemsets

Inference: {food,place} is the most frequent itemset which is present in our data set.

It is present 91 times in 1000 sentences.

We now find Governor – Dependent pairs using the Stanford Dependency parser (SDP).

GD pairs help us find relationship between 2 words

for(k in 1:nrow(Assomatrix))

{

temp<- as.data.frame(getDependency(annotateString(Assomatrix$Text[k])))

#find GD Pairs of sentence k

temp <- temp[which((temp$governor %in% noun$word & temp$dependent %in% lexicon$word) | (temp$dependent %in% noun$word & temp$governor %in% lexicon$word)),]

if(k==1)

Dependency\_final <- temp

else

Dependency\_final<- rbind(Dependency\_final,temp)

}

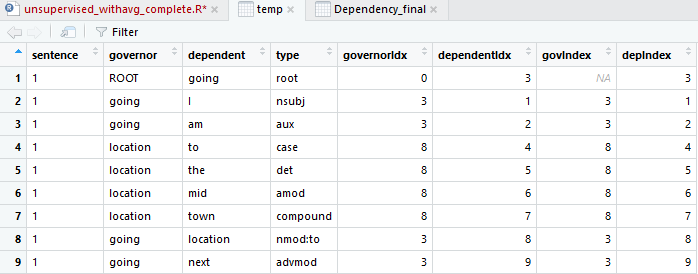


Fig.9: Governor-Dependent Pairs

We also need to consider negated words like” did not like the food”

By default like has a polarity of +1 but in this case the sentence will have -1 since like is negated by the use of “not”

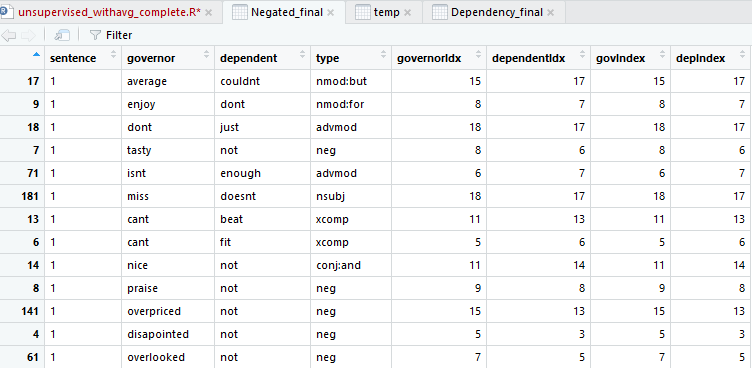


Fig.10: Negation list Filtered

We then create an extra column in the Dependency\_final dataframe for the GD pairs.

#load polarity in Dependency\_final dataframe

lexicon$word<- as.character(lexicon$word)

for (i in 1: nrow(Dependency\_final))

Dependency\_final$polarity[i]<- lexicon$value[match(Dependency\_final[i,which(Dependency\_final[i,] %in% lexicon$word)] , lexicon$word)]

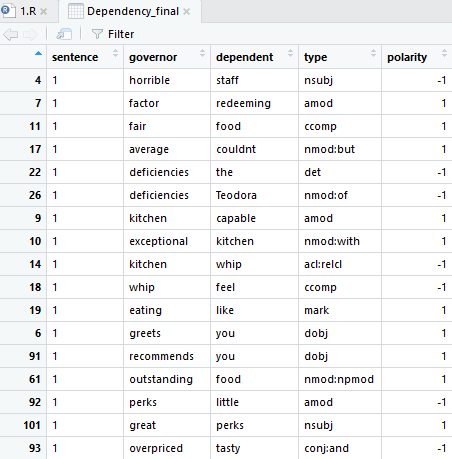


Fig.11: Polarity for GD Pairs

Since we now have GD pairs with polarity and the negated pairs, we can correctly predict the polarity of each sentence of the test set.

