Text Mining Sentiment from Hotel Reviews

Ian Jeffries

5/21/2019

The following R Markdown outlines the steps used by Ian Jeffries to mine sentiment from over 21,000 hotel reviews on resorts located in the Republic of Maldives, a South Asian country located in the Indian Ocean.

Install necessary packages

The following code will install the packages used in the project:

```
#list of packages used
packages <- c("tm", "wordcloud", "lubridate", "SnowballC", "dplyr", "tidyr", "ggplot2")

#check to see if package is already installed, if not, install
for(p in packages){
  if(!require(p, character.only = TRUE)) {
    install.packages(p)
    library(p, character.only = TRUE)
  }
}
```

Import the Hotel Review Dataset

Import the data from my github page

```
#import data from github
path <- "https://raw.githubusercontent.com/ianjeffries/review-sentiment-analysis/master/
data/maldives_hotel_reviews.csv"

reviews <- read.csv(path, header = TRUE, stringsAsFactors = FALSE)

#see structure of dataframe
str(reviews)
```

```
## 'data.frame':    21093 obs. of  8 variables:
##  $ Review.id         : chr  "rn629306037" "rn623195151" "rn627907077" "rn623362603" ...
##  $ Hotel.Name        : chr  "Holiday Inn Resort Kandooma Maldives" "Holiday Inn Resort Kandooma Maldi
##  $ Total.review.count: int  1679 1679 1679 1679 1679 1679 1679 1679 1679 1679 ...
##  $ Review.Date       : chr  "29-Oct-18" "8-Oct-18" "24-Oct-18" "8-Oct-18" ...
##  $ Review_viaMobile  : chr  "via mobile " "via mobile " "via mobile " NA ...
##  $ Guest.Location    : chr  "Waikiki, Hawaii" "Berkshire, United Kingdom" "Edinburgh, United Kingdom
##  $ Review.Heading    : chr  "Amazing!!!" "Stunning, wonderful and peaceful " "We don't want to leave
##  $ Review            : chr  "This is one of my most favorite resort that I have ever visited in Maldi
```

Cleaning the Data

The data can now be prepared for analysis.

```r
#change date to date format
reviews$Review.Date <- dmy(reviews$Review.Date)

#fix "invalid multibyte" error
for (i in c(1, 2, 5:8)) {
  reviews[ ,i] <- enc2utf8(reviews[,i])
}

#remove rows without review
reviews <- reviews[!reviews$Review.Heading=="", ]

#add in review number
reviews$review.number <- 1:21039
```

Text Cleanup

The header and review columns both hold key guest sentiments, and will both be used to mine for guest opinion. By putting each into their own vector, useless characters such as punctuation and digits can be removed.

```r
#create separate text vectors for headings and reviews
header_vec <- reviews$Review.Heading
review_vec <- reviews$Review

#remove blank spaces from beginning of reviews
header_vec <- gsub("^ ", "", header_vec)
review_vec <- gsub("^ ", "", review_vec)

#remove blank spaces from end of reviews
header_vec <- gsub(" $", "", header_vec)
review_vec <- gsub(" $", "", review_vec)

#convert to lower class
header_vec <- tolower(header_vec)
review_vec <- tolower(review_vec)

#remove links
header_vec <- gsub("http\\S+\\s*", "", header_vec)
review_vec <- gsub("http\\S+\\s*", "", review_vec)

#remove punctuation
header_vec <- gsub("[[:punct:]]", "", header_vec)
review_vec <- gsub("[[:punct:]]", "", review_vec)

#remove digits
header_vec <- gsub("[[:digit:]]", "", header_vec)
review_vec <- gsub("[[:digit:]]", "", review_vec)

#remove irrelevant words (no information gleaned from these terms)
header_vec <- gsub("maldives", "", header_vec)
header_vec <- gsub("hotel", "", header_vec)
header_vec <- gsub("resort", "", header_vec)
header_vec <- gsub("island", "", header_vec)
```

```
header_vec <- gsub("staff", "", header_vec)
header_vec <- gsub("stayed", "", header_vec)
review_vec <- gsub("maldives", "", review_vec)
review_vec <- gsub("hotel", "", review_vec)
review_vec <- gsub("resort", "", review_vec)
review_vec <- gsub("island", "", review_vec)
review_vec <- gsub("staff", "", review_vec)
review_vec <- gsub("stayed", "", review_vec)

#convert to corpus for text mining
header_vec <- Corpus(VectorSource(header_vec))
review_vec <- Corpus(VectorSource(review_vec))

#remove stop words and white space from header
header_vec <- tm_map(header_vec, removeWords,stopwords("english"))
header_vec <- tm_map(header_vec, stripWhitespace)

#remove stop words and white space from review body
review_vec <- tm_map(review_vec, removeWords,stopwords("english"))
review_vec <- tm_map(review_vec, stripWhitespace)
```

