Text Analytics - Amazon Fine Food Reviews

AMRITA SHARMA &
PREETHI RANGANATHAN

Table of Contents

Dataset Overview

Dataset consists of 568,452 Amazon Fine Food Reviews.
Attributes:

- Review id
- Product Category Id
- User Id
- User profile name
- Number of users who voted for the helpfulness survey
- Number of users who found the review helpful
- Rating (1-5)
- Review Date & Time
- Review
- Review Title

Some sample reviews

Positive review

Text
1:
I have bought several of the Vitality canned dog food products and have found them all to be of good quality. The product looks more like a stew than a processed meat and it smells better. My Labrador is finicky and she appreciates this product better than most.

Negative Review

Definitely not worth buying flavored water with a few teaspoons of beans and rice that doesn't taste like normal beans and rice. I wont ever buy this again!

Neutral Review

It's great to have agave in a portable format. But is is difficult to open. The directions say to pinch to open. They are not so easy to pinch. I have found if you bend the tube about 1 inch above the end then pinch the end it helps. But the agave often gets on your fingers.

Data Cleanup - Spam Review Identification

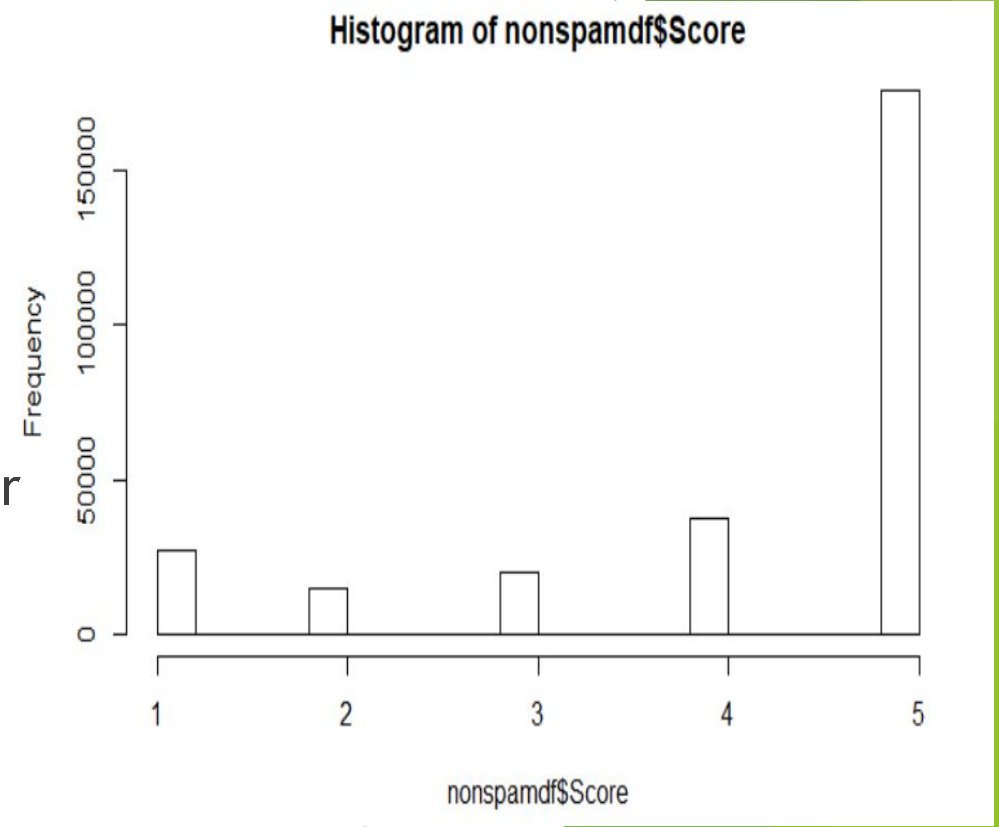
- Fake reviews
- To manipulate online customers’ opinions on products being sold
- Usually provided by bots
- To make their own business look good or damage someone else’s business
- Multiple reviews given by the same user exactly at the same time

Business Question 1:

What are the 3 most important aspects focused in bad reviews?