#make polarity coloumn in Assomatrix

#if Dependency final gov-dep pairs are present (GREP) then set polarity for that sentence = calculated polarity

#problem: multiple pairs point to the same sentence so previous value is overwritten

Assomatrix$polarity <-0

for(i in 1:nrow(Assomatrix))

{ pola =0

for(j in 1:nrow(Dependency\_final))

{

if(isTRUE((grep(Dependency\_final$governor[j],Assomatrix$Text[i]))&(grep(Dependency\_final$dependent[j],Assomatrix$Text[i]))))

{

pola = pola +Dependency\_final$polarity[j]

}

}

Assomatrix$polarity[i]<- pola

}

for(i in 1:nrow(Assomatrix))

{

if(Assomatrix$polarity[i]>=0)

Assomatrix$polarity[i] = 1

else if(Assomatrix$polarity[i] <0)

Assomatrix$polarity[i] = -1

}

#then run word to word grep in each sentence for Negation pairs

#if any pair is present then multiply polarity by -1

for(i in 1:nrow(Assomatrix))

{

for(j in 1:nrow(Negated\_final))

{

if(isTRUE((grep(Negated\_final$governor[j],Assomatrix$Text[i]))&(grep(Negated\_final$dependent[j],Assomatrix$Text[i]))))

{

Assomatrix$polarity[i]<- Assomatrix$polarity[i]\*(-1)

}

}

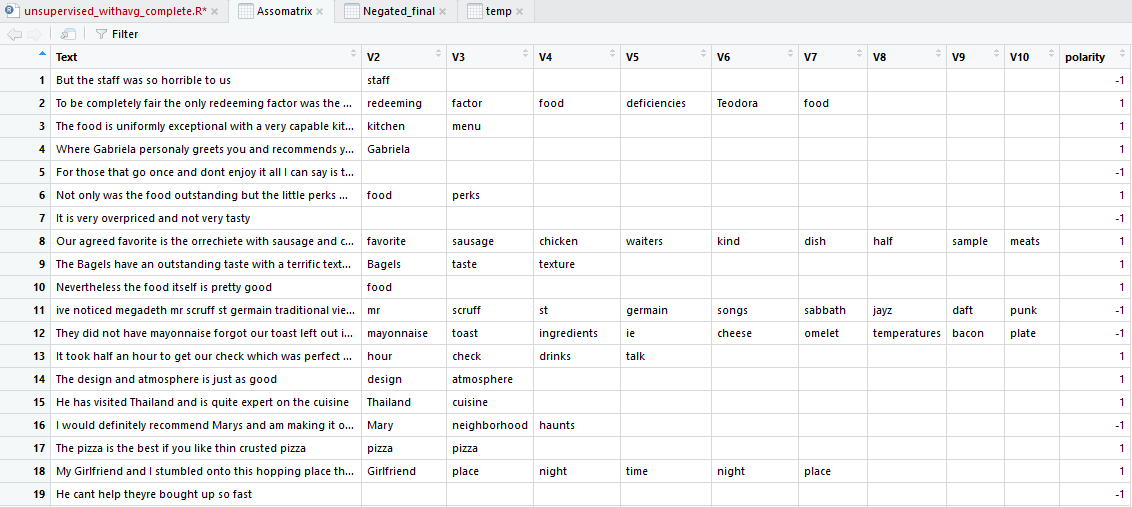


Fig.12: Polarity of each sentence

To find the accuracy of my model, we filter the testset to find the sentences for which we have the polarity .

Polarity in the test data could either be positive or negative. If the polarity is +ve then we convert it to 1 and -1 if it is –ve.

#To find the accuracy

doc <-docxml[which((docxml$aspectCategories.aspectCategory=="positive")|(docxml$aspectCategories.aspectCategory=="negative" )),]

#we only need first 5 col

docqw <-as.data.frame(doc[,(2:5)])

docqw$.attrs<- NULL

docqw$aspectTerm <- NULL

docqw$polarity <-0

#if negative then -1 and vis-a-vis

for(i in 1:nrow(docqw))

{

if(docqw$aspectCategory[i] == "negative")

docqw$polarity[i] = -1

else if(docqw$aspectCategory[i] == "positive")

docqw$polarity[i] = 1

}

docqw <- as.data.frame(unique(docqw))

docqw$text<-as.character(docqw$text)

#remove puntuations

for (i in 1:nrow(docqw))

docqw$text[i]<- gsub("[[:punct:]]"," ", docqw$text[i])

matching <- 0

for (i in 1:nrow(docqw))

for(j in 1:nrow(Assomatrix))

{

if(identical(docqw$text[i],Assomatrix$Text[j]))

if(docqw$polarity[i]== Assomatrix$polarity[j])

matching = matching +1

}

# matched 1451 out of 1892 = 0.766

**7.2 Supervised Approach**

We will now try to use a supervised approach and train a model to predict the polarity of the test data.

Supervised Approach has 4 Subtasks:

**7.2.1 Opinion Target Extraction**

The objective of OTE slot is to extract all opinion target expressions in a restaurant review, OTE could be a word or multiple words.

