DATA607 Fall 2018 Final Project

John K. Hancock

November 12, 2018

DATA 607 - Fall 2018 - Final Project: NYTimes Movie Reviews Sentiment Analysis

I. Introduction

This project explores the sentiment analysis of NY Times movie reviews to access whether it's a predictor of the box office performance of the film. The question that this project answers is: Does the sentiment of a NY Time movie review have an effect on a film's box office performance.

Re-written as Hypothesis test, it would be:

H0: The sentiment of a NY Times movie review has no effect on box office performance. HA: The sentiment of a NY Time movie review has an effect on box office performance.

II. Methodology

A. Data was collected from two sources, the NY Times API and the website "the-numbers.com" which tracks box office statistics for all motion picture releases in the United States.

B. The data from these two sources were joined by a movie's title creating a single data frame.

C. Explore the new, combined data set.

D. Prepare the data for Analysis

E. Perform tidytext sentiment analysis using the NRC, Bing, and AFINN lexicon

F. Perform regression analysis between the sentiment and box office performance

G. Conclusion, Reflections, and Sources

III. Data Collection and Import the Data

A. Webscrape the-numbers.com website.

Launched by Bruce Nash in 1997, the-numbers.com website tracks movie financial data. Below, I scraped data for all 2018 domestic movie releases.

```
url <- 'https://www.the-numbers.com/market/2018/top-grossing-movies'

movies<- url  %>%
  read_html() %>%
  html_nodes(xpath = '//*[@id="page_filling_chart"]/table') %>%
  html_table(fill = TRUE)

movies <- movies[[1]]


movies_df<- as.data.frame(movies)
```

B. Access the NYTimes API.

The following sections are taken from my Week 9 NYTimes API assignment, “Working with the NY Times API”.

Note, some of the code in this section has been commented out. When I submitted the project proposal, I was able to access the NYTimes API and download the critics data. However, on subsequent runs of this code, it was unable to make a connection to the NY Times API. I may have exceeded some sort of limit on accessing this API and downloading the data. Fortunately, at the time of my submission of the project proposal, I saved off the downloads from both the-numbers.com and NY Times webscrape and API into csv files and preserved them locally.Data Saved off into csv files:

–DO NOT RUN–

```
# NYTimes_KEY <- "3f11a8ef5cf94222beda574c715815e8"
# baseURL = paste0("https://api.nytimes.com/svc/movies/v2/reviews/search.json?api-key=", NYTimes
_KEY, sep="")
```

–DO NOT RUN–

```
#Create an empty list called pages
# pages <- list()
# #create a for loop which makes 1000 calls to the API
# for(i in seq(0,1000,2)){
#   NY_TIMES_CRITICS <- fromJSON(paste0(baseURL, "&offset=", i))
#   #Add the results of the calls to the pages list
#   pages[[i+1]] <- NY_TIMES_CRITICS$results

#}
```

–DO NOT RUN–

```
# NY_TIMES_CRITICS_df <- rbind_pages(pages)
# dim(NY_TIMES_CRITICS_df)
```

–DO NOT RUN–

```
# NY_TIMES_CRITICS_df <- NY_TIMES_CRITICS_df[!duplicated(NY_TIMES_CRITICS_df$link$url), ]
# NY_TIMES_CRITICS_df$link
```

–DO NOT RUN–

```
# Body <- list()
# X_path = '//*[contains(concat( " ", @class, " " ), concat( " ", "css-1572rug", " " ))]'
# for (i in seq(1,1017)){
#     getContent <- read_html(NY_TIMES_CRITICS_df$link$url[i])
#     articles <- html_nodes(getContent, xpath = X_path )
#     Body[i] <- list(html_text(articles))
#
#
#
# }
```

–DO NOT RUN–

```
# for (i in seq(1, 1017)){
#     NY_TIMES_CRITICS_df$Body[[i]] <- Body[[i]]
# }
#
```

C. Import the Data from csv Files

As stated earlier, I ran the data collection code earlier and preserved the data to two csv files for future reference. In the code below, I loaded the data from the two csv files.

```
Box_Office_df <- read.csv('BoxOffice2018.csv')

glimpse(Box_Office_df)
```

```
## Observations: 526
## Variables: 9
## $ X            <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15...
## $ Rank         <fct> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15...
## $ Movie        <fct> Black Panther, Avengers: Infinity War, Incredible...
## $ ReleaseDate  <fct> 2/16/2018, 4/27/2018, 6/15/2018, 6/22/2018, 5/18/...
## $ Distributor  <fct> Walt Disney, Walt Disney, Walt Disney, Universal,...
## $ Genre        <fct> Action, Action, Adventure, Action, Action, Advent...
## $ MPAA         <fct> PG-13, PG-13, PG, PG-13, R, PG-13, PG-13, PG-13, ...
## $ X2018.Gross  <fct> $700,059,566, $678,815,482, $608,517,638, $416,76...
## $ Tickets.Sold <fct> 78,044,544, 75,676,196, 67,839,201, 46,462,580, 3...
```

```
NYTIMES_Reviews_df <- read.csv('NYTimes.csv')
glimpse(NYTIMES_Reviews_df)
```

```
## Observations: 1,040
## Variables: 18
## $ X                        <fct> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12...
## $ display_title            <fct> Infinite Football, Number 37, Weightl...
## $ mpaa_rating              <fct> , , R, R, PG-13, R, R, , R, , , R, , ...
## $ critics_pick             <int> 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ byline                   <fct> GLENN KENNY, JEANNETTE CATSOULIS, KEN...
## $ headline                 <fct> □Infinite Football□ Review: Soccer an...
## $ summary_short            <fct> By its end, the Romanian director Cor...
## $ publication_date         <fct> 2018-11-08, 2018-11-08, 2018-11-08, 2...
## $ opening_date             <fct> 2018-11-09, 2018-11-09, 2018-11-09, 2...
## $ date_updated             <fct> 2018-11-08 12:04:10, 2018-11-08 12:04...
## $ link.type                <fct> article, article, article, article, a...
## $ link.url                 <fct> http://www.nytimes.com/2018/11/08/mov...
## $ link.suggested_link_text <fct> Read the New York Times Review of Inf...
## $ multimedia.type          <fct> mediumThreeByTwo210, mediumThreeByTwo...
## $ multimedia.src           <fct> https://static01.nyt.com/images/2018/...
## $ multimedia.width         <fct> 210, 210, 210, 210, 210, 210, 210, 21...
## $ multimedia.height        <fct> 140, 140, 140, 140, 140, 140, 140, 14...
## $ Body                     <fct> The surfaces of this documentary are,...
```

IV. Transform the Data

The goal of this section is to transform the two datasets and then combine the two datasets into one dataset, Movies. The Movies dataset will consist of the NY Time critics and reviews along with box office data from numbers-com.

First, I create a new dataset called NYTimes_Review by selecting and renaming three columns. Additionally, I changed two of the columns to characters, leaving the critic variable as a factor.

```
#Create a new tbl by selecting three columns and rename the columns
NYTimes_Review <- NYTIMES_Reviews_df %>%
                  select(display_title,  byline, Body) %>%
                  rename(Title = display_title, Critic=byline, Review=Body )

#convert the Title and Review variables as character
NYTimes_Review$Title <- as.character(NYTimes_Review$Title)
NYTimes_Review$Review <- as.character(NYTimes_Review$Review)


glimpse(NYTimes_Review)
```

```
## Observations: 1,040
## Variables: 3
## $ Title  <chr> "Infinite Football", "Number 37", "Weightless", "The Ba...
## $ Critic <fct> GLENN KENNY, JEANNETTE CATSOULIS, KEN JAWOROWSKI, A.O. ...
## $ Review <chr> "The surfaces of this documentary are, as is customary ...
```

Data from the-numbers.com website:

```
#Convert the date format to year, month, day
Box_Office_df$ReleaseDate <- as.Date(Box_Office_df$ReleaseDate, format = "%m/%d/%Y")

#Select and rename the columns scraped from the website, the-numbers.com for clearer names and l
imit the data to movies released in 2018.
Box_Office <- Box_Office_df %>%
              select(Rank, Movie, Distributor, ReleaseDate, Genre, MPAA, X2018.Gross, Tickets.So
ld) %>%
              rename(Title=Movie, Gross = X2018.Gross, Tickets = Tickets.Sold, Movie_Studio=Dist
ributor) %>%
              filter(ReleaseDate > '2018-01-01')
```

```
## Warning: package 'bindrcpp' was built under R version 3.5.1
```

```
Box_Office$Title <- as.character(Box_Office$Title)
Box_Office$Gross <- as.numeric(gsub('\\$|,', '', Box_Office$Gross))
Box_Office$Tickets<- as.numeric(gsub('\\$|,', '', Box_Office$Tickets))
Box_Office$Rank <- as.numeric(as.character(Box_Office$Rank))

glimpse(Box_Office)
```

```
## Observations: 420
## Variables: 8
## $ Rank         <dbl> 1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1...
## $ Title        <chr> "Black Panther", "Avengers: Infinity War", "Incre...
## $ Movie_Studio <fct> Walt Disney, Walt Disney, Walt Disney, Universal,...
## $ ReleaseDate  <date> 2018-02-16, 2018-04-27, 2018-06-15, 2018-06-22, ...
## $ Genre        <fct> Action, Action, Adventure, Action, Action, Action...
## $ MPAA         <fct> PG-13, PG-13, PG, PG-13, R, PG-13, PG-13, PG-13, ...
## $ Gross        <dbl> 700059566, 678815482, 608517638, 416769345, 31849...
## $ Tickets      <dbl> 78044544, 75676196, 67839201, 46462580, 35506290,...
```

The final step is to join the two datasets by the Title of the movie, and remove any entries where there is no critic or review

```
Movies <- left_join(Box_Office, NYTimes_Review, by= 'Title')
Movies <- Movies[!is.na(Movies$Critic) | !is.na(Movies$Review),]
```

Next, I checked the spelling names of the critics, and I see that A.O. Scott has two different spelling variations, “A. O. SCOTT” and “A.O. SCOTT”. It’s a subtle difference but it will become important for Exploratory Data Analysis

```
unique(Movies$Critic)
```

```
##  [1] MANOHLA DARGIS      A.O. SCOTT          GLENN KENNY
##  [4] JEANNETTE CATSOULIS A. O. SCOTT         JASON ZINOMAN
##  [7] BEN KENIGSBERG      AISHA HARRIS        TEO BUGBEE
## [10] KEN JAWOROWSKI      BILGE EBIRI         WESLEY MORRIS
## [13] HELEN T. VERONGOS   JENNIFER SZALAI
## 22 Levels:  A. O. SCOTT A.O. SCOTT AISHA HARRIS ... WESLEY MORRIS
```

