Data Wrangling - News Article Analysis

Advaith Rao Ayush Oturkar Jeanie Hung Vanshita Gupta

29 April 2023

Contents

1 Loading required Packages

```
# Specify your packages
my_packages <- c("tidyverse",
                 "gutenbergr",
                 "dplyr",
                 "tidytext",
                 "tidyverse",
                 "dplyr",
                 "textdata",
                 "tm",
                 "topicmodels",
                 "reshape2",
                 "gridExtra" ,
                 "httr",
                 "jsonlite",
                 "wordcloud2",
                 "webshot",
                 "htmlwidgets",
                 "glue",
                 "textmineR",
                 "viridis",
                 "text2vec",
                 "webshot",
                 "stringr")

# Get a list of packages which are not installed in the background
not_installed <- my_packages[
  !(my_packages %in% installed.packages()[ , "Package"])
]    # Extract not installed packages

# Install all the non-installed packages
if(length(not_installed)) install.packages(not_installed)

library(tidyverse)
library(gutenbergr)
library(dplyr)
library(tidytext)
library(textdata)
library(topicmodels)
library(reshape2)
library(gridExtra)
library(ggplot2)
library(httr)
```

```
library(jsonlite)
library(htmlwidgets)
library(glue)
library(wordcloud2)
library(textmineR)
library(topicmodels)
library(tm)
library(viridis)
library(text2vec)
library(webshot)
library(stringr)
library(tm)

webshot::install_phantomjs()
```

2 Combine news data from different sources

```
# Combining the data
#The Data Fetch part of our project can be viewed under the markdown file:
#./webscraping.Rmd

# 1. Read the cnn and bing data
cnn_bbc_df <- read.csv("data/cnn_bbc_news_data_1.csv")
bing_df <- read.csv("data/bing_news_data_1.csv")

# 2. Read the NYtimes data
nytimes_df <- read.csv("data/newcatcher_news_data_nytimes-news.csv")
nytimes_df <- nytimes_df %>% rename(news_source = source)

# 3. Read the associated news data
associated_press_df <- read.csv("data/newcatcher_news_data_associatedpress-news.csv")
associated_press_df <- associated_press_df %>% rename(news_source = source)

# 4. Read the reuters news data
reuters_df <- read.csv("data/newcatcher_news_data_reuters-news.csv")
reuters_df <- reuters_df %>% rename(news_source = source)

# 5. Read the guardian news data
guardian <- read.csv("data/newcatcher_news_data_theguardian-news.csv")
guardian <- guardian %>% rename(news_source = source)

# 6. Read the Washington Post news data
washpost_df <- read.csv("data/newcatcher_news_data_washingtonpost-news.csv")
washpost_df <- washpost_df %>% rename(news_source = source)

# 7. Read the CNBC Post news data
cnbc_news_df <- read.csv("data/cnbc_news_data_1.csv")
cnbc_news_df <- cnbc_news_df %>% rename(news_source = source)

# 8. Row bind all the above data
news_df <- rbind(
  cnn_bbc_df, bing_df, nytimes_df, associated_press_df, reuters_df, guardian, washpost_d
)

# Lets print the head
head(news_df)
```

```
##   article_type                                                headline
```

```
## 1  NewsArticle    Why Putin cares about Russia's athletes competing abroad
## 2  NewsArticle        Al Jaffee: Record-breaking US cartoonist dies at 102
## 3  NewsArticle                      Listen: British Swimming Championships
## 4  NewsArticle   Bullying and race bias 'commonplace' in equestrian sports
## 5  NewsArticle                    EFL coverage set to remain on Sky Sports
## 6  NewsArticle 100m Olympic champion Fraser-Pryce runs in son's sports day
##                                                                       url
## 1       https://www.bbc.co.uk/news/world-europe-65241285
## 2 https://www.bbc.co.uk/news/entertainment-arts-65238630
## 3     https://www.bbc.co.uk/sport/live/swimming/65190437
## 4        https://www.bbc.co.uk/sport/equestrian/65192884
## 5          https://www.bbc.co.uk/sport/football/65167442
## 6      https://www.bbc.co.uk/sport/av/athletics/65148492
##
## 1 While Russia's brutal invasion of Ukraine drags into its second year, its athletes
## 2 Award-winning American cartoonist Al Jaffee, renowned for his work on satirical mag
## 3  British record holder, Ben Proud will be one of the favourites in the 50m freestyl
## 4 The research was commissioned by British Equestrian and its member bodies, and carr
## 5  The EFL's current TV deal was a 35% increase on their previous agreement with Sky
## 6
##           published_at category news_source
## 1 2023-04-16T00:03:50Z   sports    bbc-news
## 2 2023-04-11T10:01:57Z   sports    bbc-news
## 3 2023-04-08T16:43:50Z   sports    bbc-news
## 4 2023-04-05T23:01:19Z   sports    bbc-news
## 5 2023-04-03T15:43:10Z   sports    bbc-news
## 6 2023-04-01T11:13:59Z   sports    bbc-news
```