Prepare for Text Mining

Two text files, a positive and negative lexicon, will be used to identify "positive" and "negative" words. These can be loaded from my github page.

```
#load positive and negative lexicons
path1 <- "https://raw.githubusercontent.com/ianjeffries/review-sentiment-analysis/master/
data/negative-lexicon.txt"

path2 <- "https://raw.githubusercontent.com/ianjeffries/review-sentiment-analysis/master/
data/positive-lexicon.txt"

#read in lexicons
neg_lexicon <- read.csv(path1)
pos_lexicon <- read.csv(path2)

#create vector to store line-item results for sentiment analysis
header_results <- matrix(rep(0, 63117), ncol = 3,
                         dimnames = list(c(1:21039), c("review.number", "header.pos.count",
                                                       "header.neg.count")))
#convert to dataframe
header_results <- as.data.frame(header_results)

#create vector to store line-item results for sentiment analysis
review_results <- matrix(rep(0, 63117), ncol = 3,
                         dimnames = list(c(1:21039), c("review.number", "review.pos.count",
                                                       "review.neg.count")))
#convert to dataframe
review_results <- as.data.frame(review_results)
```

Generate Wordclouds

Wordclouds can be used to view the most frequently occuring words.

```
#generate header wordcloud
wordcloud(header_vec,
          min.freq = 3,
          colors=brewer.pal(8, "Dark2"),
          random.color = TRUE,
          random.order = FALSE,
          max.words = 75)
```


```
#generate review body wordcloud
wordcloud(review_vec,
          min.freq = 3,
          colors=brewer.pal(8, "Dark2"),
          random.color = TRUE,
          random.order = FALSE,
          max.words = 75)
```

Find Guest Sentiment

The wordclouds show that mainly positive words are the most commonly occuring, but we can dig in even further. The following loop compares each word in the review header and body to the positive and negative lexicons, and assigns a count for every positive or negative match. A negative, positive, or neutral label is then assigned to each review, based on the total percentage of positive or negative words.

```
#find sentiment in review header
for(i in 1:length(header_vec)) {
  #All the words in headers
  corpus_words <- list(strsplit(header_vec[[i]]$content, split = " "))
  #positive words in headers
  pos_count <- length(intersect(unlist(corpus_words), unlist(pos_lexicon)))
  #negative words in current review
  neg_count <- length(intersect(unlist(corpus_words), unlist(neg_lexicon)))
  header_results[i, 1] <- i
  header_results[i, 2] <- pos_count ## track positive count
  header_results[i, 3] <- neg_count ## track negative count
}

#find sentiment in review body
for(i in 1:length(review_vec)) {
  #All the words in headers
  corpus_words <- list(strsplit(review_vec[[i]]$content, split = " "))
  #positive words in headers
```

```
  pos_count <- length(intersect(unlist(corpus_words), unlist(pos_lexicon)))
  #negative words in current review
  neg_count <- length(intersect(unlist(corpus_words), unlist(neg_lexicon)))
  review_results[i, 1] <- i
  review_results[i, 2] <- pos_count ## track positive count
  review_results[i, 3] <- neg_count ## track negative count
}

#add header sentiment back with dataset
final_sentiment <- left_join(reviews, select(header_results, review.number,
                                             header.pos.count, header.neg.count),
                             by = c("review.number"))

#add body sentiment back with dataset
final_sentiment <- left_join(final_sentiment, select(review_results, review.number,
                                                     review.pos.count, review.neg.count),
                             by = c("review.number"))

#drop irrelevant columns
final_sentiment <- final_sentiment[, -which(names(final_sentiment) %in%
                                              c("Total.review.count",
                                                "Review_viaMobile",
                                                "Review.Heading", "Review"))]

#rearrange
final_sentiment <- final_sentiment[,c(5,1,2,3,4,6,7,8,9)]

#view header positive sentiment percentage
sum(final_sentiment$header.pos.count) / (sum(final_sentiment$header.pos.count)+
                                           sum(final_sentiment$header.neg.count))
```

```
## [1] 0.941919
```

```
#view header negative sentiment percentage
sum(final_sentiment$header.neg.count) / (sum(final_sentiment$header.pos.count)+
                                           sum(final_sentiment$header.neg.count))
```

```
## [1] 0.05808105
```

```
#view body positive sentiment percentage
sum(final_sentiment$review.pos.count) / (sum(final_sentiment$review.pos.count)+
                                           sum(final_sentiment$review.neg.count))
```

```
## [1] 0.8968929
```

```
#view body negative sentiment percentage
sum(final_sentiment$review.neg.count) / (sum(final_sentiment$review.pos.count)+
                                           sum(final_sentiment$review.neg.count))
```

