# Project2-HarvardX-Capstone-Course.R

alevcieslinski

2022-03-09

```r
#An Introduction to Topic Modeling and Sentiment Analysis:
#Amazon Food Reviews Data
#This analysis is based on product reviews from Amazon. The original dataset is
#quite large with the following statistics:
#Number of reviews  568,454
#Number of users    256,059
#Number of products 74,258
#I ran my analysis on the whole dataset, but R was very slow, so I decided to
#sample by productid and use the smaller dataset to complete this project.  I
# also brought in a subset of the original dataset to make R coding more
#efficient.
#The purpose of this analysis is to do text analytics and topic modeling on a
#sample of Amazon product reviews, mainly food reviews.  I also conducted
#sentiment analysis.

#Install and load packages into R
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(tidytext)
library(tidyverse) # metapackage of all tidyverse packages
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.1.6     v purrr   0.3.4
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(lubridate)
```

```
##
```

```
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(lda)
library(topicmodels)
library(tm)
```

```
## Loading required package: NLP

##
## Attaching package: 'NLP'

## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(widyr)
library(Matrix)
```

```
##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
library(textdata)
library(syuzhet)
library(readxl)
library(latexpdf)
library(tinytex)
theme_set(theme_light())

#load dataset into R and eliminate null reviews, assign lower cases to variables
reviews <- read_excel("/Users/alevcieslinski/Downloads/reviews3.xls")
reviews <- na.omit(reviews)
reviews <- reviews %>%
  rename_with(str_to_lower)

#Check the structure of data
str(reviews)
```

```
## tibble [5,018 x 3] (S3: tbl_df/tbl/data.frame)
##  $ productid: chr [1:5018] "6641040" "6641040" "6641040" "6641040" ...
```

```
##  $ text     : chr [1:5018] "A charming, rhyming book that describes the circumstances under which you
##  $ score    : num [1:5018] 4 4 5 5 4 5 5 1 4 5 ...
```