3 Data Cleaning

```r
# 1. Duplicate data drop:
cat("Total Rows before duplicate treatment :", nrow(news_df))
```

```
## Total Rows before duplicate treatment : 5579
```

```r
news_df <- distinct(news_df)
cat("Total Rows after duplicate treatment :", nrow(news_df))
```

```
## Total Rows after duplicate treatment : 5579
```

```r
# 2 Cleaning Heading and description
clean_text_column <- function(df, col_name) {
  # convert text to lowercase
  df[[col_name]] <- tolower(df[[col_name]])

  # remove punctuation
  df[[col_name]] <- str_replace_all(df[[col_name]], "[[:punct:]]", "")

  # remove numbers
  df[[col_name]] <- str_replace_all(df[[col_name]], "[[:digit:]]", "")

  # remove non-ASCII characters
  df[[col_name]] <- str_replace_all(df[[col_name]], "[^[:ascii:]]", "")

  # remove specific special characters
  df[[col_name]] <- str_replace_all(df[[col_name]], "\\+|\\-|\\/|\\n", " ")

  # remove leading/trailing whitespace
  df[[col_name]] <- str_trim(df[[col_name]])

  # remove stop words (optional)
  df[[col_name]] <- removeWords(df[[col_name]], stopwords("english"))

  # return cleaned data.frame
  return(df)
}

# assume your data.frame is called `news_df` and the text column is called `article_te
news_df <- clean_text_column(news_df, "headline")
news_df <- clean_text_column(news_df, "description")
```

```r
unique(news_df$news_source)
```

```
## [1] "bbc-news"             "cnn"                  "bing"
## [4] "nytimes-news"         "associatedpress-news" "reuters-news"
## [7] "theguardian-news"     "washingtonpost-news"  "cnbc-news"
```