Dataset Preparation

- User Id over Profile Name
- Removed Null entries
- Removed repeated observations
- Removed entries where number of people who found the review helpful was greater than total surveyed
- Bad Ratings - 1 & 2 Ratings



Model Analysis - Topic Modelling

- Topic models provide a simple way to extract topics and themes from large volumes of unlabeled text.
- Latent Dirichlet Allocation(LDA)
- Unsupervised Learning Technique
- Mallet LDA package
- MALLET is a Java-based package for statistical natural language processing, document classification, clustering, topic modeling.
- Efficient Parallel Processing

Topic Modelling on Bad Reviews

Topic Modelling on text of bad reviews

```
```{r}
lower_text =tolower(baddf$Text)
ctext = Corpus(VectorSource(lower_text))
rm(data)

mallet.instances <- mallet.import(as.character(seq(1:length(lower_text))),
                                  lower_text,
                                  "/C:/Amrita/HardDisk/SpringTerm2018/MachineLearning/stopWords.txt")

topic.model <- MalletLDA(num.topics=3)
topic.model$loadDocuments(mallet.instances)
topic.model$setAlphaOptimization(20, 100) # optimise parameters after every 20 iterations which will be preceeded by 100 burnin
topic.model$train(1000) # train the model
topic.model$maximize(10)
doc.topics <- mallet.doc.topics(topic.model, smoothed=T, normalized=T)
topic.words <- mallet.topic.words(topic.model, smoothed=T, normalized=T)
topics.labels <- rep("",3)
for (topic in 1:3) topics.labels[topic] <- paste(mallet.top.words(topic.model, topic.words[topic,], num.top.words=80)$words,
collapse=" ")
topics.labels

```
```

Common Reasons of Complaints - Bad Reviews

- Health effects of defective food products.
- Misleading/deceptive products, shipping, packaging services, damaged and expired products.
- Taste, flavor and smell aspects of the product.

Insights /Suggestions

- Scrutinize the vendors who are shipping defective products to improve customer experience
- Improve the delivery distribution channel and also analyze the underlying root causes of why the goods are damaged
- Packaging standards should also be improved for perishable/fragile items

Business Question 2:

What are the most important characteristics of a Helpful Review?

Why?

- 85% of customers trust online reviews as much as a personal recommendation
- Hence measuring helpfulness goes a long way in developing the trust of a customer
- Idea behind the question is to develop a model which would identify if a review would be helpful or not

Dataset Preparation...

- Helpfulness Score - Metric to measure the helpfulness of each review
- Helpfulness Score = $\dfrac{\text{Helpful}}{\text{Helpful} + \text{Unhelpful}}$

- Ranges from 0 to 1.
- Created a Binary Variable with threshold as 0.5
- Reviews which didn't have a helpfulness voting
  - Removed those observations
  - Created a new category by considering these reviews as neutral

Features

- Word Count
- Sentence Count
- Rating
- Sentiment Score
- Readability Score
- Similarity Score

Sentiment Score

- Analyze whether strong sentiments make a review helpful or not
- SentimentR package
- Calculated Polarity Score for each review which ranges from -1 to +1

```{r}
library(sentimentr)
sentimentdf <- with(nonspamdf, sentiment_by(get_sentences(Text), list(Id)))
write.csv(sentimentdf, file = "sentimentdf.csv",row.names=TRUE, na="")
```

Readability Index

- The index estimates the years of formal education needed to understand text on a first reading
- Readability package
- Gunning Fog Index per review
- Index ranges from 1-100.(Hard-Easy)
- Reviews with high index are easier to read than reviews with low

```
library(readability)
readable <- with(nonspamdf, readability(Text, list(Id)))
write.csv(readable, file = "readability.csv",row.names=TRUE, na="")
```