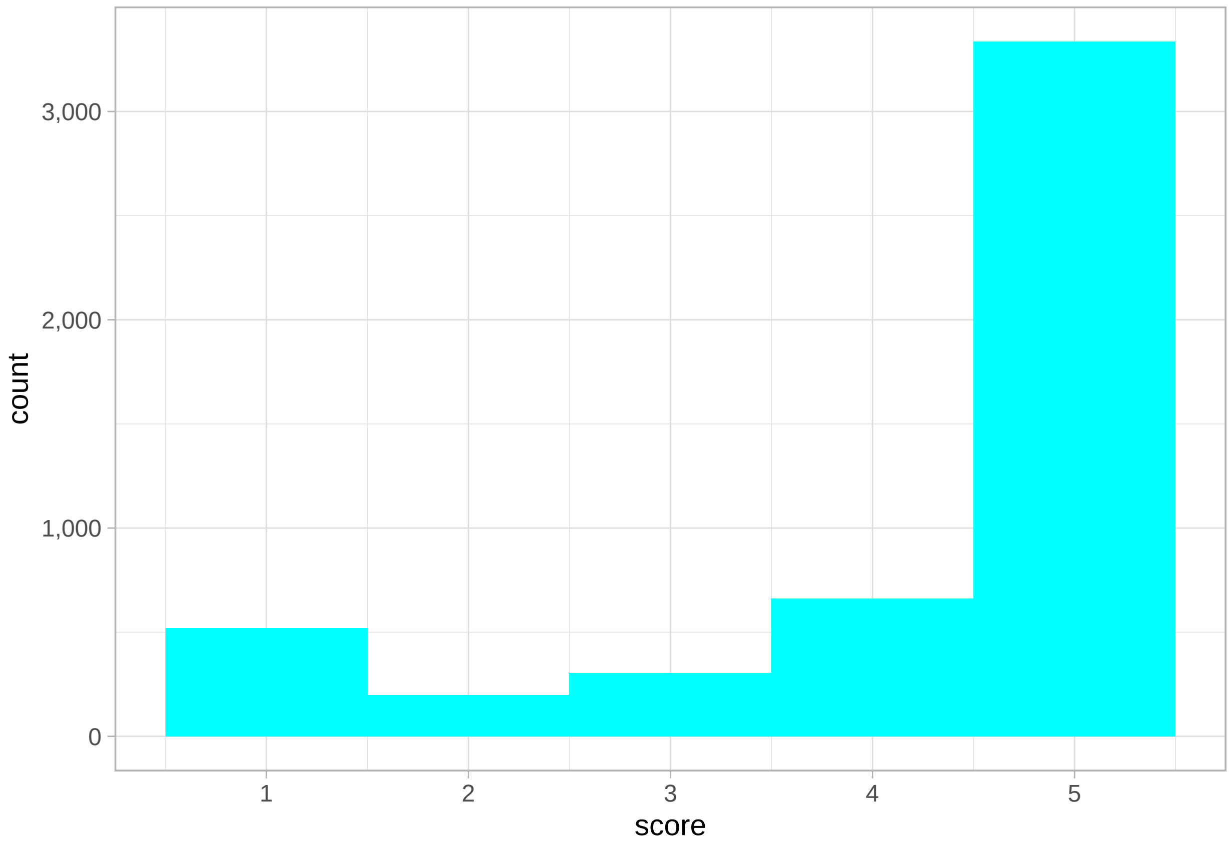
```
#data cleanup
reviews$text <- gsub("<[^>]+>", "", reviews$text)

#summary data. I brought in productid, text(review), and score that customers
#assigned products.

reviews %>% summarize(unique_products = length(unique(productid)),
                      unique_text = length(unique(text)),
                      unique_score = length(unique(score)))
```

```
## # A tibble: 1 x 3
##   unique_products unique_text unique_score
##             <int>       <int>        <int>
## 1             700        4107            5
```

```
#count by score (customer review score)
reviews %>%
  count(score, sort=TRUE)
```

```
## # A tibble: 5 x 2
##   score     n
##   <dbl> <int>
## 1     5  3336
## 2     4   661
## 3     1   520
## 4     3   304
## 5     2   197
```

```
#visual representation of score distribution
reviews %>%
  ggplot(aes(score)) +
  geom_histogram(fill="cyan", binwidth=1) +
  scale_y_continuous(labels = scales::comma)
```




```r
#tokenization, stopwords
#count words by score and productid
#In order to do the text analysis and topic modeling, we have to create words
#from each line of text.
food_reviews <- reviews %>%
  unnest_tokens(input=text, output=word) %>%
  anti_join(stop_words, by='word') %>%
  filter(str_detect(word, "[a-z]")) %>%
  count(productid, word, score, sort=TRUE) %>%
  ungroup()

total_words <- food_reviews %>%
  group_by(productid) %>%
  summarize(total = sum(n))

food_reviews <- left_join(food_reviews, total_words)
```

```
## Joining, by = "productid"
```

```r
#tf_idf
#The statistic tf-idf is intended to measure how important a word is to
#a document in a collection (or corpus) of documents. In this case, we're
#looking at words by productid.

food_reviews <- food_reviews %>%
  bind_tf_idf(word, productid, n) %>%
  arrange(desc(tf_idf))
glimpse(food_reviews)
```

```
## Rows: 97,158
## Columns: 8
## $ productid <chr> "B0000DD8RB", "B0000DD8RB", "B0000DG56I", "B0000D956L", "B00~
## $ word      <chr> "popper", "spits", "spicey", "candies", "stomach.so", "valle~
## $ score     <dbl> 3, 3, 4, 4, 2, 5, 5, 5, 5, 5, 1, 1, 4, 1, 5, 5, 5, 5, 5, 5, ~
## $ n         <int> 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 9, 1, 1, 1, 1, 1, 1, ~
## $ total     <int> 4, 4, 4, 5, 5, 5, 11, 4, 5, 3, 6, 6, 5, 55, 5, 7, 7, 7, 7, 7~
## $ tf        <dbl> 0.2500000, 0.2500000, 0.2500000, 0.4000000, 0.2000000, 0.200~
## $ idf       <dbl> 5.857933, 5.857933, 5.857933, 3.660709, 6.551080, 6.551080, ~
## $ tf_idf    <dbl> 1.4644833, 1.4644833, 1.4644833, 1.4642834, 1.3102161, 1.310~
```