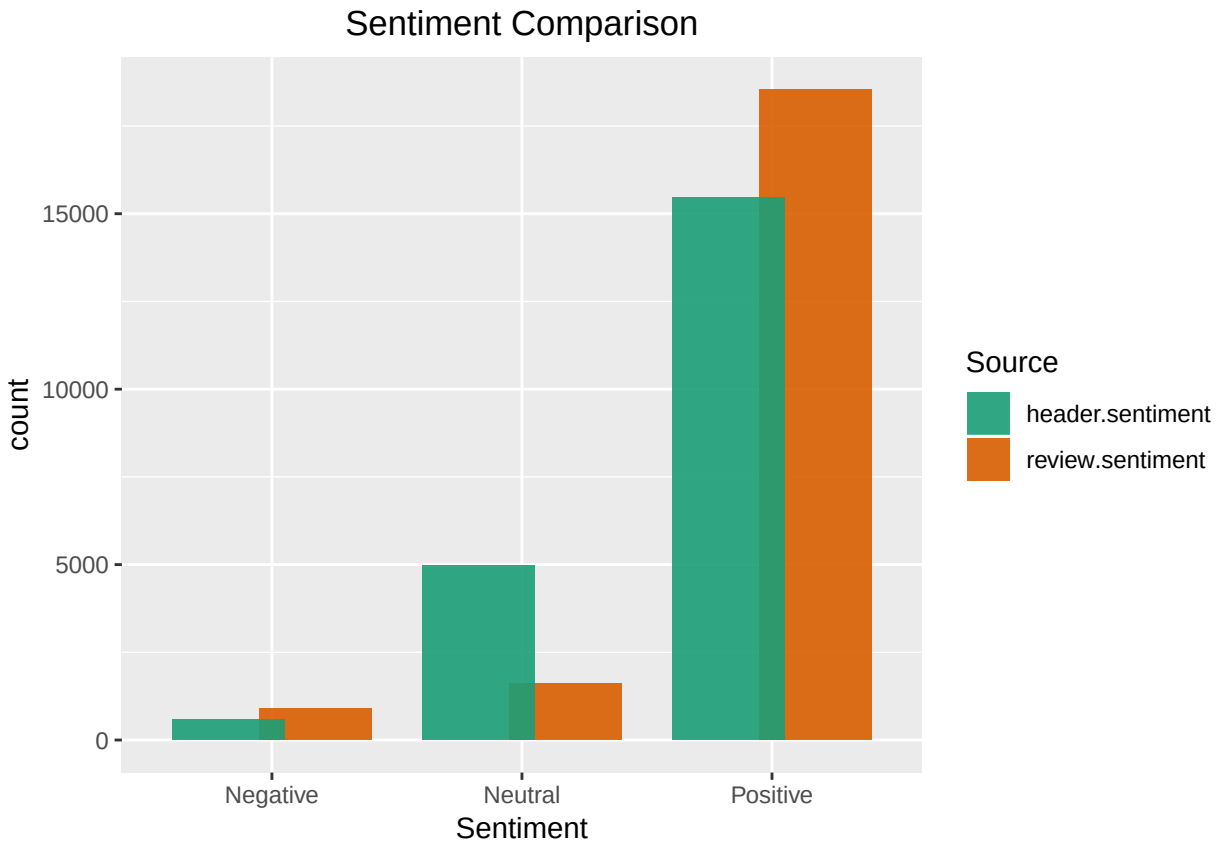
```
## [1] 0.1031071
```

```r
#calculate overall sentiment percentage by row
final_sentiment$header.pos.percent <- round(final_sentiment$header.pos.count /
                                            (final_sentiment$header.pos.count +
                                             final_sentiment$header.neg.count) * 100, 2)

final_sentiment$header.neg.percent <- round(final_sentiment$header.neg.count /
                                            (final_sentiment$header.pos.count +

final_sentiment$review.pos.percent <- round(final_sentiment$review.pos.count /
                                            (final_sentiment$review.pos.count +

final_sentiment$review.neg.percent <- round(final_sentiment$review.neg.count /
                                            (final_sentiment$review.pos.count +

#change NaN to 50-50 (both zero)
final_sentiment$header.pos.percent <- gsub("NaN", 50, final_sentiment$header.pos.percent)
final_sentiment$header.neg.percent <- gsub("NaN", 50, final_sentiment$header.neg.percent)
final_sentiment$review.pos.percent <- gsub("NaN", 50, final_sentiment$review.pos.percent)
final_sentiment$review.neg.percent <- gsub("NaN", 50, final_sentiment$review.neg.percent)

#change percentage to numeric
for (i in 10:13) {
  final_sentiment[,i] <- as.numeric(final_sentiment[,i])
}

#add in overall sentiment label
final_sentiment$header.sentiment <- ""
final_sentiment$review.sentiment <- ""

#calculate sentiment labels for review headers
for (i in 1:nrow(final_sentiment)) {
  final_sentiment[i, which(names(final_sentiment) == "header.sentiment")]  <-
    if(final_sentiment[i, which(names(final_sentiment) == "header.pos.percent")] > 50) {
      "Positive"
    } else if (final_sentiment[i, which(names(final_sentiment) ==
                                        "header.pos.percent")] == 50) {
      "Neutral"
    } else {
      "Negative"
    }
}

#calculate sentiment labels for review body
for (i in 1:nrow(final_sentiment)) {
  final_sentiment[i, which(names(final_sentiment) == "review.sentiment")]  <-
    if(final_sentiment[i, which(names(final_sentiment) == "review.pos.percent")] > 50) {
      "Positive"
    } else if (final_sentiment[i, which(names(final_sentiment) == "review.pos.percent")] == 50) {
      "Neutral"
    } else {
      "Negative"
    }
}
```