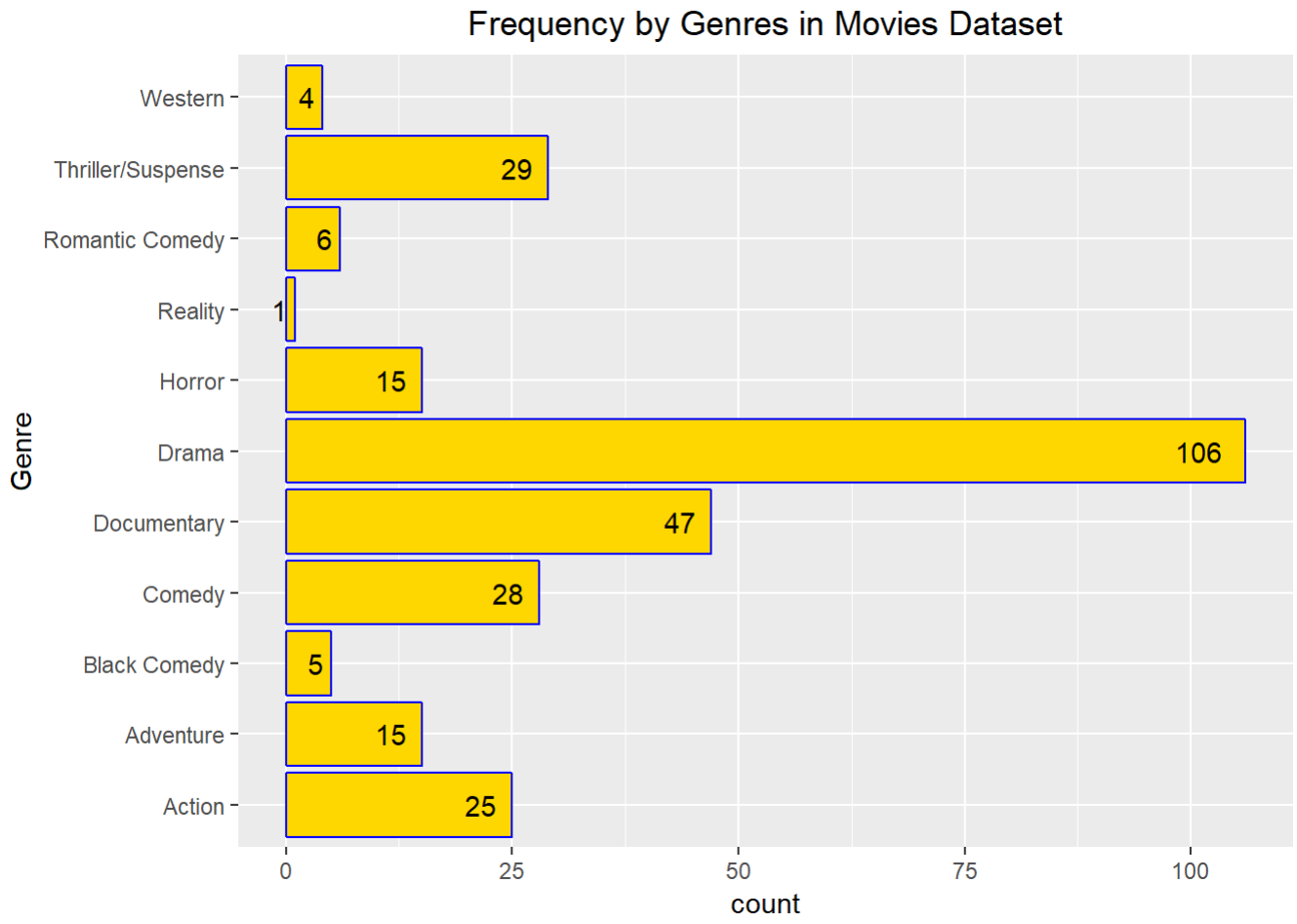
```
Movies<- Movies[!is.na(Movies$Review),]
Movies$Critic <- (gsub('A.O. SCOTT', 'A. O. SCOTT', Movies$Critic))
```

V. Exploratory Data Analysis

Frequency of Genres

From the frequency chart below, we can see that Dramas are far and away the most prevelant of the movie genres, more than double than it closest genre, Documentary.

```
ggplot(data=Movies, aes(x=Genre)) +
    geom_bar(stat="count", color="blue", fill="gold") +
    geom_text(stat='count', aes(label=..count..), hjust=1.5) +
    labs(title="Frequency by Genres in Movies Dataset") +
    theme(plot.title = element_text(hjust = 0.5)) +
    coord_flip()
```




```
dev.copy(png,'genre_freq.png')
```

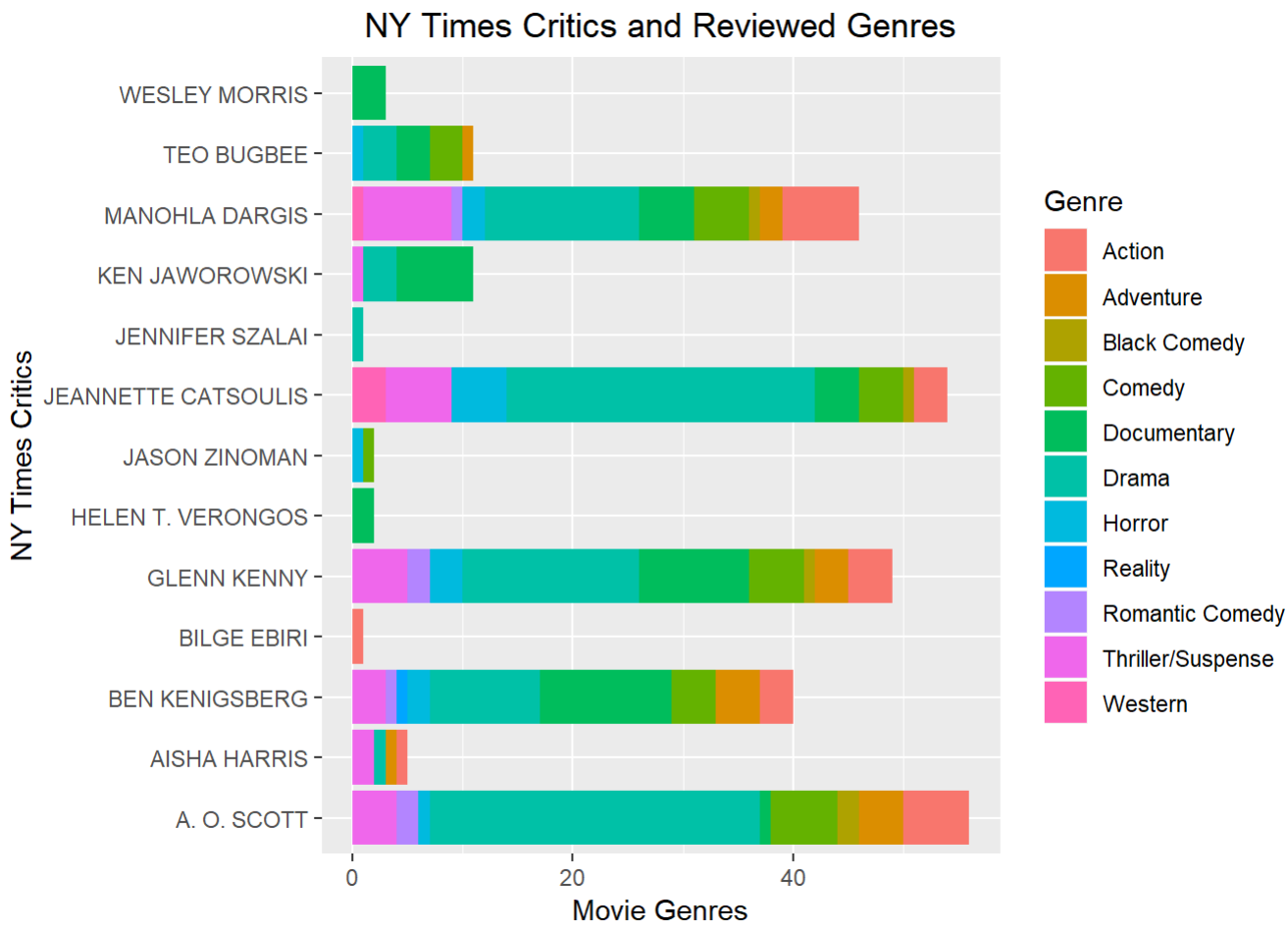
```
## png
##   3
```

```
dev.off()
```

```
## png
##   2
```

For the NY Times critics, A.O.SCOTT has the most reviews in our dataset followed by JEANNETTE CATSOULIS and MANOHLA DARGIS. These three critics along with GLENN KENNY and BEN KENIGSBERG review movies across all genres whereas the remaining critics reviewed a limited range of the genres.

```
critics <- Movies %>%
           select(Critic, Genre) %>%
           arrange(Critic)

ggplot(data=critics, aes(x = Critic, fill=Genre)) +
    geom_bar(stat="count") +
    labs(x="NY Times Critics", y="Movie Genres", title="NY Times Critics and Reviewed Genres") +
    theme(plot.title = element_text(hjust = 0.5)) +
    coord_flip()
```




```
dev.copy(png,'critics_freq.png')
```

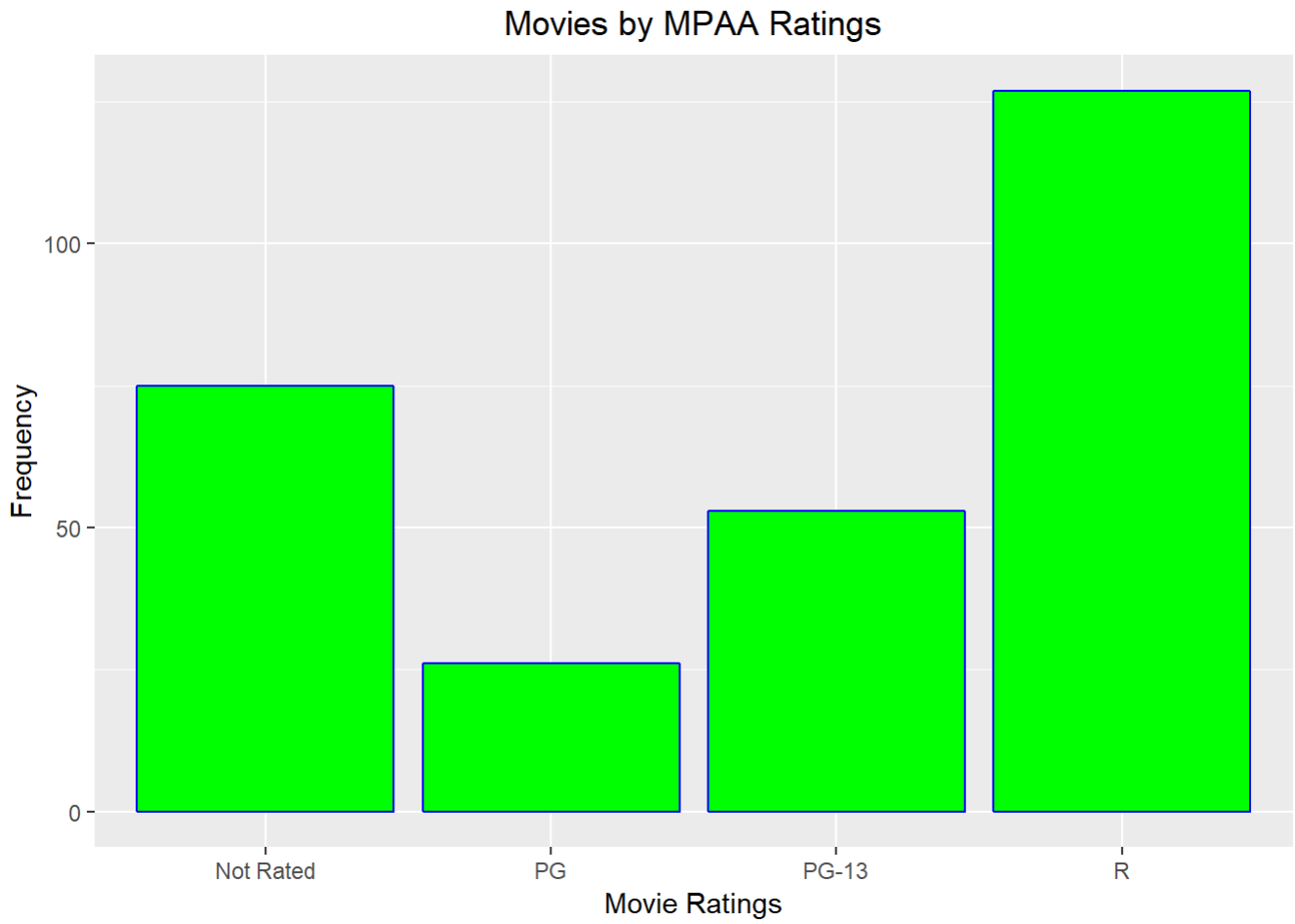
```
## png
##   3
```

```
dev.off()
```

```
## png
##   2
```