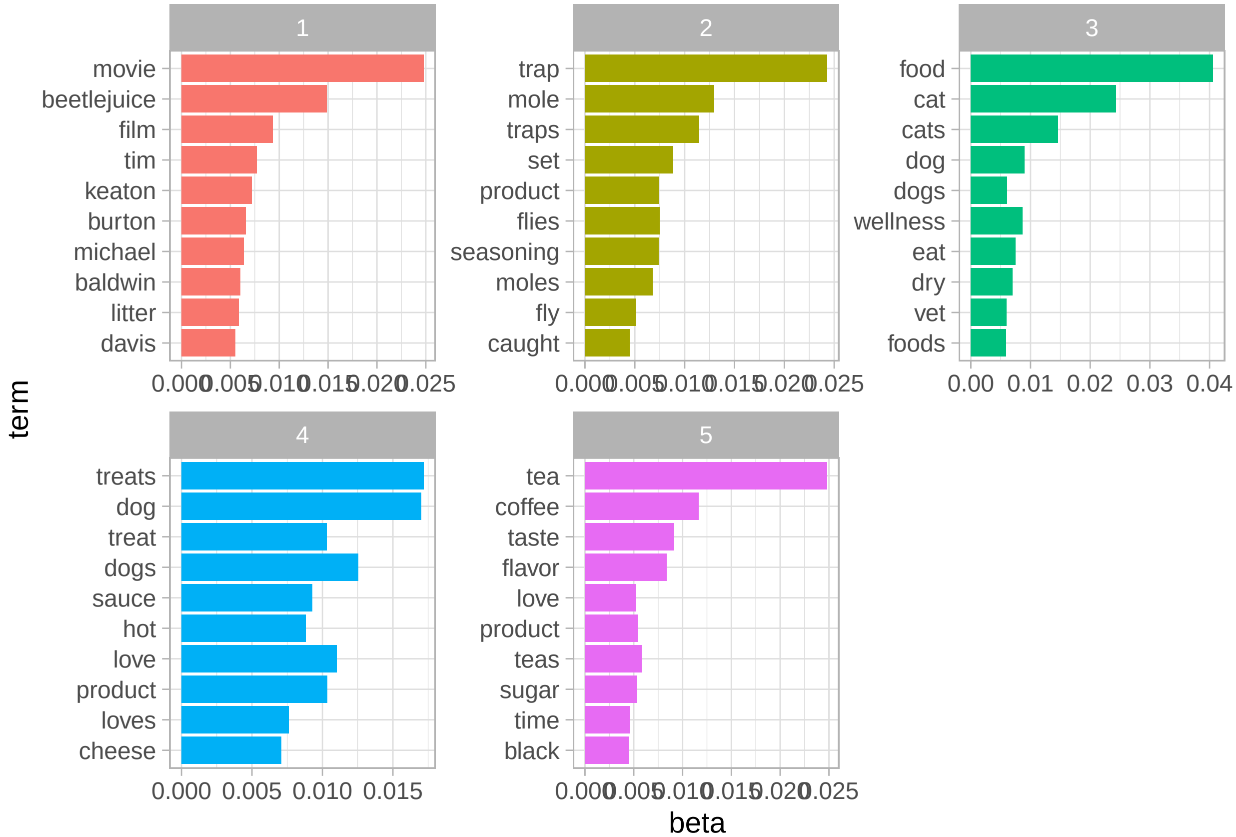
```
#The LDA() function operates on a Document Term Matrix (DTM), so we need to
#create a DTM from the reviews. The dataframe food_reviews containing all of
#the word counts by productid will then be cast into a document term matrix with
#the following code.

dtm <- food_reviews %>%
  cast_dtm(document=productid, term=word, value=n)
dtm
```

```
## <<DocumentTermMatrix (documents: 700, terms: 16790)>>
## Non-/sparse entries: 82193/11670807
## Sparsity           : 99%
## Maximal term length: 29
## Weighting          : term frequency (tf)
```

```
#When we do topic modeling, we need to set some parameters, i.e.,the number of
#topics to find in the corpus. For simplicity, we chose 5 topics.  LDA is a
#clustering algorithm.

lda <- LDA(dtm, k = 5, control = list(seed = 1968))
lda
```

```
## A LDA_VEM topic model with 5 topics.
```

```
#We can look at which words occur most commonly within each topic. Beta is
#equal to P(word|topic), the conditional probablity that the word is observed,
#given the topic.  The code below look at the top 10 words for each of the five
#topics modeled.  Even though the reviews are for food items, there are some
#records with other product reviews. Interestingly, topic modeling was able to
#detect reviews about movies and identify those texts as a distinct set of topic
#and words as shown by the first topic in the plot below.

review_topics <- tidy(lda, matrix = "beta")
review_top_terms <- review_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

review_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()
```




```
#The product id for the movie reviews is B00004CI84 and we can see the original
#texts for this productid below for further validation:

reviews %>% filter(reviews$productid == "B00004CI84")
```

```
## # A tibble: 189 x 3
##    productid  text                                                          score
##    <chr>      <chr>                                                         <dbl>
##  1 B00004CI84 "... this little gem was the beginning of a brief \"thing\"~     4
##  2 B00004CI84 "\"It keeps getting funnier every time I see it,\"  says Be~     5
##  3 B00004CI84 "\"Pee-Wee's Big Adventure\" was director