For this purpose, we have used CRF (Conditional Random Field) which have proved its performance in information extraction.

Eg. In the context “food was good” , food is the Opinion Target Word.

for(i in 1:nrow(sentence))

{

temp\_bind <- as.data.frame(unlist(strsplit(sentence$text[i], ",")))

if(i==1)

sent\_split <- temp\_bind

else

sent\_split <- rbind(sent\_split,temp\_bind)

}

names(sent\_split)[1]<- c("text")

#to find context and string

sent\_split$noun <-0

#remove noun which are same as in lexicon

noun<- noun[-which(noun$word %in% lexicon$word),]

#remove stopwords

noun$word = removeWords(noun$word, stopwords("english"))

#find context

for(i in 1:nrow(sent\_split))

for(j in 1:nrow(noun))

{

if(noun$word[j] %in% tolower(unlist(strsplit(sent\_split$Text[i]," "))))

sent\_split$noun[i] <- c(noun$word[j])

}

sent\_split<- sent\_split[which(sent\_split$noun == 1 ),]

sent\_split<- sent\_split[-which(sent\_split$noun==""),]

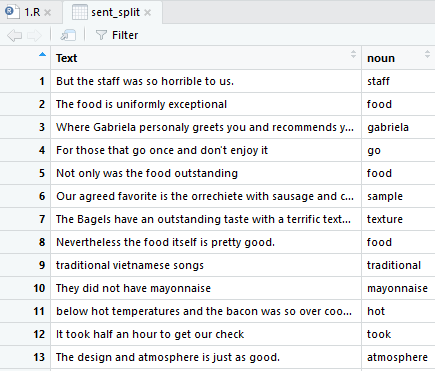


Fig.13: Context – Opinion Target word Extraction

#to remove/NNP , /NN and so on ....

words$word <- as.character(words$word)

for(i in 1:nrow(words))

words$word[i]<- gsub("/+[[:alpha:]]\*","",words$word[i])

for(i in 1:nrow(words))

words$word[i]<- gsub("[[:punct:]]","",words$word[i])



Fig.14:POS tagged words

**7.2.2 Feature Extraction ( Sentiment Polarity )**

**7.2.2.1 Word N-Grams**

We find Unigrams and Bigrams for each context.

#to make ngrams of 2

library("quanteda")

words2 <- as.data.frame(sent\_split$Text %>%

tokens(sent\_split$Text, what = "sentence") %>%

as.character() %>%

tokens(ngrams = 2, remove\_punct = TRUE) %>%

as.character())

names(words2)[1]<-c("word")

#to replace the " " for"\_"

for(i in 1:nrow(words2))

words2$bigrams[i]<- gsub("\_"," ",words2$bigrams[i])

words<- rbind(words,words2)

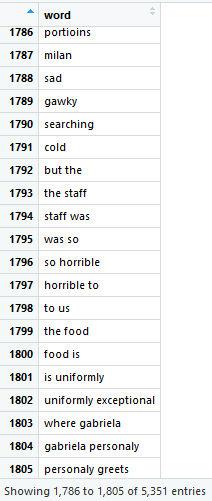


Fig.15:N-Gram DataFrame

**7.2.2.2 Document Term Frequency Extraction**

master <- matrix(0, nrow=nrow(words), ncol=nrow(train500))

for(i in 1:nrow(train500))

{

for(j in 1:1791)

{

master[j,i] <- length(which(unlist(strsplit(tolower(train500$Text[i])," "))== words$word[j]))

}

for(j in 1792:nrow(words))

{ master[j,i] <- str\_count(tolower(train500$Text[i]), words$word[j])}

}

#made head the subset of master since the data is too big

head<- as.data.frame(master[c(1:nrow(master)),c(1:10)])

master <- t(master)

head <- as.data.frame(master)

row.names(master)<-words$word

names(head)<-seq(length=nrow(train500))