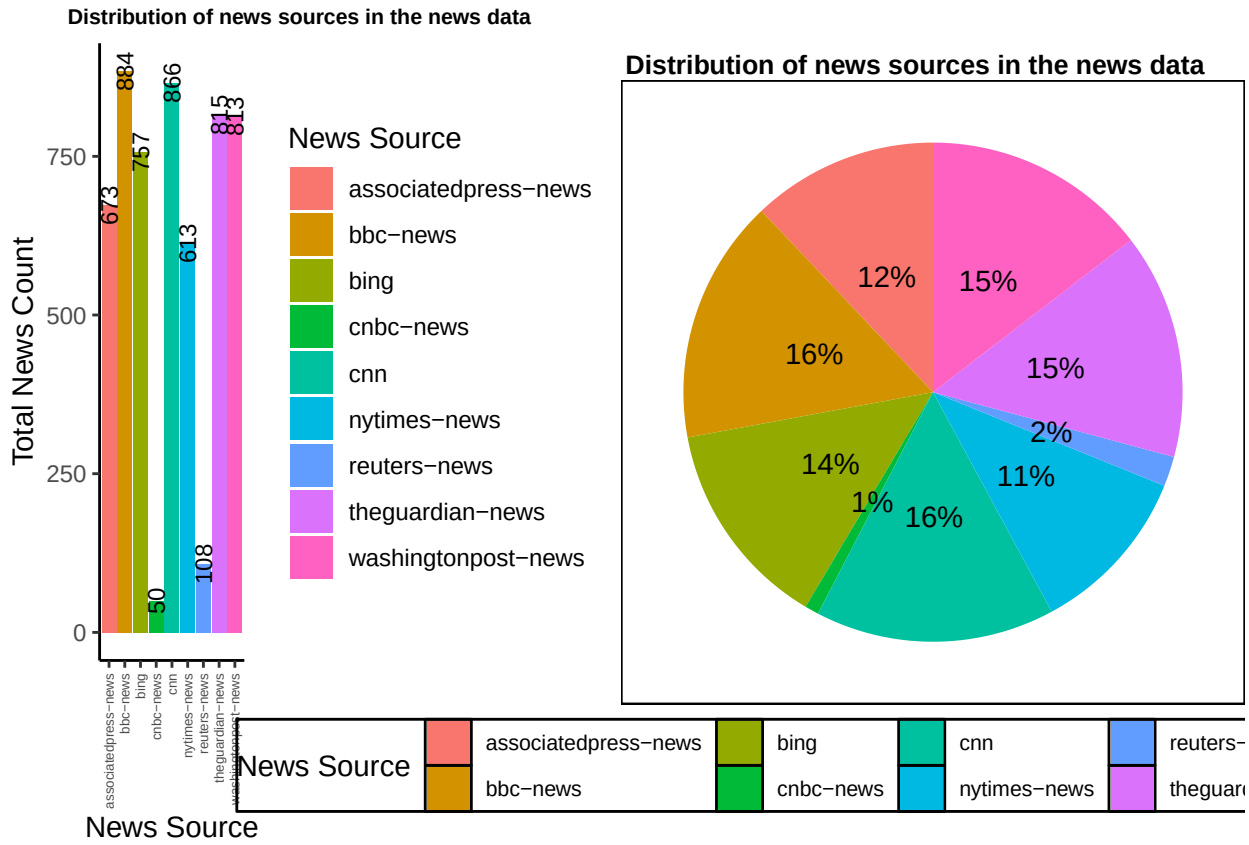
```r
# 3. Cleaning category columns
news_df$category <- tolower(news_df$category)
news_df$category <- gsub("scienceandtechnology", "technology", news_df$category)
news_df$category <- gsub("\\bhealth\\b", "healthcare", news_df$category)

# 4. Add unique news id to the news data
news_df <- news_df %>%
  mutate(news_id = 1:nrow(news_df))
```