Visualize First Pass

A graph can now be created to vizualize the overall sentiment of the reviews.

```
#rearrange dataset
final_sentiment <- final_sentiment[ ,c(1:7, 10, 11, 14, 8, 9, 12, 13, 15)]

#create matirx to graph sentiment results
sent_comparison <- matrix(rep(0, 42078), ncol = 2,
                          dimnames = list(c(1:21039), c("header.sentiment",
                                                        "review.sentiment")))

#convert matrix to dataframe
sent_comparison <- as.data.frame(sent_comparison)

#add in results
sent_comparison[ , 1:2] <- final_sentiment[ ,c(10,15)]

#make dataframe narrow to graph
sent_comparison <- gather(sent_comparison, key = Source, value = Sentiment, 1:2)

#plot to see sentiment breakdown between review header and body
theme_update(plot.title = element_text(hjust = 0.5))
ggplot(sent_comparison, aes(x = Sentiment, fill = Source)) +
  geom_histogram(stat = "count", position = position_dodge(width = .7), alpha = .9) +
  ggtitle("Sentiment Comparison") +
  scale_fill_brewer(palette = "Dark2")
```



Combine Header and Body Sentiment

The total positive and negative word count for the review body and header will be combined, in an attempt to glean even more information on guest sentiment.

```
#combine review and header sentiment counts for greater clarity
final_sentiment2 <- final_sentiment[ , 1:5]
final_sentiment2$pos.count <- final_sentiment$header.pos.count +
                              final_sentiment$review.pos.count

final_sentiment2$neg.count <- final_sentiment$header.neg.count +
                              final_sentiment$review.neg.count

#create sentiment percetage
final_sentiment2$pos.percent <- round(final_sentiment2$pos.count /
                                      (final_sentiment2$pos.count +

final_sentiment2$pos.percent <- gsub("NaN", 50, final_sentiment2$pos.percent)

final_sentiment2$neg.percent <- round(final_sentiment2$neg.count /
                                      (final_sentiment2$pos.count +
                                       final_sentiment2$neg.count) * 100, 2)

final_sentiment2$neg.percent <- gsub("NaN", 50, final_sentiment2$neg.percent)
```

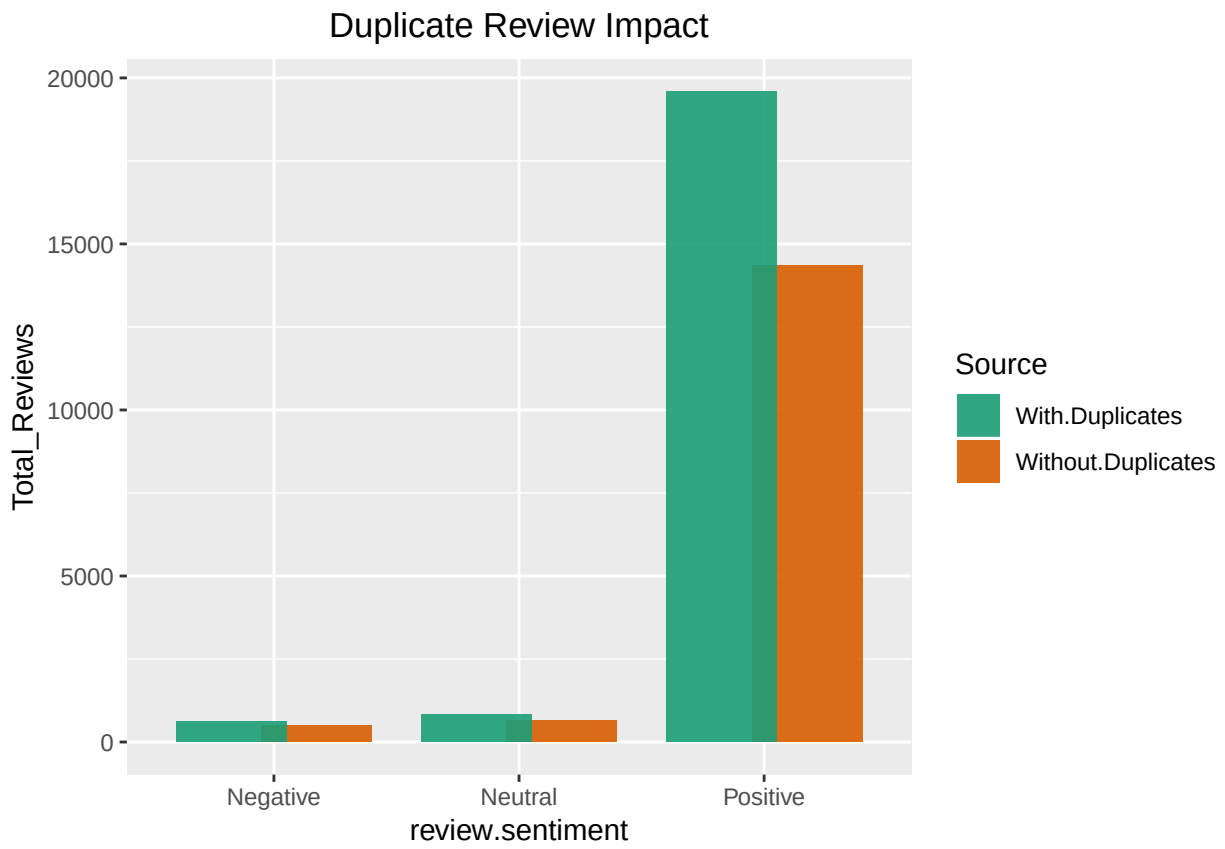
```r
#change percentage to numeric
for (i in 8:9) {
  final_sentiment2[,i] <- as.numeric(final_sentiment2[,i])
}

#add in sentiment label
final_sentiment2$review.sentiment <- ""

for (i in 1:nrow(final_sentiment2)) {
  final_sentiment2[i, which(names(final_sentiment2) == "review.sentiment")]  <-
    if(final_sentiment2[i, which(names(final_sentiment2) == "pos.percent")] > 50) {
      "Positive"
    } else if (final_sentiment2[i, which(names(final_sentiment2) == "pos.percent")] == 50) {
      "Neutral"
    } else {
      "Negative"
    }
}
```