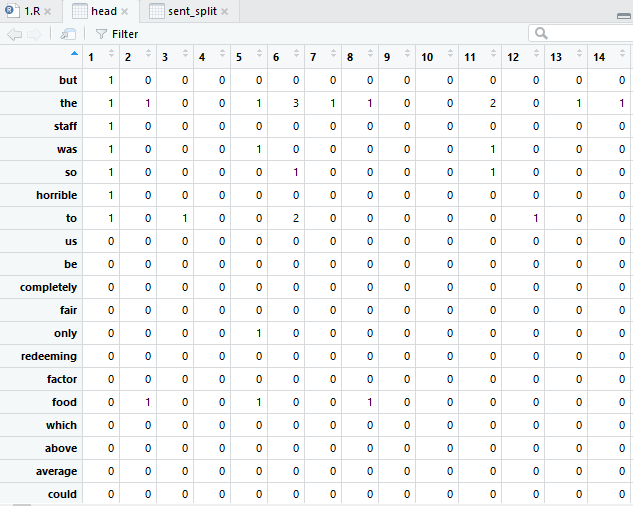


Fig.16:Term-Document Frequency Matrix

**7.2.2.3 Z Score Features**

for( i in 1:ncol(head))

zscore[i]<-sum(zscore(head[i,]))

#zscore is saved in master$zscore

for(i in 1: nrow(master))

master$zscore[i]<- zscore[i]

head <- as.data.frame(t(master))

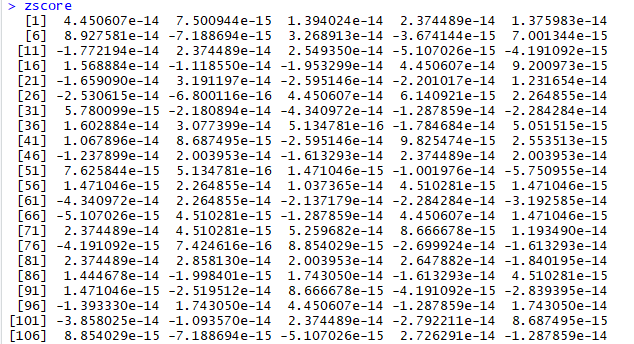


Fig.17: Z-Score for each context

**7.2.2.4 Sentiment Lexicon Based Features**

For each context the number of positive words, the number of negative ones, the number of positive words divided by the number of negative ones and the polarity of the last word are extracted from manual constructed lexicons.

sentlex <- matrix(0,500,4)

names(sentlex)<- c("+","-","+/-","last\_polarity")

#count +ve words using lexicon

# for(i in 1:nrow(train500))

# sentlex[i,1]<- length( which(lexicon$word %in% unlist(strsplit(tolower(train500$Text[i])," "))))

for(i in 1:nrow(train500))

sentlex[i,1]<-sum(lexicon$value[which(lexicon$word %in% unlist(strsplit(tolower(train500$Text[i])," ")))] > 0)

#count -ve words using lexicon

for(i in 1:nrow(train500))

sentlex[i,2]<-sum(lexicon$value[which(lexicon$word %in% unlist(strsplit(tolower(train500$Text[i])," ")))] < 0)

#work on master and then transform to head to view the data

#+/-

for(i in 1:nrow(sentlex))

{

sentlex[i,3]<-sentlex[i,1]/sentlex[i,2]

if(sentlex[i,3] == "NaN" |sentlex[i,3] == "Inf" )

sentlex[i,3]<-0

}

#polarity of the last word

for(i in 1:nrow(sentlex))

{

sentarr <- unlist(strsplit(tolower(train500$Text[i])," "))

arr <- which(unlist(strsplit(tolower(train500$Text[i])," ")) %in% lexicon$word)

last <- sentarr[arr[length(arr)]]

if(!is.integer0(lexicon$value[which(lexicon$word %in% last)]))

sentlex[i,4]<- lexicon$value[which(lexicon$word %in% last)]

else

sentlex[i,4]<-0

}

#polarity of each sentence

for(i in 1:nrow(sentlex))

{

if(sentlex$`+`[i] - sentlex$`-`[i] >0)

sentlex$polarity[i] <- 1

else if((sentlex$`+`[i] - sentlex$`-`[i] < 0))

sentlex$polarity[i]<- -1

}

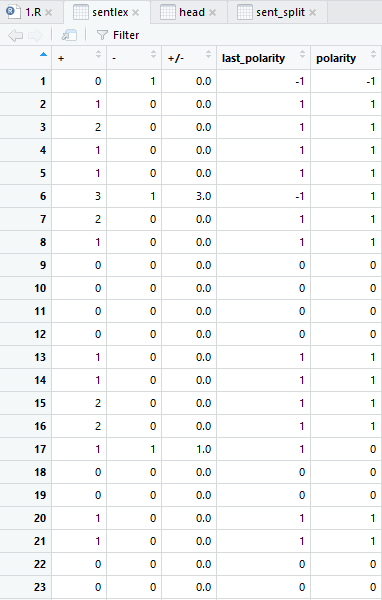
****

Fig.18:Sentlex Dataframe Snapshot

**7.2.3 Model Generation**

We have 2 sets of data , training data and testing data.