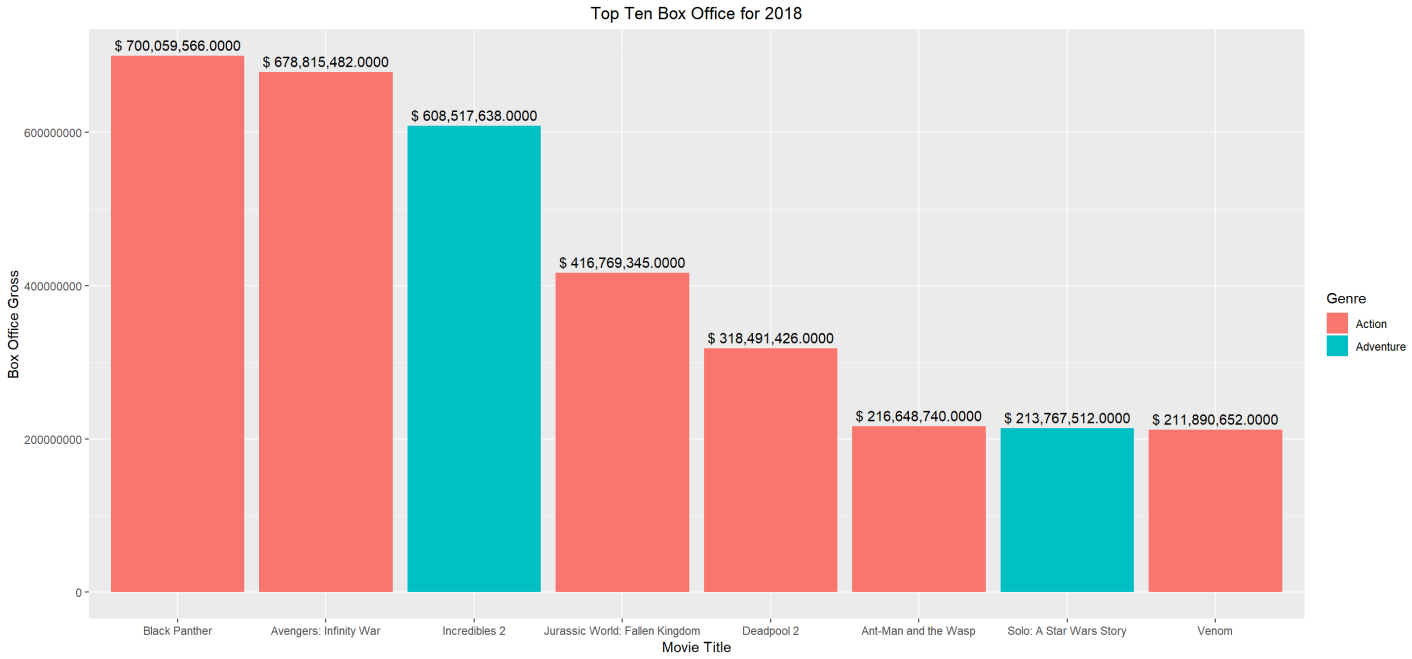
The majority of the movies in this dataset are rated “R” by the MPAA

```
options(scipen=5)

ggplot(data=Movies, aes(x=MPAA)) +
    geom_bar(stat="count", color="blue", fill="green") +
    theme(plot.title = element_text(hjust = 0.5)) +
    labs(x="Movie Ratings", y="Frequency", title="Movies by MPAA Ratings")
```



Finally, looking at the top ten highest box office gross movies, we see immediately that the top ten are dominated by Action or Adventure films. The highest grossing movie for 2018 is “The Black Panther” at $700 million.

```
topten <- Movies[Movies$Rank <=10,] %>%
          arrange(Gross)

options(scipen=5)

ggplot(data=topten, aes(x=reorder(Title, -Gross), y=Gross, fill=Genre )) +
     geom_bar(stat="identity", position=position_dodge()) +
     theme(plot.title = element_text(hjust = 0.5)) +
     geom_text(aes(label=paste('$',formatC(Gross, big.mark=',', format = 'f'))), position=positi
on_dodge(width=1), vjust=-0.5) +
     labs(x="Movie Title", y="Box Office Gross", title="Top Ten Box Office for 2018")
```



VI. Prepare Data for Analysis

One of the resources that I found most helpful was Data Camps' NLP tutorial by Debbie Liske, and their online course "Sentiment Analysis in R".

Data Camp Sentiment Analysis in R (https://www.datacamp.com/courses/sentiment-analysis-in-r-the-tidy-way)

Machine Learning and NLP using R: Topic Modeling and Music Classification by Debbie Liske (https://www.datacamp.com/community/tutorials/ML-NLP-lyric-analysis)

Code from NLP Tutorial (https://www.datacamp.com/community/tutorials/R-nlp-machine-learning)

Removing contractions and converting text to lower case

```
#This function reverses contractions into full words
remove.contractions <- function(doc) {
    doc <- gsub("won't", "will not", doc)
  doc <- gsub("can't", "can not", doc)
  doc <- gsub("n't", " not", doc)
  doc <- gsub("'ll", " will", doc)
  doc <- gsub("'re", " are", doc)
  doc <- gsub("'ve", " have", doc)
  doc <- gsub("'m", " am", doc)
  doc <- gsub("'d", " would", doc)
  return(doc)
}

#Applied to the Movies$Review
Movies$Review<- sapply(Movies$Review, remove.contractions)
#Reviews were set to lower case
Movies$Review <- sapply(Movies$Review, tolower)
```

For sentiment analysis, I tried to remove words from the reviews that may have a negative sentiment but are used to describe the movie. For example, the word, "Marvel", carries positive sentiment, but it's also the name of a major motion picture studio. Reviews of these movies will inevitably contain multiple instances of this word which may impact the overall sentiment. For this, these words are removed from the Reviews.

```
undesirable_words <- c("marvel", "prison", "mystery", "prison", "joke", "death", "kill", "impossible", "plot", "monster")
```

VII. TidyText Sentiment Analysis

In the code block below, I used the function unnest_tokens by word on the Review variable. This has the effect of parsing out every word from every Reivew into its own observation. I then remove the common stop-words, limiting it to only distinct words, and filter out the undesirable and short words.

```
library(tidytext)
```

```
## Warning: package 'tidytext' was built under R version 3.5.1
```

```
tidy_reviews <- Movies  %>%
  unnest_tokens(word, Review) %>%
  anti_join(stop_words) %>%
  distinct() %>%
  filter(!word %in% undesirable_words) %>%
  filter(nchar(word) > 3)
```

```
## Joining, by = "word"
```

The final result is 10 variables and 56,099 observations.

```
dim(tidy_reviews)
```

```
## [1] 56020    10
```

In the following sections, I applied the three most common lexicon libraries, NRC, Bing, and AFINN to the tidy text data.

Senitment Lexicon - NRC

NRC from Saif Mohammad and Peter Turney. The NRC Emotion Lexicon is a list of English words and their associations with eight basic emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive). The annotations were manually done by crowdsourcing. For more information, please see NRC Word-Emotion Association Lexicon (https://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm)

```
#This code block takes the tidy_reviews dataframe and provides a count of the words per title.
totals <- tidy_reviews %>%
  count(Title) %>%
   rename(total_words = n)

#Next the totals data frame is joined by Title with the tidy_reviews data frame to form movie_co
unts which is breaks out each word of the review into its own entry.
movie_counts <- tidy_reviews %>%
  left_join(totals, by = "Title")

names(movie_counts)
```

```
##  [1] "Rank"         "Title"        "Movie_Studio" "ReleaseDate"
##  [5] "Genre"        "MPAA"         "Gross"        "Tickets"
##  [9] "Critic"       "word"         "total_words"
```

In this code block, I execute the NRC sentiment by performing an inner join between the words in the Movie Reviews and the sentiments in the NRC lexicon.

A unique look at the sentiments show all of the sentiments based on the words in the movie reviews variable.

```
unique(critic_nrc_sentiment$sentiment)
```

```
##  [1] "surprise"     "negative"     "sadness"      "anger"
##  [5] "anticipation" "disgust"      "fear"         "joy"
##  [9] "positive"     "trust"
```

NRC Negative Sentiment

In the code block below:

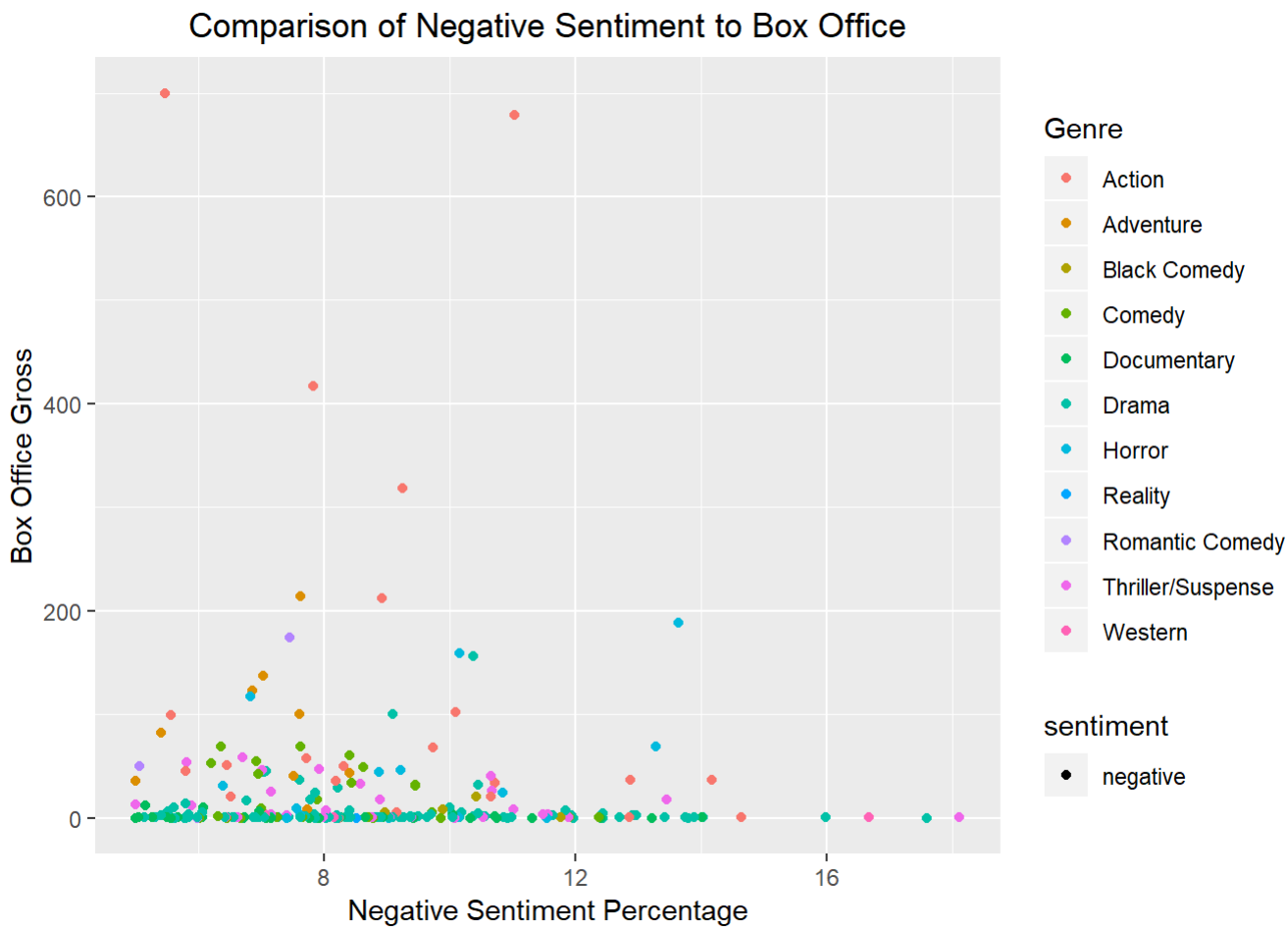
- Grouping by Title and Sentiment, it provides a word count per sentiment
- It then creates a percentage for each sentiment
- Filters out where the negative sentiment is more than 5% of the total number of words.
- Joined the negative sentiment data frame to the Movies data frame

```
negative_nrc_sentiment <- critic_nrc_sentiment %>%
#Count the total words in each review
    count(Title, sentiment, total_words) %>%
    ungroup() %>%
#Create a new variable percent which computes the sentiment divideded by the number of words
    mutate(percent = n / total_words) %>%
#Filter if the sentiment is negative and the percentage of negative sentiment is equal to or gre
ater than 5%
    filter(sentiment == "negative") %>%
    filter(percent >= 0.05) %>%
    arrange(desc(percent))



NRC_negative_sentiment_Box_Office <- left_join(negative_nrc_sentiment, Movies, by= 'Title')
head(NRC_negative_sentiment_Box_Office,10)
```