Remove Duplicate Reviews

During the visualization phase, I noticed that there were duplicate reviews present (possibly created by bots to inflate positive reviews). The following visualizes the impact of duplicate reviews.

```r
#create dataset for time series analysis (this is where I found the duplicates)
time_plot <- final_sentiment2
time_plot$Review.Year <- final_sentiment2$Review.Date
time_plot$Review.Year <- as.POSIXct(time_plot$Review.Year)
time_plot$Review.Year <- format(time_plot$Review.Year, "%Y")
time_plot$Review.Date <- as.POSIXct(time_plot$Review.Date)
time_plot$Review.Date <- format(time_plot$Review.Date, "%Y-%m")

#create matrix to look at duplicates
dup_removal <- matrix(rep(0,9), ncol = 3, dimnames = list(1:3, c("review.sentiment",
                                                                 "With.Duplicates",
                                                                 "Without.Duplicates")))

#convert to dataframe
dup_removal <- as.data.frame(dup_removal)

#look at effect of removing duplicates
dup_removal[,c(1,3)] <- time_plot %>%
  select(Review.id, pos.count, neg.count, review.sentiment) %>%
  mutate(review.count = pos.count * 0 +1) %>%
  gather(key = Review.Count, value = count, c(2,3,5)) %>%
  group_by(Review.id, review.sentiment, Review.Count) %>%
  summarize(count = sum(count)) %>%
  spread(Review.Count, count) %>%
  mutate(positive.sentiment.percent = round(sum(pos.count) /
                                              (sum(neg.count) + sum(pos.count)) * 100, 2),
         negative.sentiment.percent = round(sum(neg.count) /
                                              (sum(neg.count) + sum(pos.count)) * 100, 2)) %>%
  arrange(desc(review.count)) %>%
```

```
  filter(review.count < 8) %>%
  group_by(review.sentiment) %>%
  summarize(review.count = sum(review.count))

#bring in with duplicates to compare
dup_removal[,c(1,2)] <- time_plot %>%
  select(Review.id, pos.count, neg.count, review.sentiment) %>%
  mutate(review.count = pos.count * 0 +1) %>%
  gather(key = Review.Count, value = count, c(2,3,5)) %>%
  group_by(Review.id, review.sentiment, Review.Count) %>%
  summarize(count = sum(count)) %>%
  spread(Review.Count, count) %>%
  mutate(positive.sentiment.percent = round(sum(pos.count) /
                                              (sum(neg.count) + sum(pos.count)) * 100, 2),
         negative.sentiment.percent = round(sum(neg.count) /
                                              (sum(neg.count) + sum(pos.count)) * 100, 2)) %>%
  group_by(review.sentiment) %>%
  summarize(review.count = sum(review.count))

#make dataset narrow for graphing
dup_removal <- gather(dup_removal, key = Source, value = Total_Reviews, 2:3)

#graph to compare sentiment with and without duplicates
ggplot(dup_removal, aes(x = review.sentiment, y = Total_Reviews, fill = Source)) +
  geom_bar(stat = "identity", position = position_dodge(width = .7), alpha = .9) +
  ggtitle("Duplicate Review Impact") +
  scale_fill_brewer(palette = "Dark2")
```