Tim Burton's firs~     5
##  4 B00004CI84 "[[ASIN:B001AGXEAG Beetlejuice (20th Anniversary Deluxe Edi~     4
##  5 B00004CI84 "*** 1/2 stars rating for &quot;Beetlejuice&quot;. This mov~     3
##  6 B00004CI84 "Beetlejuice (20th Anniversary Deluxe Edition)I am very imp~     4
##  7 B00004CI84 "A cool movie with some very funny and amusing moments, a y~     4
##  8 B00004CI84 "a couple dies.they live in a house as spirits and then som~     4
##  9 B00004CI84 "A dead couple attempts to scare a modern family out of the~     5
## 10 B00004CI84 "A haunting movie in which everything is played for laughs.~     1
## # ... with 179 more rows
```

```
#summarize betas below
summary(review_topics)
```

```
##      topic        term                beta
##  Min.   :1   Length:83950       Min.   :0.000e+00
##  1st Qu.:2   Class :character   1st Qu.:0.000e+00
##  Median :3   Mode  :character   Median :0.000e+00
##  Mean   :3                      Mean   :5.956e-05
```

```
##   3rd Qu.:4                          3rd Qu.:3.348e-05
##   Max.   :5                          Max.   :4.057e-02
```

```
#While beta refers to the conditional probability of a word given a topic,
#gamma is the per-document likelihood of each topic. In the code below,
#for example, we see how the algorithm allocates topics across the first
#two productids.

review_documents <- tidy(lda, matrix = "gamma")
review_documents <- arrange(review_documents, document)
review_documents
```

```
## # A tibble: 3,500 x 3
##    document   topic   gamma
##    <chr>      <int>   <dbl>
##  1 141278509X     1 0.00134
##  2 141278509X     2 0.00134
##  3 141278509X     3 0.00134
##  4 141278509X     4 0.00134
##  5 141278509X     5 0.995
##  6 2734888454     1 0.00159
##  7 2734888454     2 0.00159
##  8 2734888454     3 0.00159
##  9 2734888454     4 0.994
## 10 2734888454     5 0.00159
## # ... with 3,490 more rows
```

```
# Compute perplexity score
#Perplexity score is a measure of how well the model predicts a sample.  The
#lower the score, the better the model is deemed to be.  Of course, 1 perplexity
#score is not that meaningful as we need a distribution of benchmarks to
#be able to determine the lowest score.

perplexity(object=lda, newdata=dtm)
```