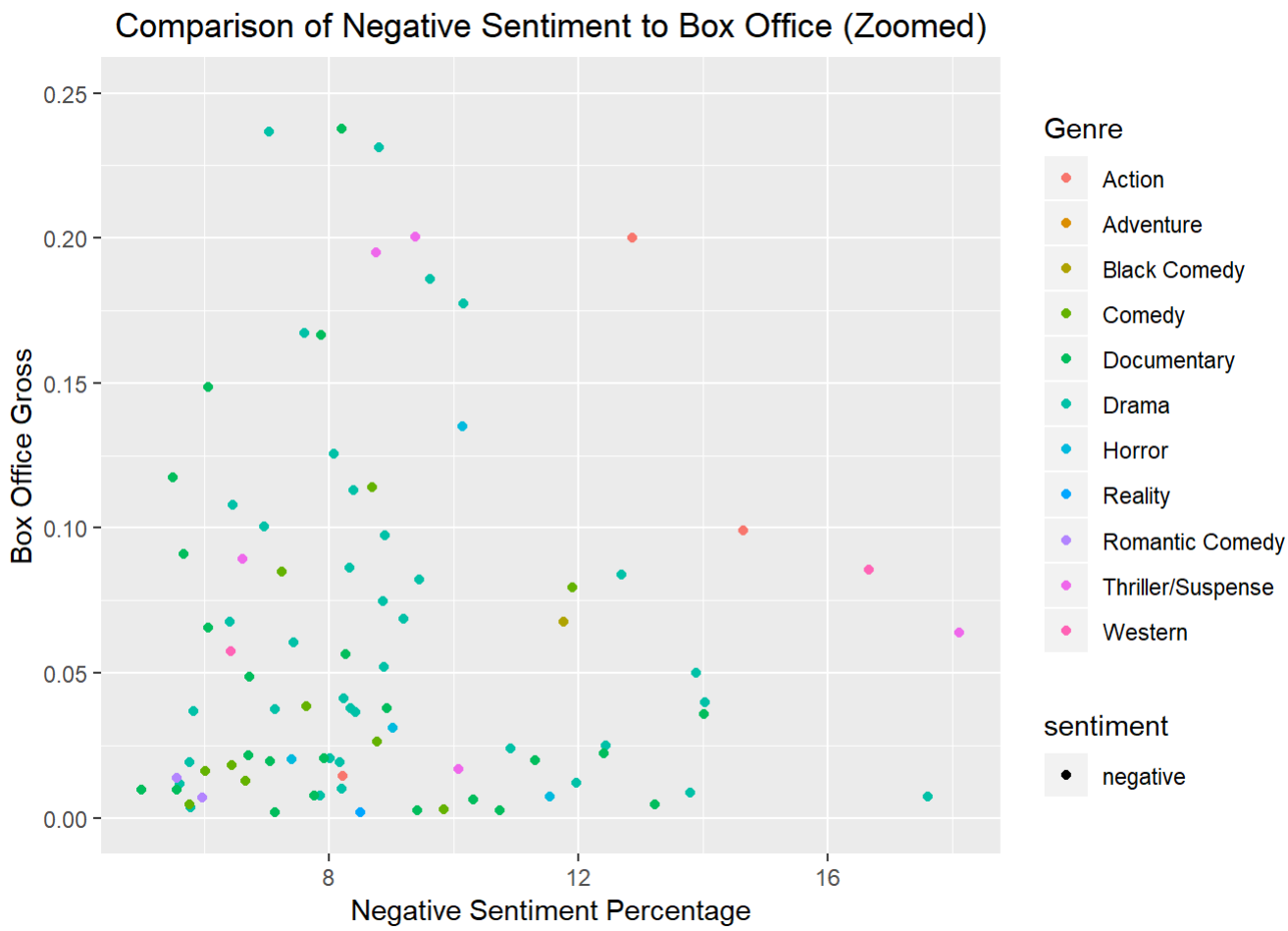
```
## # A tibble: 10 x 14
##    Title        sentiment total_words     n percent  Rank Movie_Studio
##    <chr>        <chr>           <int> <int>   <dbl> <dbl> <fct>
##  1 Nancy        negative          127    23   0.181   364 Samuel Goldwyn ~
##  2 Beauty and ~ negative          125    22   0.176   479 Oscilloscope Pi~
##  3 Sweet Count~ negative          132    22   0.167   343 Samuel Goldwyn ~
##  4 The Happy P~ negative          194    31   0.160   256 Sony Pictures C~
##  5 Revenge      negative          198    29   0.146   334 Neon
##  6 Mile 22      negative          127    18   0.142    77 STX Entertainme~
##  7 Submission   negative          278    39   0.140   390 Great Point Med~
##  8 The Road Mo~ negative          107    15   0.140   403 Oscilloscope Pi~
##  9 Black 47     negative          108    15   0.139   379 IFC Films
## 10 November     negative          145    20   0.138   472 Oscilloscope Pi~
## # ... with 7 more variables: ReleaseDate <date>, Genre <fct>, MPAA <fct>,
## #   Gross <dbl>, Tickets <dbl>, Critic <chr>, Review <chr>
```

In the plots below, I compared reviews with negative percentage of greater than 5% to that film's box office. The second plot zooms in,

```
ggplot(NRC_negative_sentiment_Box_Office , aes(x=percent*100, y=Gross/1000000, fill=sentiment, c
ol=Genre)) +
    geom_point() +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Negative Sentiment Percentage", y="Box Office Gross", title="Comparison of Negative
Sentiment to Box Office")
```

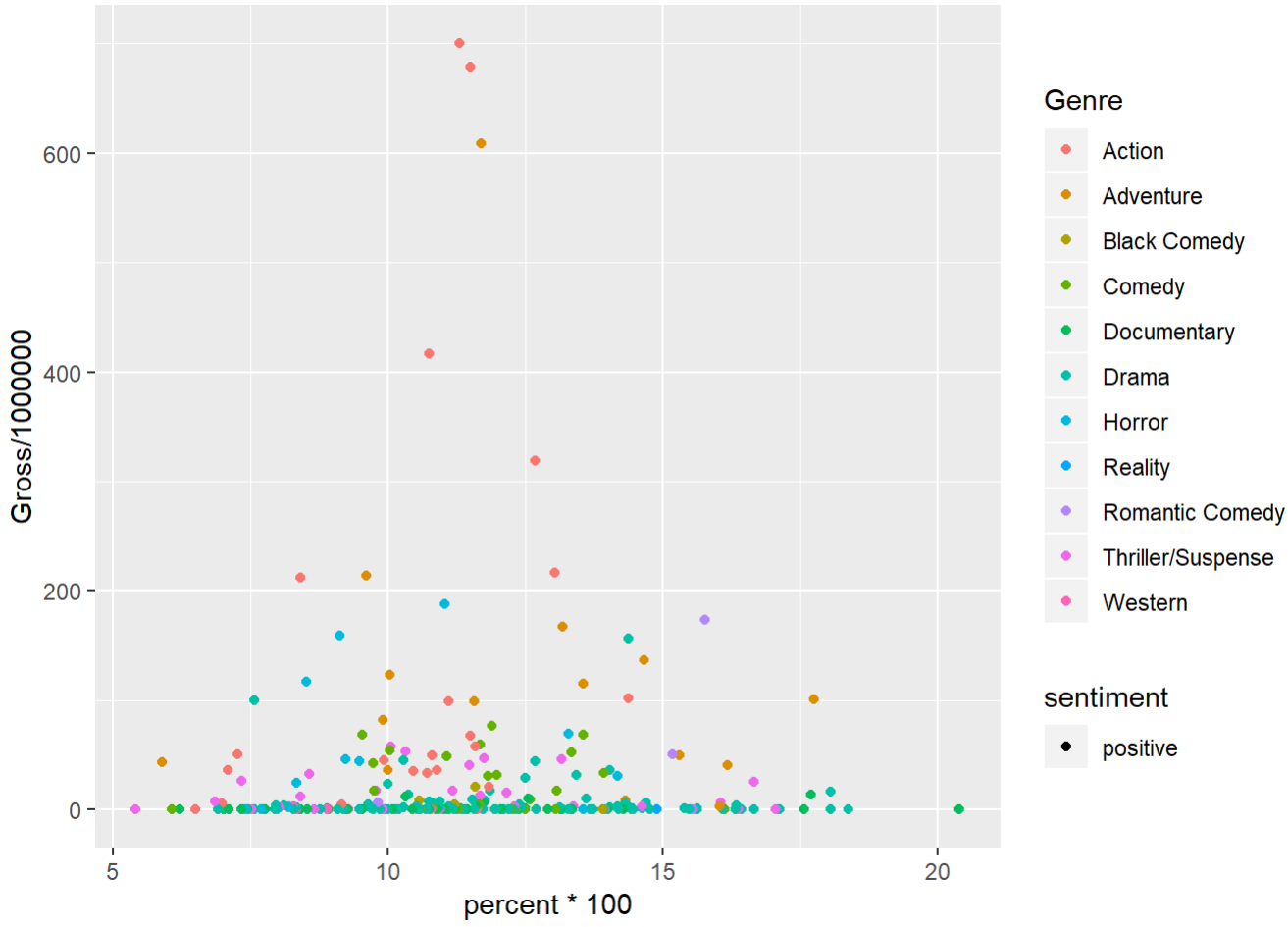



```
ggplot(NRC_negative_sentiment_Box_Office , aes(x=percent*100, y=Gross/1000000, fill=sentiment, c
ol=Genre)) +
    geom_point() +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Negative Sentiment Percentage", y="Box Office Gross", title="Comparison of Negative
Sentiment to Box Office (Zoomed)")+
      ylim(c(0,.25))
```

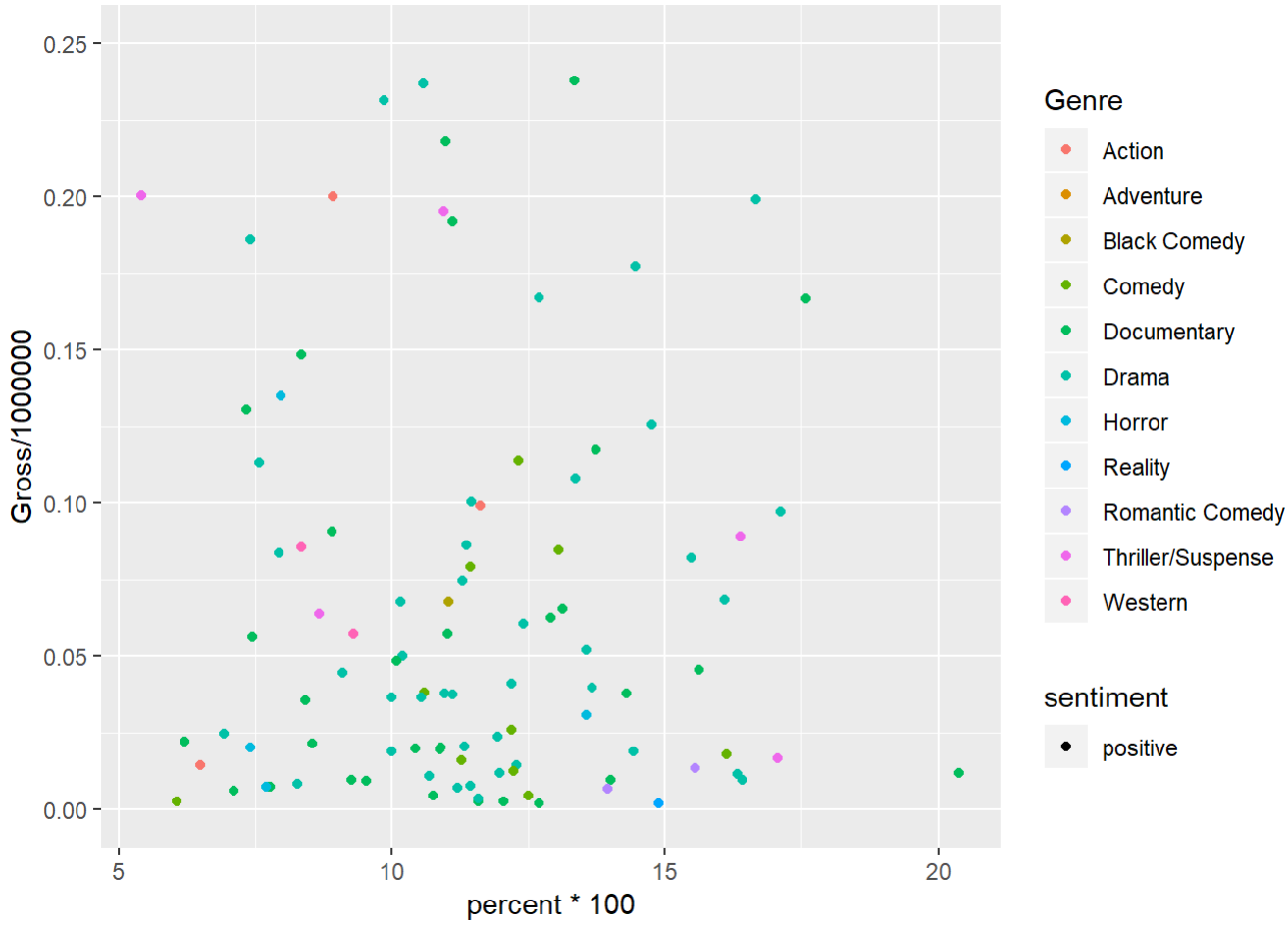



```
positive_nrc_sentiment <- critic_nrc_sentiment %>%
    count(Title, sentiment, total_words) %>%
    ungroup() %>%
    mutate(percent = n / total_words) %>%
    filter(sentiment == "positive") %>%
    filter(percent >= 0.05) %>%
    arrange(desc(percent))

NRC_positive_sentiment_Box_Office <- left_join(positive_nrc_sentiment, Movies, by= 'Title')
```

```
ggplot(NRC_positive_sentiment_Box_Office , aes(x=percent*100, y=Gross/1000000, fill=sentiment, c
ol=Genre)) +
    geom_point()
```




```
ggplot(NRC_positive_sentiment_Box_Office, aes(x=percent*100, y=Gross/1000000, fill=sentiment, co
l=Genre)) +
    geom_point() +
    ylim(c(0,.25))
```



Senitment Lexicon - BING

The Bing lexicon was developed by Bing Liu and collaborators. Opinion Mining, Sentiment Analysis, and Opinion Spam Detection (https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html) The bing lexicon categorizes words in a binary fashion into positive and negative categories.