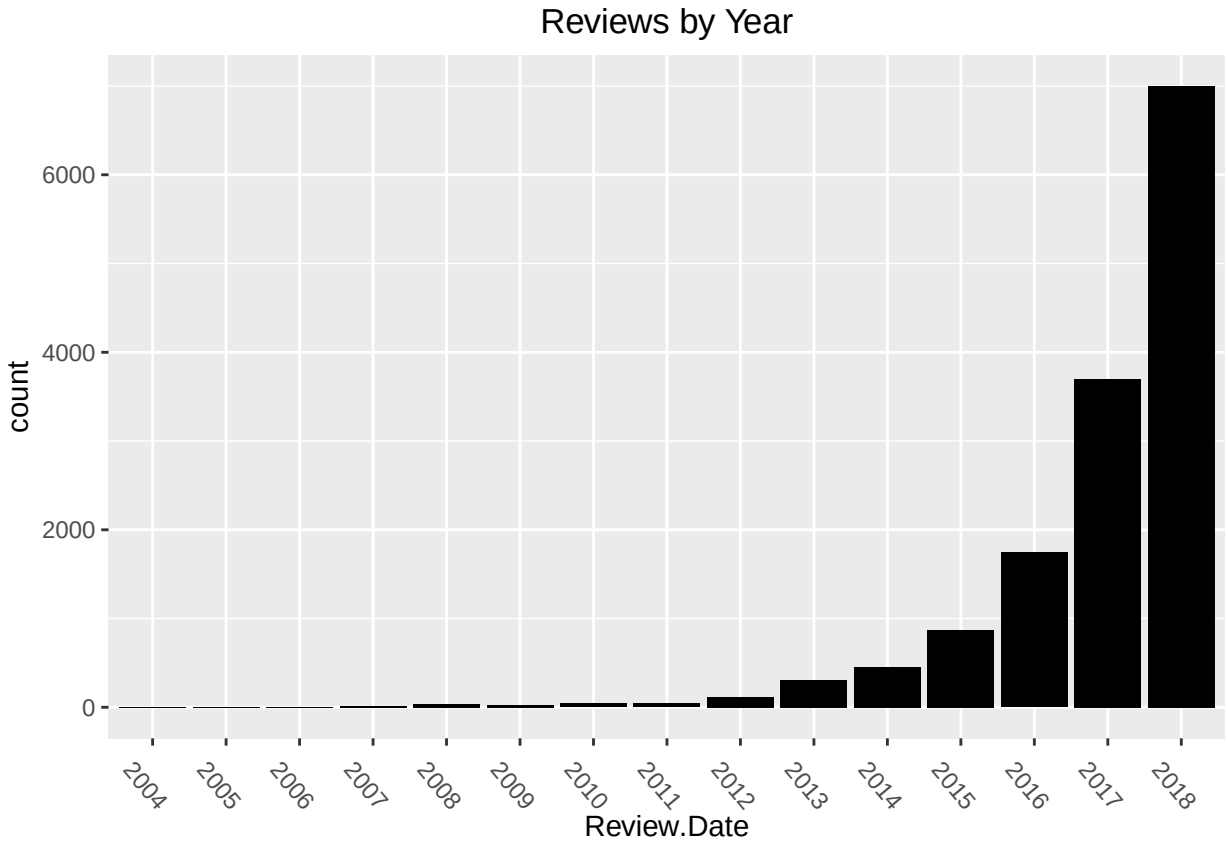



```
#remove duplicates from time_plot
deduped_final_data <- unique(final_sentiment2[ , c(2:10)])
```