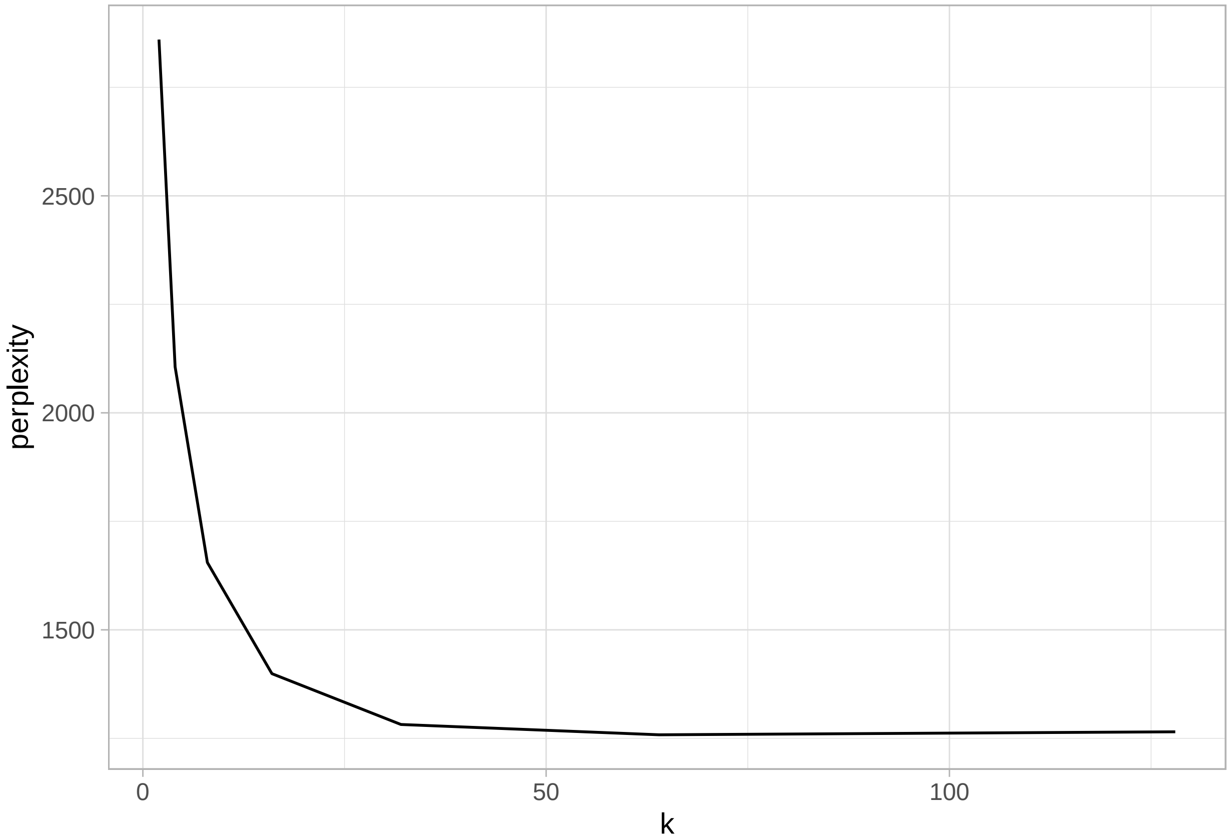
```
## [1] 1896.082
```

```
#Finding the optimal number of k scores
#create a dataframe to store the perplexity scores for different values of k
#As sated above, we create a set of perplexity scores based on different values
#of k to determine the best model.
p = data.frame(k = c(2,4,8,16,32,64,128), perplexity = NA)

# loop over the values of k in data.frame p
for (i in 1:nrow(p)) {
  print(p$k[i])
  #calculate perplexity for the given value of k
  m = LDA(dtm, method = "Gibbs", k = p$k[i],  control = list(alpha = 0.01))
  # store result in our data.frame
  p$perplexity[i] = perplexity(m, dtm)
}
```

```
## [1] 2
## [1] 4
## [1] 8
## [1] 16
```