The same NRC workflow process was followed using the Bing lexicon

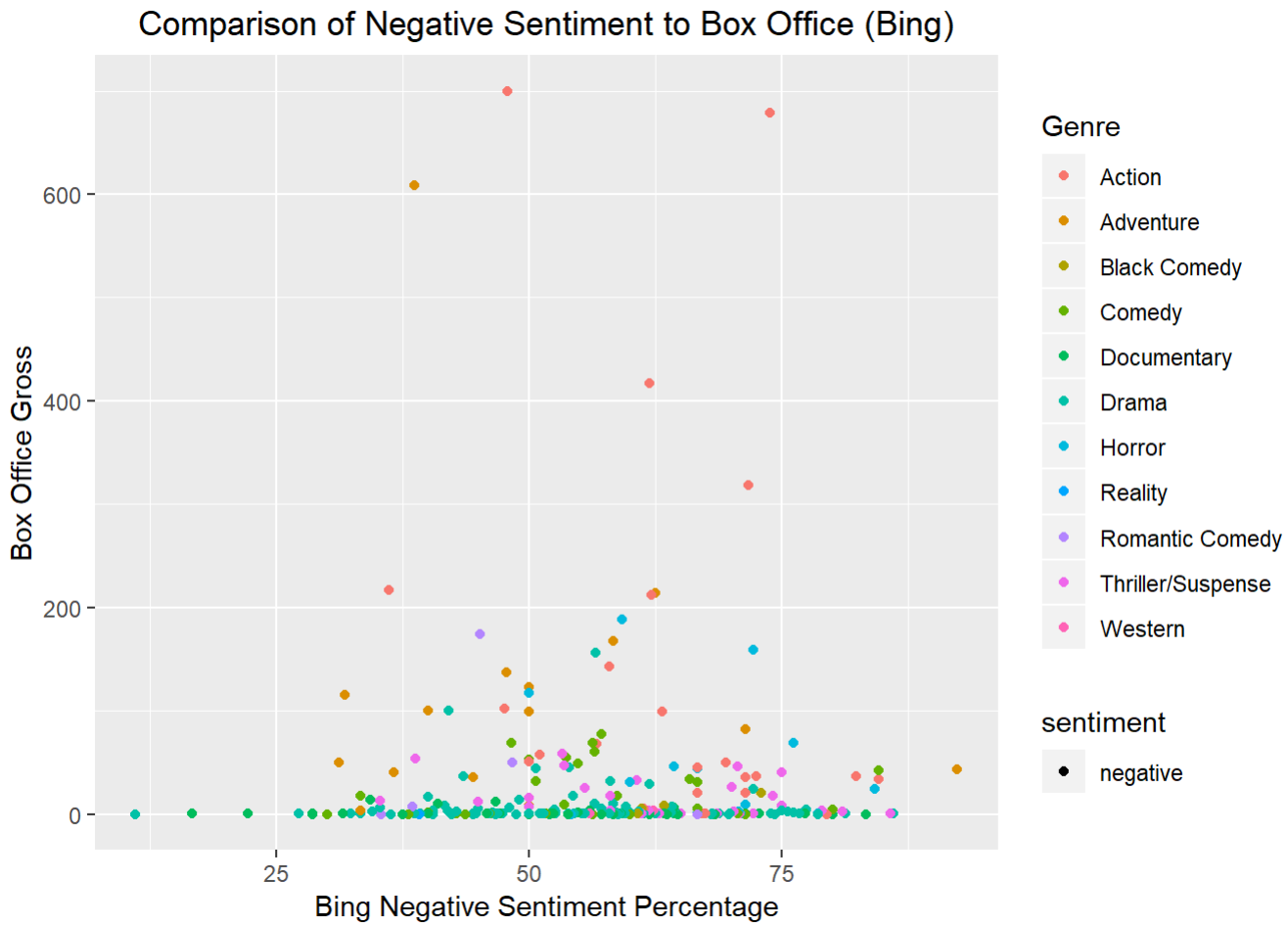
```
critic_bing_sentiment <- movie_counts %>%
    inner_join(get_sentiments("bing")) %>%
    count(Title, Genre, sentiment, sort = TRUE) %>%
    ungroup()
```

```
## Joining, by = "word"
```

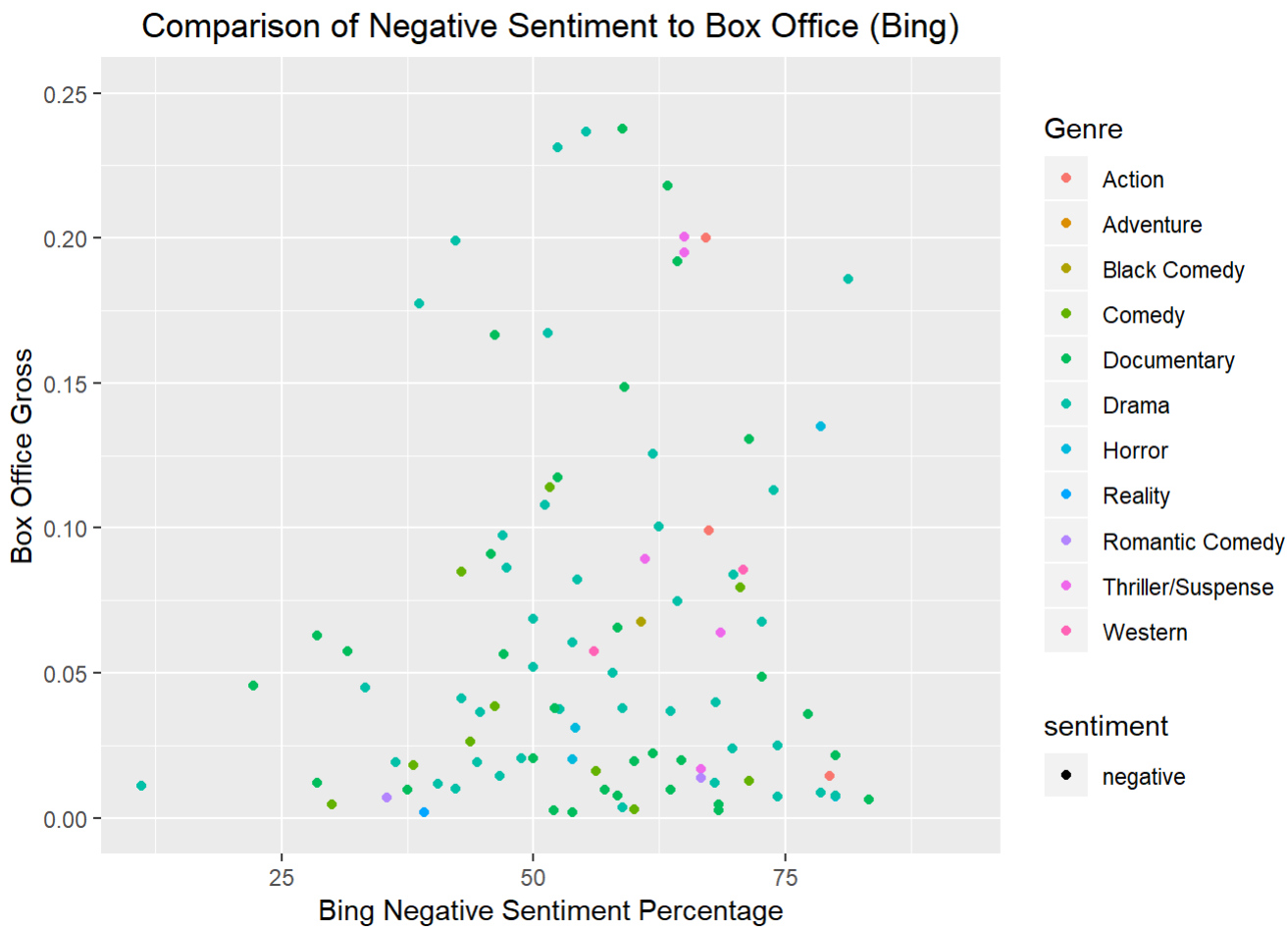
BING negative sentiments

```
BING_negative_sentiment <- critic_bing_sentiment %>%
    group_by(Title) %>%
    mutate(Total = sum(n), percent = n / Total) %>%
    filter(sentiment == "negative") %>%
    filter(percent >= 0.05) %>%
    arrange(desc(percent))
```

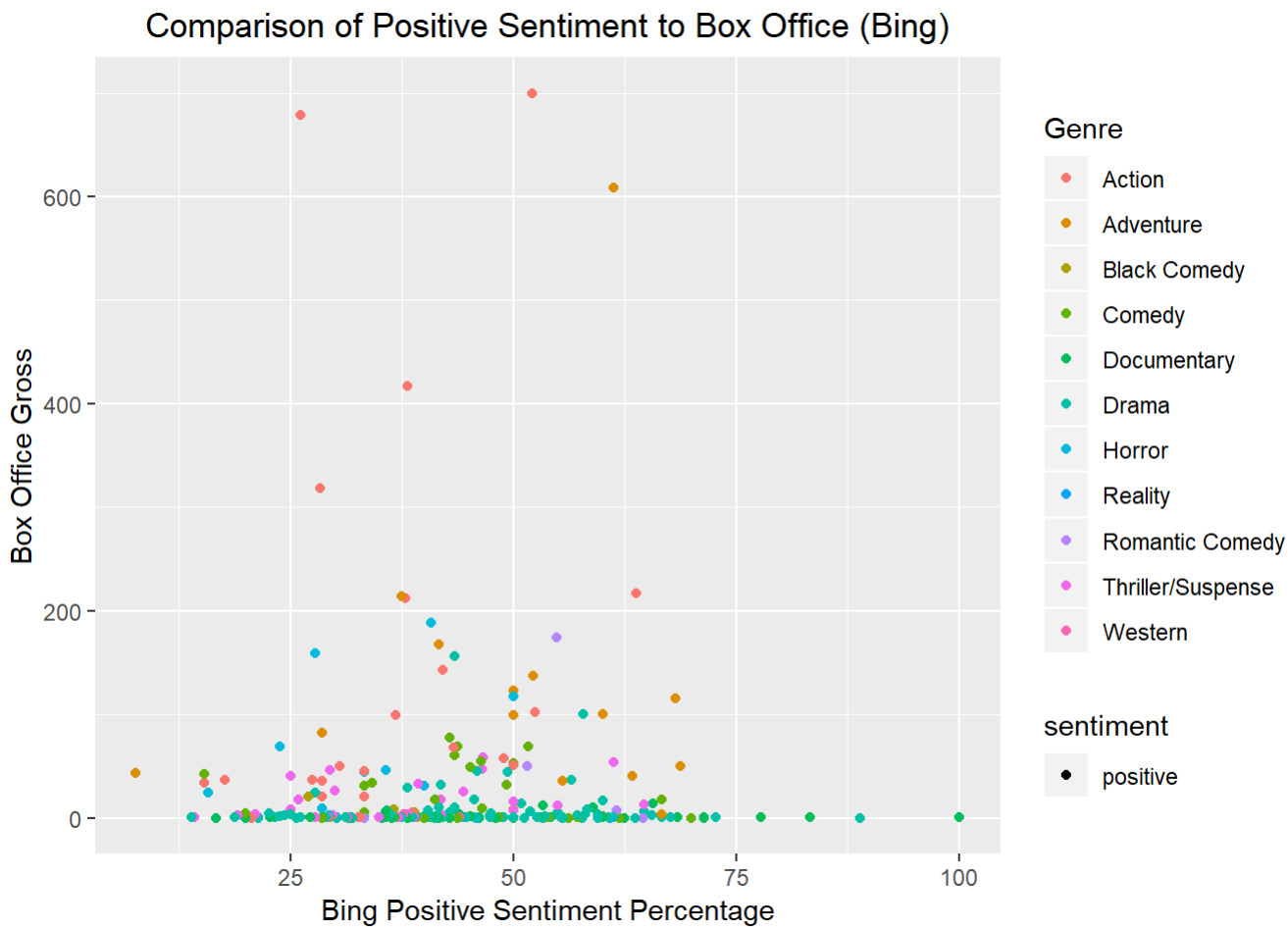
```
Bing_negative_sentiment_Box_Office <- left_join(BING_negative_sentiment , Movies, by= c('Title', 'Genre'))

ggplot(Bing_negative_sentiment_Box_Office, aes(x=percent*100, y=Gross/1000000, fill=sentiment, col=Genre)) +
    geom_point() +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Bing Negative Sentiment Percentage", y="Box Office Gross", title="Comparison of Negative Sentiment to Box Office (Bing)")
```