Visualize Final Results

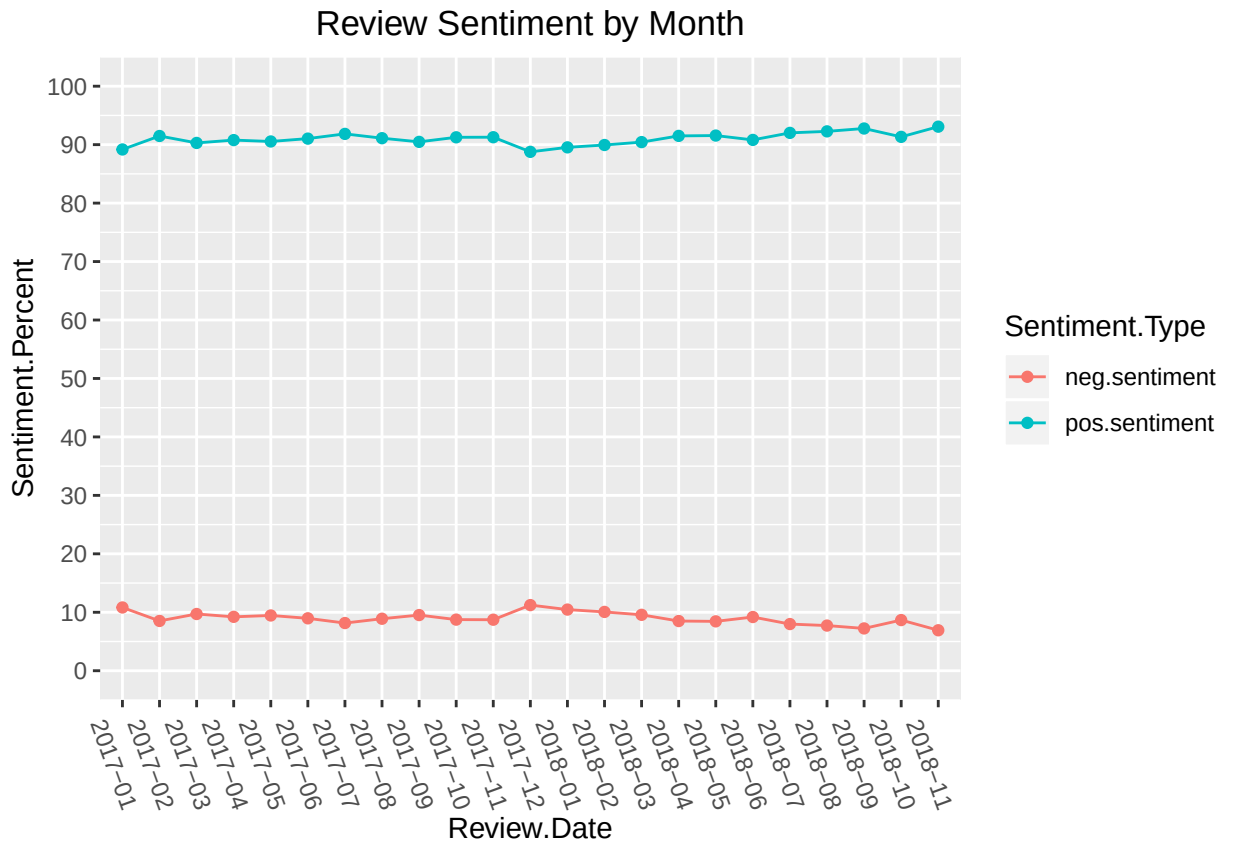
Finally, graphs can be created to see the number of reviews by year, the total review sentiment by month for all resorts in Maldives, and the sentiment of the top 12 hotels, defined by the total word count of reviews for that specific resort. (more words means more text to work with for analysis)

```
#plot reviews by date
year_plot <- deduped_final_data
year_plot$Review.Date <- as.POSIXct(year_plot$Review.Date)
year_plot$Review.Date <- format(year_plot$Review.Date, "%Y")
ggplot(year_plot, aes(x = Review.Date)) +
  geom_histogram(stat = "count", fill = "black") +
  ggtitle("Reviews by Year") +
  theme(axis.text.x = element_text(angle = -50))
```