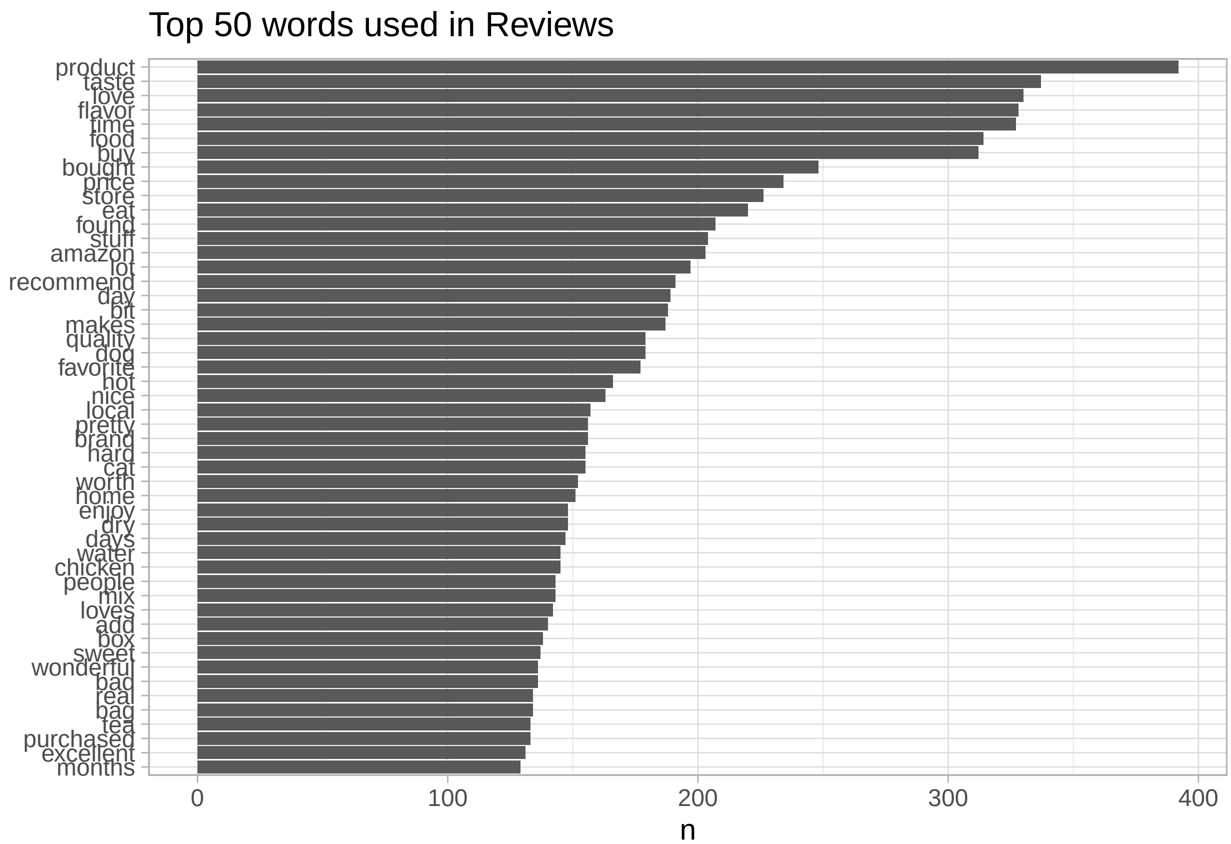
```
## [1] 32
## [1] 64
## [1] 128
```

```
#plot perplexity & values of k
ggplot(p, aes(x=k, y=perplexity)) + geom_line()
```




```
#we see that the perplexity score sharply declines for values of k greater
#than 25.

#wordcloud
#Next, we can create a wordcloud to see a visual depiction of relative
#importance of words by generating the counts of words in the corpus
word_frequencies <- food_reviews %>%
  count(word)

#Create the wordcloud
wordcloud(words=word_frequencies$word,
          freq=word_frequencies$n,
          min.freq=5,
          max.words=50,
          colors=c("DarkOrange","Blue"),
          scale=c(3,0.3))
```


```
#top 50 words that were used on product reviews
food_reviews %>%
  count(word, sort = TRUE) %>%
  head(50) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col() +
  xlab(NULL) +
  labs(title="Top 50 words used in Reviews") +
  coord_flip()
```




```
#While the wordcloud and the top 50 words graph is helpful in understanding
#what seems to be most used "word/s", it doesn't help us understand
#sentiments and emotions, which is what we will explore next.

#Sentiment analysis
#Most words by themselves do not describe sentiment. I used the sentiment
#function below to assign sentiments to the text from the "reviews" dataset.
#Once we run the code and look at s created by the get nrc sentiment , we see
#that each row represents a review and each column represents the different
#sentiments along with positive and negative score for that review. We also
#created a final review score that is represented below:

sentiment_data_all <- iconv(reviews$text)
s <- get_nrc_sentiment(sentiment_data_all)
```

```
## Warning: `spread_()` was deprecated in tidyr 1.2.0.
## Please use `spread()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