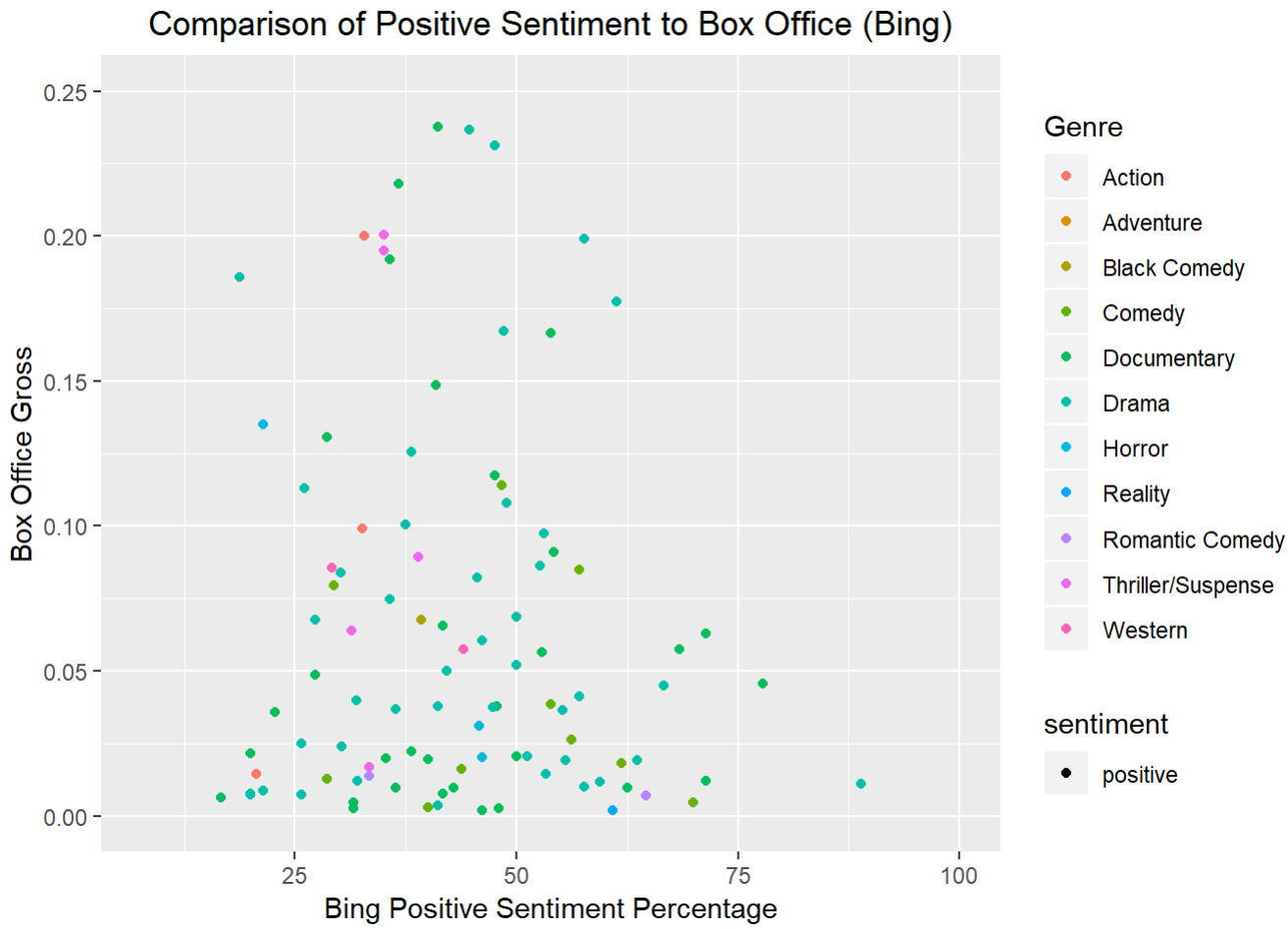

```
ggplot(Bing_negative_sentiment_Box_Office, aes(x=percent*100, y=Gross/1000000, fill=sentiment, col=Genre)) +
    geom_point() +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Bing Negative Sentiment Percentage", y="Box Office Gross", title="Comparison of Negative Sentiment to Box Office (Bing)") +
    ylim(c(0,.25))
```




```
BING_positive_sentiment <- critic_bing_sentiment %>%
    group_by(Title) %>%
    mutate(Total = sum(n), percent = n / Total) %>%
    filter(sentiment == "positive") %>%
    filter(percent >= 0.05) %>%
    arrange(desc(percent))

Bing_positive_sentiment_Box_Office <- left_join(BING_positive_sentiment , Movies, by= c('Title',
'Genre'))

ggplot(Bing_positive_sentiment_Box_Office, aes(x=percent*100, y=Gross/1000000, fill=sentiment, c
ol=Genre)) +
    geom_point() +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Bing Positive Sentiment Percentage", y="Box Office Gross", title="Comparison of Pos
itive Sentiment to Box Office (Bing)")
```




```
ggplot(Bing_positive_sentiment_Box_Office, aes(x=percent*100, y=Gross/1000000, fill=sentiment, c
ol=Genre)) +
    geom_point() +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Bing Positive Sentiment Percentage", y="Box Office Gross", title="Comparison of Pos
itive Sentiment to Box Office (Bing)") +
    ylim(c(0,.25))
```



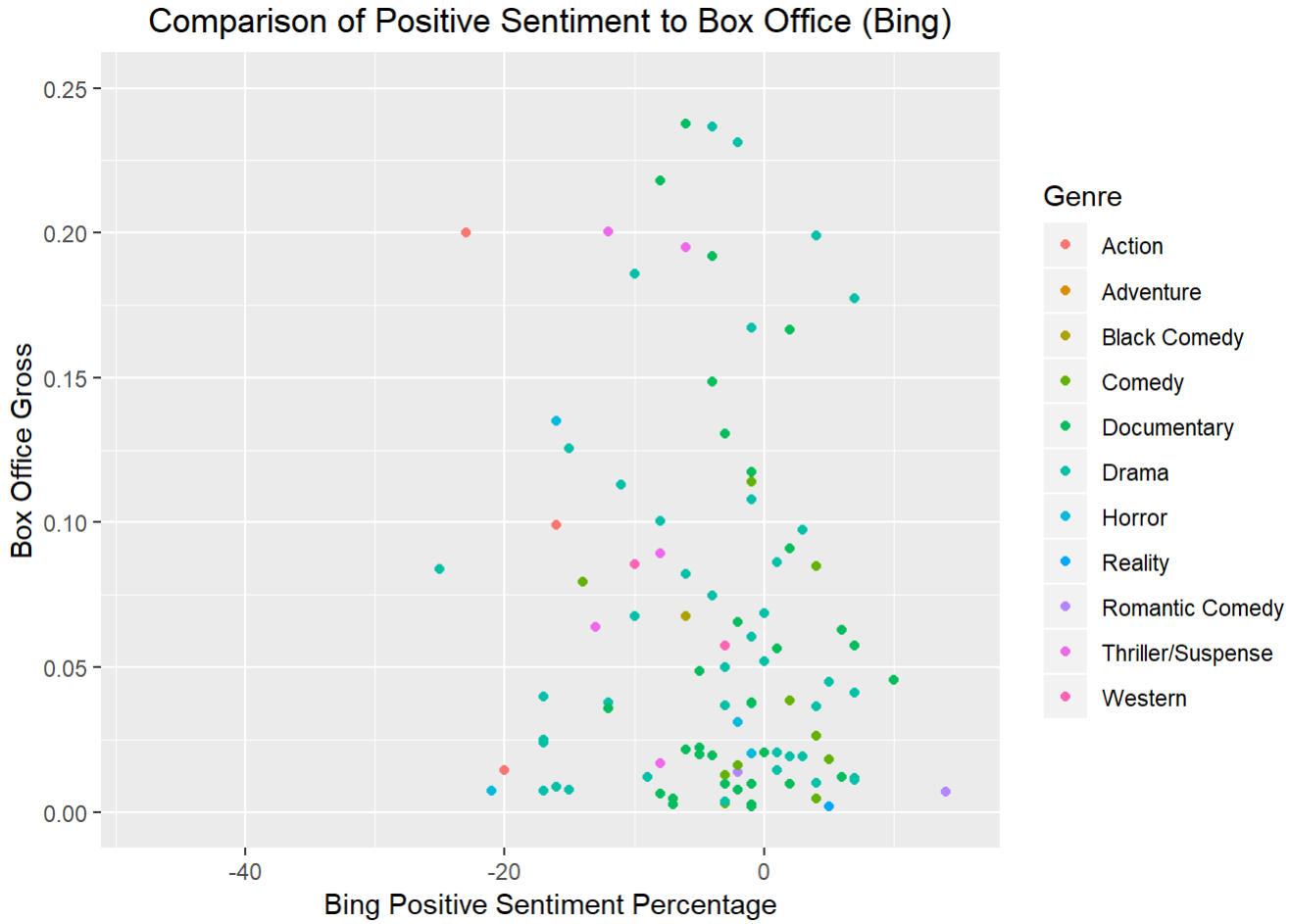
Bing Overall Sentiment

Since the Bing lexicon provides a binary choice, I took a look at the overall difference between positive and negative sentiments. In the code block below, the variable Total_Sentiment represents the difference between the positive and negative sentiment of the Review. Note the film with the highest overall total sentiment was “A Wrinkle in Time”, and the film with the lowest Total_sentiment was “Halloween”.

```
critic_bing_sentiment2 <- critic_bing_sentiment %>%
spread(sentiment, n, fill = 0) %>%
mutate(Total_sentiment = positive-negative) %>%
arrange(desc(Total_sentiment))

critic_bing_total_sentiment <- left_join(critic_bing_sentiment2, Movies, by= c('Title','Genre'))
head(critic_bing_total_sentiment,5)
```

```
## # A tibble: 5 x 13
##   Title    Genre   negative positive Total_sentiment  Rank Movie_Studio
##   <chr>    <fct>      <dbl>    <dbl>           <dbl> <dbl> <fct>
## 1 A Wrink~ Advent~       30       45              15    28 Walt Disney
## 2 Duck Bu~ Romant~       17       31              14   481 The Orchard
## 3 Incredi~ Advent~       24       38              14     3 Walt Disney
## 4 Ant-Man~ Action        17       30              13     8 Walt Disney
## 5 Science~ Docume~        3       15              12   281 National Geogr~
## # ... with 6 more variables: ReleaseDate <date>, MPAA <fct>, Gross <dbl>,
## #   Tickets <dbl>, Critic <chr>, Review <chr>
```

```
ggplot(critic_bing_total_sentiment, aes(x=Total_sentiment, y=Gross/1000000, col=Genre)) +
    geom_point() +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Bing Positive Sentiment Percentage", y="Box Office Gross", title="Comparison of Pos
itive Sentiment to Box Office (Bing)")+
    ylim(c(0,.25))
```



The words driving the sentiment analysis are shown in the code and visualization below. The conclusion that jumped out at me was that the words driving both the positive and negative sentiments may not be reflective of the review as a whole. In reading a NY Times review, I noticed that they included a somewhat detailed summary of the movie.