traindata <- as.data.frame(train500[1:400,c(1,3)])

testdata <- as.data.frame(train500[401:500,c(1,3)])

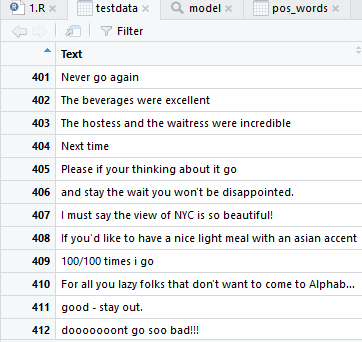


Fig.19: Testdata Dataframe Snippet

# SEPARATE TEXT VECTOR TO CREATE Source(),

# Corpus() CONSTRUCTOR FOR DOCUMENT TERM

# MATRIX TAKES Source()

trainvector <- as.vector(traindata$Text)

testvector <- as.vector(testdata$Text)

# CREATE SOURCE FOR VECTORS

trainsource <- VectorSource(trainvector)

testsource <- VectorSource(testvector)

# CREATE CORPUS FOR DATA

traincorpus <- VCorpus(trainsource)

testcorpus <- VCorpus(testsource)

# PERFORMING THE VARIOUS TRANSFORMATION on "traincorpus" and "testcorpus" DATASETS #SUCH AS TRIM WHITESPACE, REMOVE PUNCTUATION, REMOVE STOPWORDS.

traincorpus <- tm\_map(traincorpus,stripWhitespace)

traincorpus <- tm\_map(traincorpus,tolower)

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

traincorpus<- tm\_map(traincorpus,removePunctuation)

traincorpus <- tm\_map(traincorpus, PlainTextDocument);

testcorpus <- tm\_map(testcorpus,stripWhitespace)

testcorpus <- tm\_map(testcorpus,tolower)

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

testcorpus<- tm\_map(testcorpus,removePunctuation)

testcorpus <- tm\_map(testcorpus, PlainTextDocument);

# CREATE TERM DOCUMENT MATRIX

trainmatrix <- t(TermDocumentMatrix(traincorpus))

testmatrix <- t(TermDocumentMatrix(testcorpus))

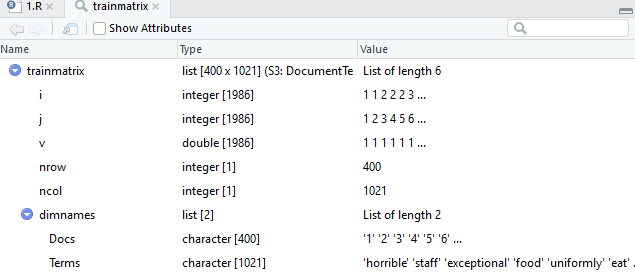


Fig.20: Trainmatrix snippet

# TRAIN NAIVE BAYES MODEL USING trainmatrix DATA AND traindata$polarity CLASS VECTOR

model <- naiveBayes(as.matrix(trainmatrix),as.factor(traindata$polarity))

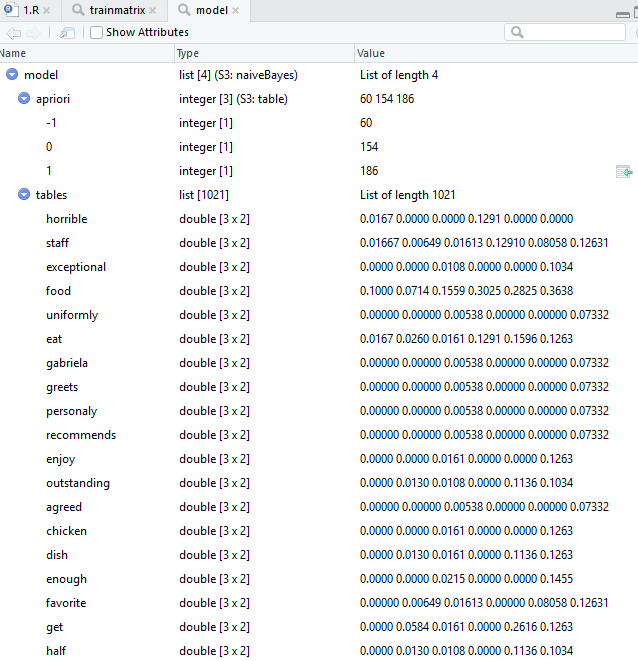


Fig.21: Dataframe snapshot of the Model

# PREDICTION

log\_predict <- predict(model,as.matrix(testmatrix))