```
s$score <- s$positive - s$negative
head(sentiment_data_all)
```

```
## [1] "A charming, rhyming book that describes the circumstances under which you eat (or don't) chicken
## [2] "A very entertaining rhyming story--cleaver and catchy.The illustrations are imaginative and fit
## [3] "All of my children love this book.  My first grader got it for Christmas and loves to read the 
## [4] "Classic children's book, can't go wrong. I read it when i was a kid and ordered it 20yrs later.
## [5] "Get the movie or sound track and sing along with Carol King. This is great stuff, my whole exten
```

```
## [6] "Great book, perfect condition arrived in a short amount of time, long before the expected delive
```

```r
s[1:10,]
```

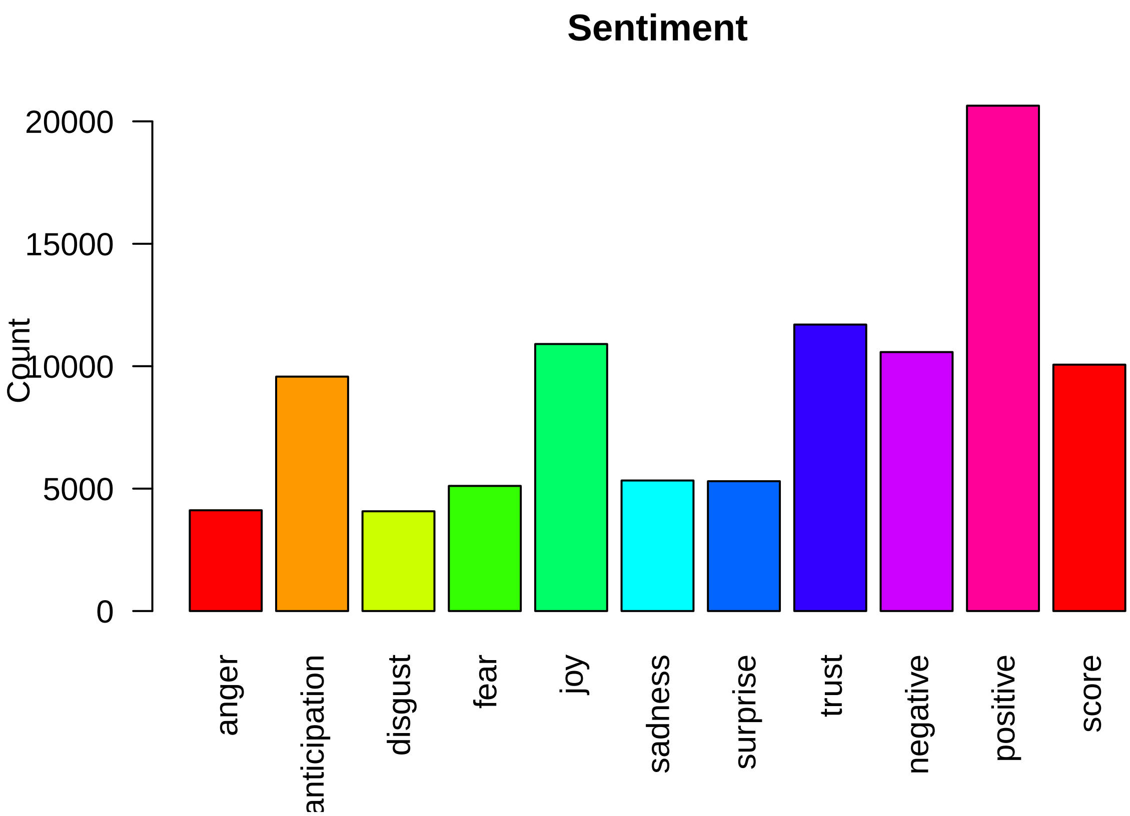
```
##    anger anticipation disgust fear joy sadness surprise trust negative positive
## 1      1            1       0    2   2       2        0     2        1        6
## 2      0            2       0    0   1       0        0     0        1        2
## 3      0            2       1    1   4       1        1     5        1        9
## 4      0            0       0    0   0       0        0     0        1        1
## 5      0            2       0    0   3       2        0     2        0        4
## 6      0            5       0    0   1       0        0     1        0        2
## 7      0            1       0    0   1       0        0     3        0        1
## 8      1            5       1    0   2       2        1     3        2        4
## 9      0            0       0    0   2       0        0     2        0        3
## 10     0            1       0    0   4       1        1     1        0        4
##    score
## 1      5
## 2      1
## 3      8
## 4      0
## 5      4
## 6      2
## 7      1
## 8      2
## 9      3
## 10     4
```

```r
#check overall sentiment and represent it visually with a bar plot.
review_score <- colSums(s[,])
print(review_score)
```

```
##        anger anticipation      disgust         fear          joy      sadness
##         4115         9574         4074         5110        10905         5331
##     surprise        trust     negative     positive        score
##         5302        11700        10577        20640        10063
```