Thus, words in the review reflect both the review and summary of the movie. For example, a horror movie could be violent, have tension, fear, etc., but the review of the movie may be positive.

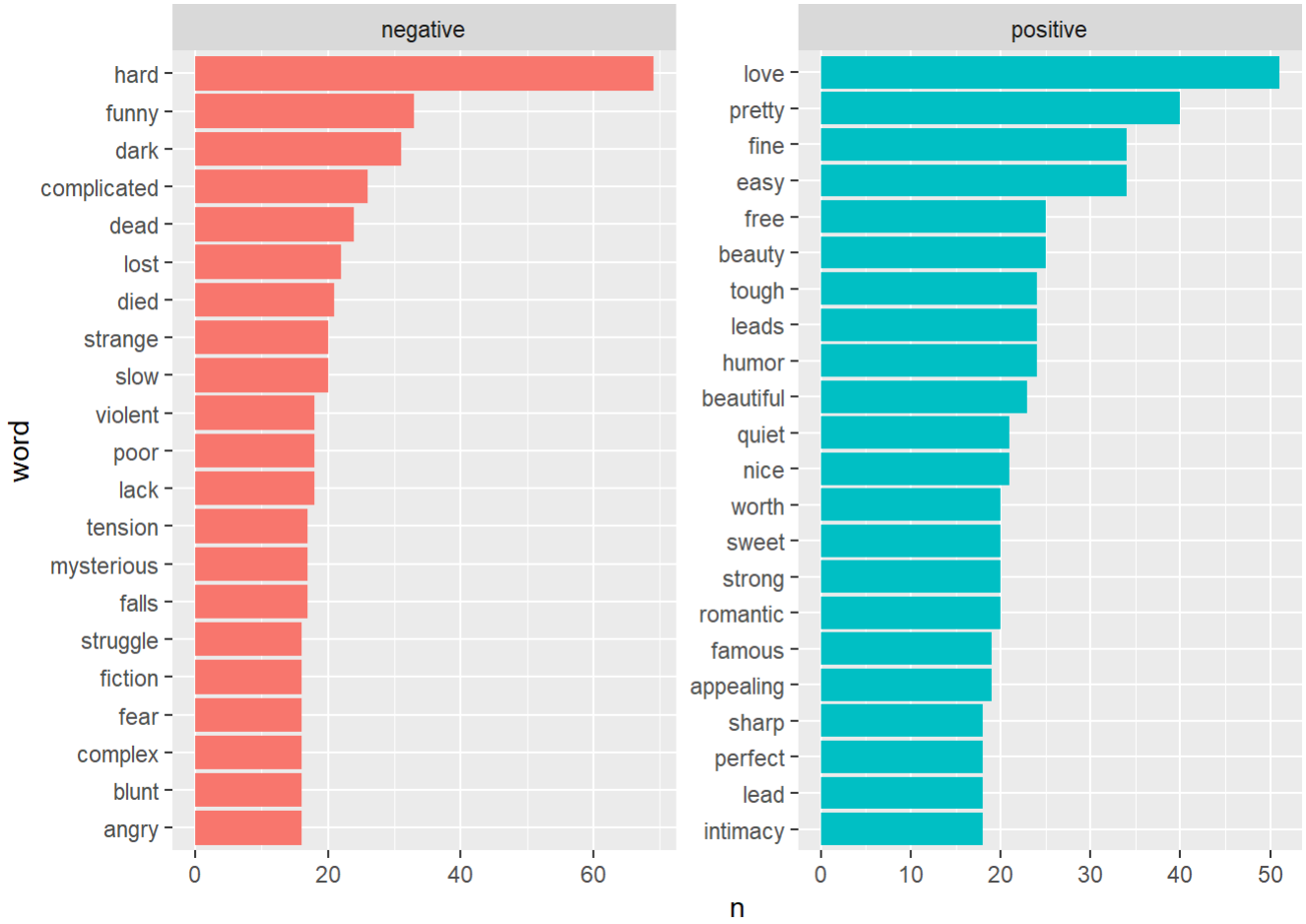
```
word_counts <- movie_counts %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment)
```

```
## Joining, by = "word"
```

```
top_words <- word_counts %>%
  group_by(sentiment) %>%
  top_n(20) %>%
  ungroup() %>%
  mutate(word = reorder(word, n))
```

```
## Selecting by n
```

```
ggplot(top_words, aes(x=word, y=n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free") +
  coord_flip()
```



Senitment Lexicon - AFINN

The third lexicon that I looked at is the AFINN lexicon. The AFINN lexicon assigns words with a score that runs between -5 and 5, with negative scores indicating negative sentiment and positive scores indicating positive sentiment.(More information about AFINN)[http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010 (http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010)]

Similar to the NRC and Bing workflows, I joined the words in each movie review to the AFINN lexicon by word.

```
critic_afinn_sentiment <- movie_counts %>%
    inner_join(get_sentiments("afinn"))
```

```
## Joining, by = "word"
```

As stated previously, the AFINN lexicon assigns words with a score that runs between -5 and 5. In the code below grouping by the title of each movie, I was able to calccute a sentiment score for each film.

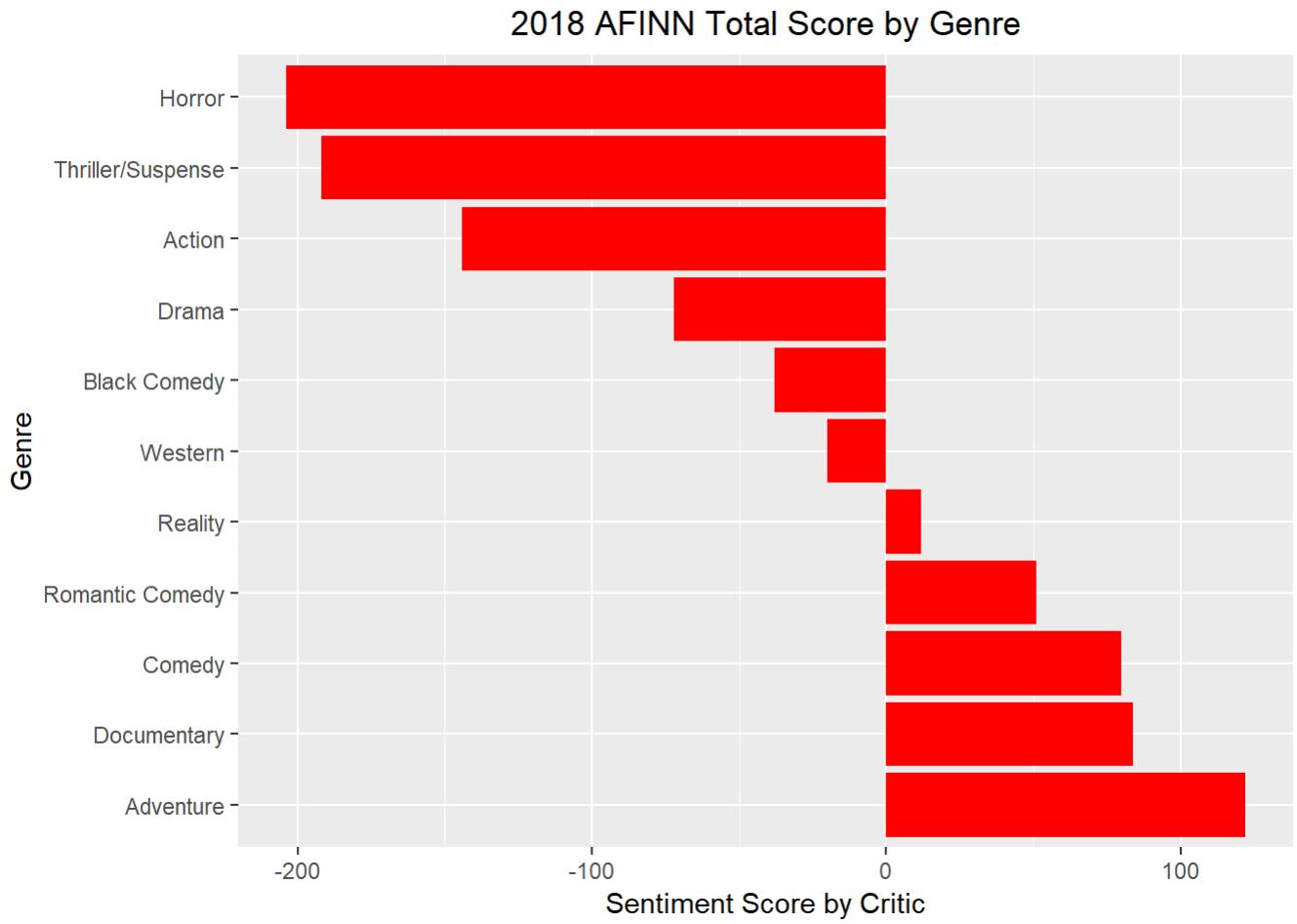
```
critic_afinn_sentiment_scores <-  critic_afinn_sentiment %>%
                                  group_by(Title) %>%
                                  mutate(Total = sum(score)) %>%
                                  select(Rank,Title,Movie_Studio,ReleaseDate,Genre,MPAA,Gross,Ti
ckets,Critic,total_words,Total) %>%
                                  distinct() %>%
                                  arrange(desc(Total))


head(critic_afinn_sentiment_scores,5)
```

```
## # A tibble: 5 x 11
## # Groups:   Title [5]
##    Rank Title  Movie_Studio ReleaseDate Genre  MPAA   Gross Tickets Critic
##   <dbl> <chr>  <fct>        <date>      <fct>  <fct>  <dbl>   <dbl> <chr>
## 1    28 A Wri~ Walt Disney  2018-03-09  Adven~ PG    1.00e8  1.12e7 A. O.~
## 2   321 Ryuic~ MUBI         2018-07-06  Docum~ Not ~ 1.17e5  1.31e4 BEN K~
## 3   119 Eight~ A24          2018-07-13  Drama  R     1.35e7  1.51e6 MANOH~
## 4   162 Blind~ Lionsgate    2018-07-20  Drama  R     4.33e6  4.83e5 GLENN~
## 5    53 Overb~ Lionsgate    2018-05-04  Roman~ PG-13 5.03e7  5.61e6 GLENN~
## # ... with 2 more variables: total_words <int>, Total <int>
```

In the next two blocks, I was able to show the AFINN total score by genre and the box office gross by genre.