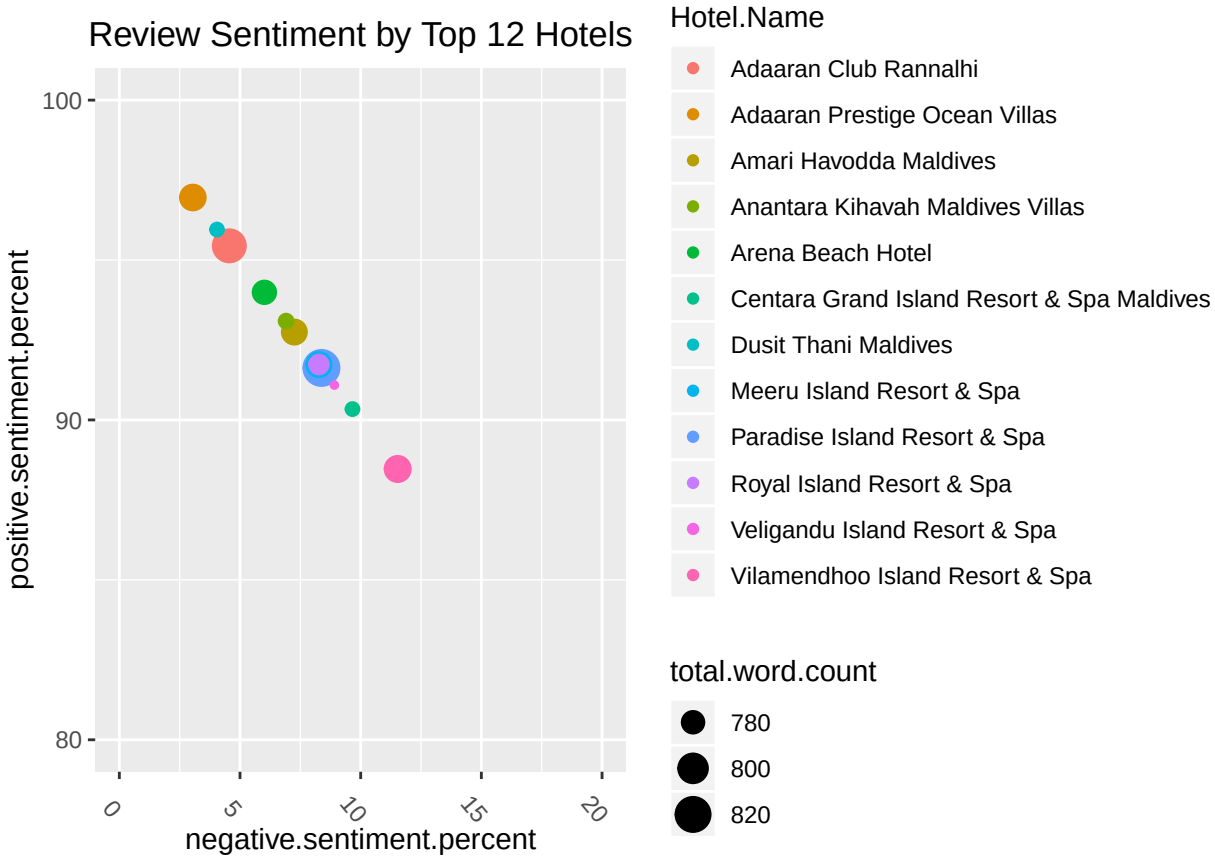

```
#recreate dataset for timeplot analysis
time_plot <- deduped_final_data
time_plot$Review.Year <- deduped_final_data$Review.Date
time_plot$Review.Year <- as.POSIXct(time_plot$Review.Year)
time_plot$Review.Year <- format(time_plot$Review.Year, "%Y")
time_plot$Review.Date <- as.POSIXct(time_plot$Review.Date)
time_plot$Review.Date <- format(time_plot$Review.Date, "%Y-%m")

#graph by month for 2017 and 2018
time_plot %>%
  select(Review.Date, pos.count, neg.count, Review.Year) %>%
  filter(Review.Year == 2018 | Review.Year == 2017) %>%
  gather(key = Review.Count, value = count, 2:3) %>%
  group_by(Review.Date, Review.Count) %>%
  summarize(count = sum(count)) %>%
  spread(Review.Count, count) %>%
  mutate(pos.sentiment = as.numeric(sum(pos.count) / (sum(neg.count) +
                                                        sum(pos.count)) * 100),
         neg.sentiment = as.numeric(sum(neg.count) / (sum(neg.count) +
                                                        sum(pos.count)) * 100)) %>%
  select(Review.Date, pos.sentiment, neg.sentiment) %>%
  gather(key = Sentiment.Type, value = Sentiment.Percent, 2:3) %>%
  ggplot(aes(x = Review.Date, y = Sentiment.Percent,
             color = Sentiment.Type, group = Sentiment.Type)) +
  geom_point() + geom_line() +
  ggtitle("Review Sentiment by Month") +
```

```
  theme(axis.text.x = element_text(angle = -70)) +
  scale_y_continuous(limits = c(0,100), breaks = c(0, 10, 20, 30, 40, 50,
                                                   60, 70, 80, 90, 100))
```




```
#graph by top 12 hotels
time_plot %>%
  select(Hotel.Name, pos.count, neg.count) %>%
  gather(key = Review.Count, value = count, 2:3) %>%
  group_by(Hotel.Name, Review.Count) %>%
  summarize(count = sum(count)) %>%
  spread(Review.Count, count) %>%
  mutate(total.word.count = pos.count + neg.count) %>%
  mutate(positive.sentiment.percent = as.numeric(sum(pos.count) / (sum(neg.count) +
                                                   sum(pos.count)) * 100),
         negative.sentiment.percent = as.numeric(sum(neg.count) / (sum(neg.count) +
                                                   sum(pos.count)) * 100)) %>%
  arrange(desc(total.word.count)) %>%
  filter(total.word.count > 762) %>%
  select(Hotel.Name, positive.sentiment.percent,
         negative.sentiment.percent, total.word.count) %>%
  ggplot(aes(x = negative.sentiment.percent, y = positive.sentiment.percent,
             color = Hotel.Name, size = total.word.count)) +
  geom_point() +
  ggtitle("Review Sentiment by Top 12 Hotels") +
  theme(axis.text.x = element_text(angle = -50)) +
```

```
  scale_y_continuous(limits = c(80,100), breaks = c(80, 90, 100)) +
  scale_x_continuous(limits = c(0,20), breaks = c(0,5,10,15,20))
```



Conclusion

In conclusion, it appears that resorts in the Maldives have very positive sentiment from the guests that have stayed there. A full write up of methods used and analysis can be found on my github page.