```r
#Bar plot
barplot(colSums(s),
        las = 2,
        col = rainbow(10),
        ylab = 'Count',
        main = 'Sentiment')
```

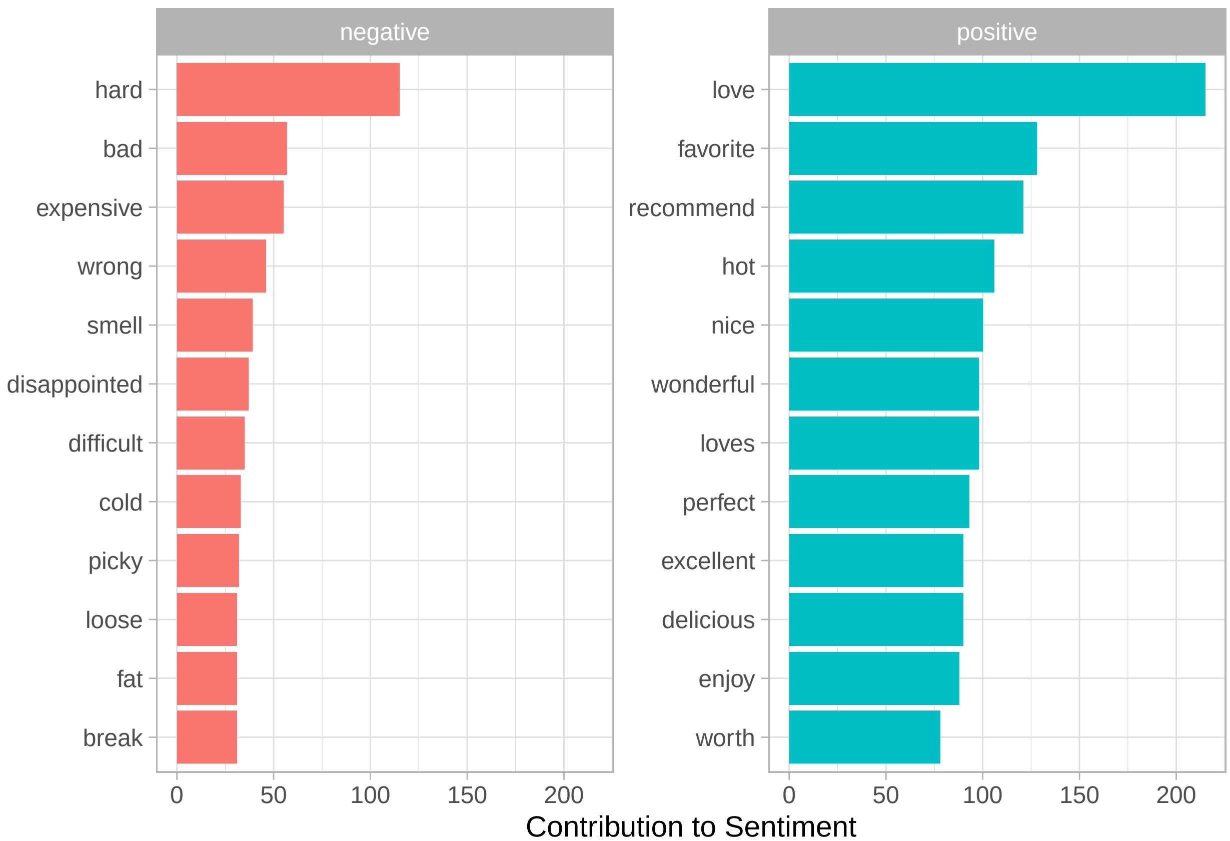
| productid | negative | positive | sentiment |
|---|---|---|---|
| 141278509X | 0 | 1 | 1 |
| 2734888454 | 2 | 4 | 2 |
| 2841233731 | 0 | 3 | 3 |
| 6641040 | 32 | 68 | 36 |
| 7310172001 | 144 | 143 | -1 |
| 7310172101 | 144 | 143 | -1 |




```r
#sentiment score by productid
#Next, we can look at the sentiment score by product id and also see a visual
#representation of how top words contribute to "negative" and "positive"
#emotions.
sentiment_data <- food_reviews %>%
  inner_join(get_sentiments("bing"), "word") %>%
  count(productid, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)

head(sentiment_data)%>%
  kable() %>%
  kable_styling(bootstrap_options = "basic", full_width = F)
```

```r
sentiment_analysis_word_count <- food_reviews %>%
  inner_join(get_sentiments("bing"), "word") %>%
  count(word, sentiment, score, sort = TRUE) %>%
  ungroup()
```

```r
sentiment_analysis_word_count %>%
  group_by(sentiment) %>%
  top_n(12, n) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to Sentiment", x = NULL) +
  coord_flip()
```




```r
#Finally, I looked at the avg. score (as provided by the initial review dataset)
#to see whether there was a significant difference between
#"positive" and "negative" sentiments. As expected, the "negative" category
#was lower than the "positive" category, but by a small margin.

#Avg. customer score by sentiment
sentiment_analysis_word_count %>%
  group_by(sentiment) %>%
  summarise_at(vars(score), list(name = mean))
```

```
## # A tibble: 2 x 2
##   sentiment  name
##   <chr>     <dbl>
## 1 negative   3.42
## 2 positive   3.67
```