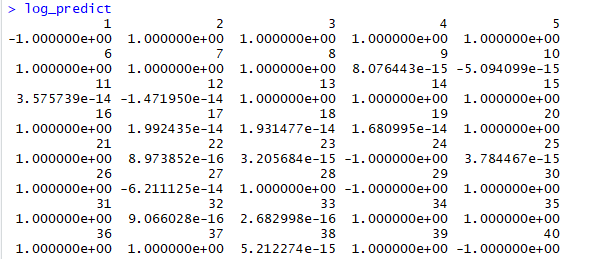


Fig.22: Results of predict()

The log\_predict function returns -1 or 1 for positive and negative polarity and -Inf<x<-1 or 1<y<Inf for incorrect responses.

# matched 81 out of 100 = 0.81

**8. PROJECT SCHEDULE**

* *January 2018*
* Study Part of speech Tagging in NLP
* *February 2018*
* Implementation of unsupervised ABSA NLP program
* *March 2018*
* Implementation of supervised ABSA NLP program
* Documentation
* *April 2018*
* Submission of report & evaluation
* Study DeepLearning Technology and begin implementation
* *May 2018*
* ABSA implementation using a DeepLearning Approach ( TensorFlow)
* *June 2018*
* Submission of report & evaluation  
    
    
  **9.REFERENCES**
* M. Hu and B. Liu, “Mining and summarizing customer reviews”. Proceedings of the 10th KDD, pp. 168–177, Seattle, WA, 2004.
* Ankit Singh, Enayat Ullah , “Aspect based Sentiment Analysis”. DOI: <https://cse.iitk.ac.in/users/cs365/2015/_submissions/enayat/report.pdf>
* Semeval 2014. “SemEval-2014 Task 4”. In: (2014). DOI: <http://alt.qcri.org/semeval2014/task4/>
* Hussam Hamdan,Patrice Bellot and Frederic Bechet ,”Lsislif: CRF and Logistic Regression for Opinion Target Extraction and Sentiment Polarity Analysis“. DOI: <http://anthology.aclweb.org/S/S15/S15-2128.pdf>
* Samuel Brody and Noemie Elhadad,” An Unsupervised Aspect-Sentiment Model for Online Reviews”. DOI: <http://people.dbmi.columbia.edu/noemie/papers/naacl10.pdf>

PROJECT DETAILS

|  |  |  |  |
| --- | --- | --- | --- |
| *Student Details* | | | |
| **Student Name** | **SHASHWAT GUPTA** | | |
| Register Number | 140905047 | Section / Roll No | C - 06 |
| Email Address | [Shashwat2211@gmail.com](mailto:Shashwat2211@gmail.com) | Phone No (M) | 9620193750 |
|  | | | |
| *Project Details* | | | |
| **Project Title** | Aspect based Sentiment Analysis using NLP (R) and Deep Learning | | |
| Project Duration | 5 Months | Date of reporting | 22 - Jan - 2018 |
|  |  | | |
| *Organization Details* | | | |
| **Organization Name** | **Impetus Infotech India Pvt. Ltd.** | | |
| Full postal address with pin code | SDF No. K-13 to 16, NSEZ, Phase-II, NOIDA - 201305 (U.P.) India | | |
| Website address | Impetus.com | | |
|  |  | | |
| *External Guide Details* | | | |
| **Name of the Guide** | **Mr. Vivek Gupta** | | |
| Designation | Senior Analytics Engineer | | |
| Full contact address with pin code | Data Science Practice , SDF No. K-13 to 16, NSEZ, Phase-II, NOIDA - 201305 (U.P.) India | | |
| Email address | [Vivek.gupta@impetus.co.in](mailto:Vivek.gupta@impetus.co.in) | Phone No (M) | 9910464101 |
|  |  | | |
| *Internal Guide Details* | | | |
| **Faculty Name** | **Mrs. Archana Praveen Kumar** | | |
| Full contact address with pin code | Dept of Computer Science & Engg, Manipal Institute of Technology, Manipal – 576 104 (Karnataka State), INDIA | | |
| Email address | [Archana.kumar@manipal.edu](mailto:Archana.kumar@manipal.edu) | | |