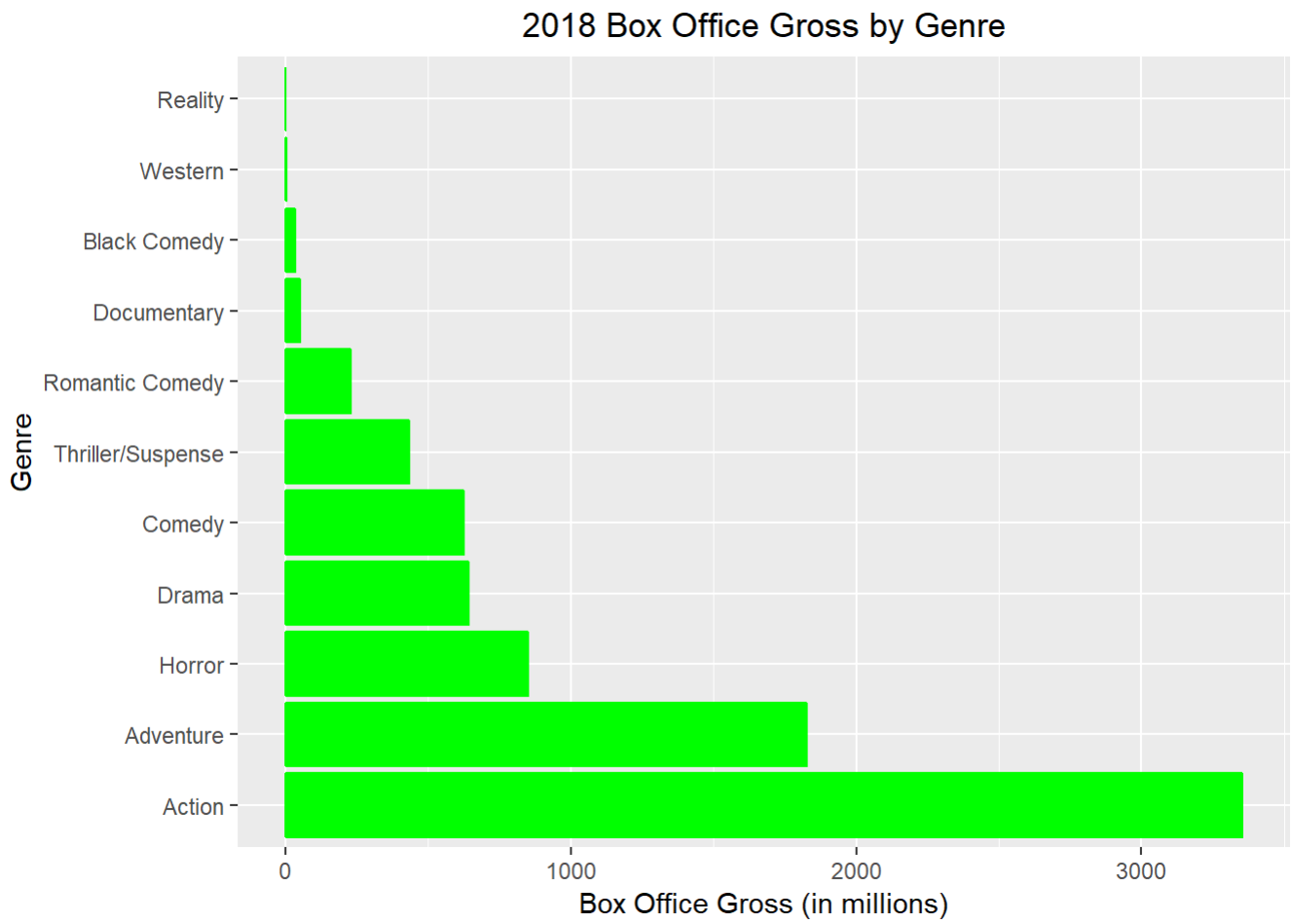
Horror movies have the lowest sentiment score from NY Time critics. However, as stated earlier, this could be due to the fact that the Times critics include summaries of the film in their reviews. Horror movies, by their very nature, are going to have more words with a higher negative sentiment than a comedy or an Adventure film.

```
critic_sentiment_by_genre <- critic_afinn_sentiment_scores %>%
    group_by(Genre) %>%
    summarise(Total_by_Genre = sum(Total)) %>%
    select(Genre, Total_by_Genre) %>%
    arrange(desc(Total_by_Genre))


ggplot(data=critic_sentiment_by_genre, aes(x=reorder(Genre, -Total_by_Genre), y=Total_by_Genre))
+
     geom_bar(stat="identity", position=position_dodge(),  fill="red") +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Genre", y="Sentiment Score by Critic", title="2018 AFINN Total Score by Genre")+
    coord_flip()
```



Regardless of whether the sentiment is driven by the critic's impression of the movie or a summary of the movie's content, this project is focused on whether the sentiment has any effect on the box office. These two plots also show no cause/effect. Movie genres, like Horror, Thriller/Suspense, Action, and Drama, have negative overall sentiment scores but they have positive box office gross.

```
Gross_by_Genre <- critic_afinn_sentiment_scores %>%
    group_by(Genre) %>%
    summarise(Genre_Gross = sum(Gross)) %>%
    select(Genre, Genre_Gross) %>%
    arrange(desc(Genre_Gross))


ggplot(data=Gross_by_Genre, aes(x=reorder(Genre, -Genre_Gross), y=Genre_Gross/1000000)) +
     geom_bar(stat="identity", position=position_dodge(), col="green", fill="green") +
     theme(plot.title = element_text(hjust = 0.5)) +
     labs(x="Genre", y="Box Office Gross (in millions)", title="2018 Box Office Gross by Genre")
+
    coord_flip()
```



VIII. Correlation and Regression Analysis

So far in the project, I have not address the Hypothesis test that I set up at the beginning.

H0: The sentiment of a NY Times movie review has no effect on box office performance. HA: The sentiment of a NY Time movie review has an effect on box office performance.

The most definitive way to answer that question would be to look at the correlation between sentiment and box, and more precisely, use Simple Linear Regression to establish a relationsip.

None of the three lexicons, NRC, Bing, nor AFINN showed a strong corrleation between the sentiment of the movie and it's box office gross. Strong correlation would be +1 or -1 (for a negative correlation). There is no correlation between sentiment and box office gross.

NRC Correlation Scores

```
cor(NRC_negative_sentiment_Box_Office$Gross, NRC_negative_sentiment_Box_Office$percent)
```

```
## [1] -0.04584898
```

BING Correlation

```
cor(critic_bing_total_sentiment$Gross, critic_bing_total_sentiment$Total_sentiment)
```

```
## [1] -0.08440203
```

AFINN Correlation

```
cor(critic_afinn_sentiment_scores$Gross, critic_afinn_sentiment_scores$Total)
```

```
## [1] 0.02822699
```

Simple Linear Regression

The p-values for each of the sentiment lexicons, (0.4054,0.154, and 0.642), are all >0.05. Such a large p-value suggests that changes in the predictor are not associated with changes in the response. IOW, sentiment of a NY Times review does not have an affect on the movie's box office. Additionally, the R-squared values are all 0 (0.002947, 0.007472, 0.0007968) which means that none of the percentage of the response variable variation that is explained by a linear model.

In sum, we cannot reject the null hypothesis.

```
Movie_Simple_LR_NRC <- lm(Gross~percent, data = NRC_negative_sentiment_Box_Office)
summary(Movie_Simple_LR_NRC)
```

```
##
## Call:
## lm(formula = Gross ~ percent, data = NRC_negative_sentiment_Box_Office)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -32880254 -28039899 -22888822  -2602389 667836262
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   40067515   18615609   2.152   0.0324 *
## percent     -143549068  204458813  -0.702   0.4833
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 79030000 on 234 degrees of freedom
## Multiple R-squared:  0.002102,   Adjusted R-squared:  -0.002162
## F-statistic: 0.4929 on 1 and 234 DF,  p-value: 0.4833
```

```
Movie_Simple_LR_BING <- lm(Gross~Total_sentiment, data = critic_bing_total_sentiment)
summary(Movie_Simple_LR_BING )
```

```
##
## Call:
## lm(formula = Gross ~ Total_sentiment, data = critic_bing_total_sentiment)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -48189761 -29573473 -24343775  -5789013 676361770
##
## Coefficients:
##                 Estimate Std. Error t value   Pr(>|t|)
## (Intercept)     25908623    5673466   4.567 0.00000753 ***
## Total_sentiment  -736942     528498  -1.394      0.164
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 83520000 on 271 degrees of freedom
## Multiple R-squared:  0.007124,   Adjusted R-squared:  0.00346
## F-statistic: 1.944 on 1 and 271 DF,  p-value: 0.1643
```

```
Movie_Simple_LR_AFINN <- lm(Gross~Total, data = critic_afinn_sentiment_scores)
summary(Movie_Simple_LR_AFINN)
```

```
##
## Call:
## lm(formula = Gross ~ Total, data = critic_afinn_sentiment_scores)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -33912498 -29644364 -26823686  -5478153 667240604
##
## Coefficients:
##             Estimate Std. Error t value     Pr(>|t|)
## (Intercept) 29704971    5089526   5.836 0.0000000152 ***
## Total         173000     372154   0.465        0.642
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 83780000 on 271 degrees of freedom
## Multiple R-squared:  0.0007968,  Adjusted R-squared:  -0.00289
## F-statistic: 0.2161 on 1 and 271 DF,  p-value: 0.6424
```

IX. Conclusions

Given the analysis above, we cannot reject the null hypothesis that the senitment of a NY Times movie review has no effect on the box revenue of that review. The caveat to this analysis is that the sentiment is based on words used in the review. NY Times movie reviews included summaries of the movie itself. Also, their reviews are very sublte and nuanced. Rarely do their reviews use hyperbolic language about the movie. Looking at the word cloud below for the top grossing films of 2018, you don’t see hyperbolic words used to either praise or demean the movie.

Even with that said, the findings of this project show that a NY Times movie review is not a predictor of box office success.

```
dev.new(width = 1000, height = 1000, unit = "px")
top25 <- Movies %>%
          filter(Rank <= 34) %>%
          unnest_tokens(word, Review) %>%
          anti_join(stop_words) %>%
          distinct() %>%
          filter(!word %in% undesirable_words) %>%
          filter(nchar(word) > 3) %>%
          inner_join(get_sentiments("bing")) %>%
          count(word, sentiment, sort = TRUE) %>%
          filter(n > 3) %>%
          arrange(desc(n))
```

```
## Joining, by = "word"
## Joining, by = "word"
```

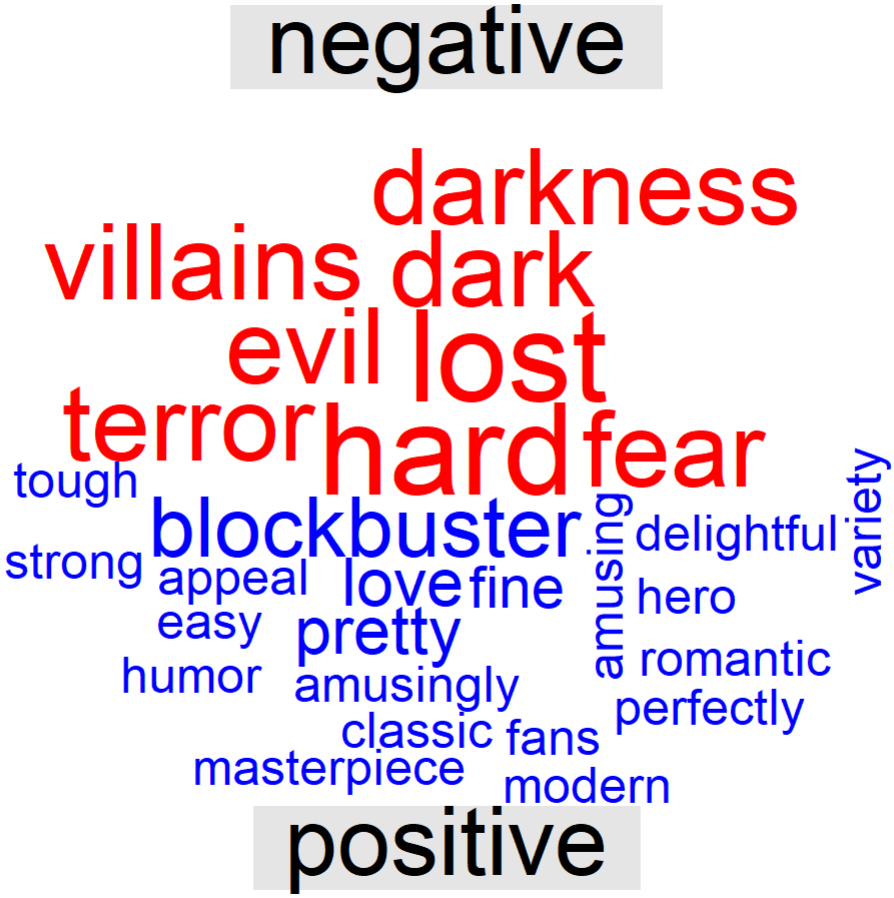
```
library(wordcloud2)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
top25  %>%
       acast(word ~ sentiment, value.var = "n", fill = 0) %>%
            comparison.cloud(colors = c("red", "blue"),
                  max.words = 500)
```



X. Citations

1. Machine Learning and NLP using R: Topic Modeling and Music Classification by Debbie Liske (https://www.datacamp.com/community/tutorials/ML-NLP-lyric-analysis)
2. NLP Tutorial (https://www.datacamp.com/community/tutorials/R-nlp-machine-learning)
3. Text Mining with R by Julia Silge and David Robinson (https://www.tidytextmining.com/)