Similarity Score

- Lexical similarity which determines how close a review is to the already identified helpful features
- Extracted top 200 words that occur the most in a helpful review
- How helpful is our review based on comparing each review to these extracted features

Similarity Score…

```r
# Taking out the helpful reviews
helpfulreviews=filtered[filtered$HelpfulnessScore_bin==1,]
summary(helpfulreviews$Text)

# Top 200 words
lower_text =tolower(helpfulreviews$Text)
ctext = Corpus(VectorSource(lower_text))
ctext_nopunc_nonum = tm_map(ctext, removeNumbers)
ctext_nopunc = tm_map(ctext_nopunc_nonum, removePunctuation)
ctext_nopunc_nonum_nostop = tm_map(ctext_nopunc,removeWords, c("shall","us","unto","will","just","nothing","can"
,"much","dont","didnt","doesnt","never",  "upon","also","let","even","now","yet", "therefore","may","away","since","nothing",
stopwords("english")))

tdm2 = TermDocumentMatrix(ctext_nopunc_nonum_nostop,control=list(wordLengths=c(4, 15),
                                   bounds = list(global = c(50,Inf))))
tdm3 = as.matrix(tdm2)
wordcount = sort(rowSums(tdm3),decreasing=TRUE)
tdm_names = names(wordcount)[1:200]
wordcloud(tdm_names,wordcount)

# Similiarity Calculation
m = length(filtered$Text)  # No of sentences in input
text=filtered$Text
jaccard = matrix(0,m,1)  #Store match index
b = tdm_names ; bb = unlist(strsplit(b," "))
for (i in 1:m) {
        a = text[i]; aa = unlist(strsplit(a," "))
         jaccard[i]  = length(intersect(aa,bb))/
                          length(union(aa,bb))
}
filtered$SimiliarityScore=jaccard
```

Approach 1: Binary Classification

Model	Accuracy	Precision	Recall	F-1 Score
LDA	76.9%	90.6%	81.6%	85.8%
Logistic Regression	77.3%	93.2%	80.6%	86.4%
KNN	69.6%	80.1%	80.7%	80.4%
Naive Bayes	76.3%	88.2%	82.4%	85.2%
XGboost	77.7%	95.6%	79.6%	86.9%

Binary Classification Results

```
Call:
glm(formula = y_train ~ Gunning_Fog_Index + word_count + ave_sentiment +
    Score + SimiliarityScore, family = binomial(link = "logit"),
    data = x_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4436   0.4391   0.5464   0.6166   1.3949

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)      -0.9332187  0.0289627 -32.221  < 2e-16 ***
Gunning_Fog_Index  0.0027711  0.0023980   1.156    0.248
word_count         0.0001410  0.0001264   1.115    0.265
ave_sentiment    -0.1937607  0.0408331  -4.745 2.08e-06 ***
Score             0.4877383  0.0052698  92.554  < 2e-16 ***
SimiliarityScore 11.3652526  0.5780690  19.661  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 127577  on 119828  degrees of freedom
Residual deviance: 115018  on 119823  degrees of freedom
AIC: 115030

Number of Fisher Scoring iterations: 4
```

Approach 2: Multinomial Classification Prep

- Reviews which didn't have a helpfulness voting are categorized in the neutral category
- 0 - Not Helpful
- 1 - Helpful
- 2 - Neutral
- Same features as binomial classification

Multinomial Classification

Model	Accuracy	Precision	Recall	F-1 Score
LDA	48.6%	27.9%	49.2%	35.6%
KNN	44.1%	44.1%	44.2%	44.2%
Multinomial Regression	49.0%	30.2%	48.6%	37.3%
Naive Bayes	48.4%	20.5%	50.7%	29.2%
XGBoost	50.2%	46.3%	40.7%	40.9%

Insights

- Surprisingly, the number of words and sentences used in a review doesn’t have an impact on the degree of helpfulness
- Reviews with strong negative sentiments are more helpful as compared to strong positive sentiments.
- Reviews containing more words which describe the product and packaging attributes and provide suggestions to the customers are more helpful

Thank you