4 EDA

4.1 1. Based on News Source

4.1.1 Distribution of news sources in the news data

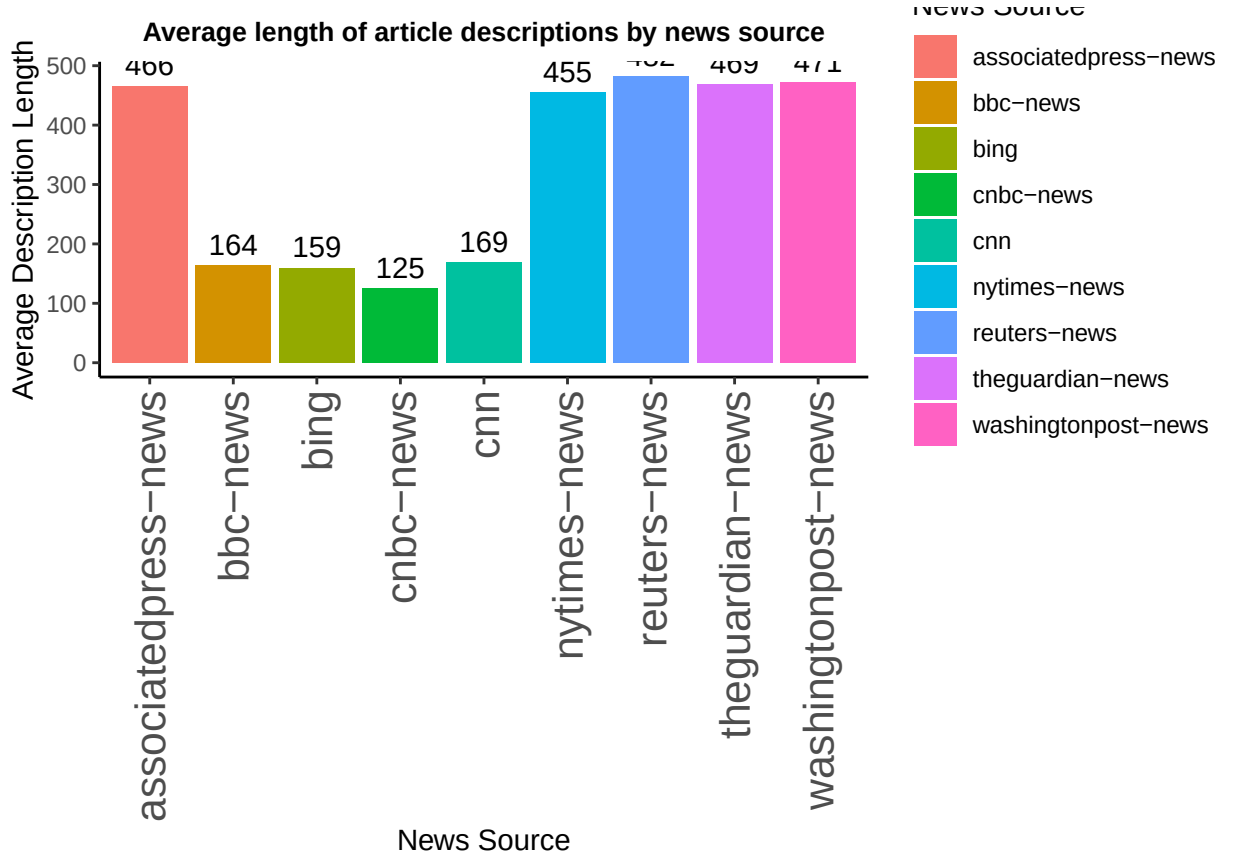
```
# Distribution of news sources in the news data
bar_chart <- news_df %>%
  count(news_source) %>%
  ggplot(aes(x = news_source, y = n, fill = news_source)) +
  geom_bar(stat = "identity") +
  ggtitle("Distribution of news sources in the news data") +
  xlab("News Source") +
  ylab("Total News Count") +
  theme_classic() +
  geom_text(aes(label = n), size = 3, angle = 90) +
  theme(plot.title = element_text(hjust = 0.1, face = "bold", size = 8)) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 5)) +
  labs(fill = "News Source") +
  theme(rect = element_rect(fill = "transparent"))

# Distribution of news sources in the news data - Pie chart view
pie_chart <- news_df %>%
  count(news_source) %>%
  mutate(percent = prop.table(n)*100) %>%
  ggplot(aes(x = "", y = n, fill = news_source)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  ggtitle("Distribution of news sources in the news data") +
  theme_void() +
  theme(plot.title = element_text(hjust = 0.1, face = "bold", size = 10), legend.positio
  theme(rect = element_rect(fill = "transparent")) +
  geom_text(aes(label = paste0(round(percent), "%")), position = position_stack(vjust =
  labs(fill = "News Source")


# create two-column grid
grid_plot <- grid.arrange(bar_chart, pie_chart, ncol = 2)
```




```
# save the plot with a larger size
ggsave(
  "Plots/bar_plot_distribution_of_news_source.png",
  plot = bar_chart,
  height = 6,
  width = 10,
  dpi = 300,
  bg = "transparent"
)

ggsave(
  "Plots/pie_plot_distribution_of_news_source.png",
  plot = pie_chart,
  height = 6,
  width = 10,
  dpi = 300,
  bg = "transparent"
)
```

4.1.2 Average length of article descriptions by news source

```
bar_chart <- news_df %>%
  group_by(news_source) %>%
  summarize(avg_desc_length = mean(nchar(description))) %>%
  ggplot(aes(x = news_source, y = avg_desc_length, fill = news_source)) +
  geom_bar(stat = "identity") +
  ggtitle("Average length of article descriptions by news source") +
  xlab("News Source") +
  ylab("Average Description Length") +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5,  face = "bold", size = 10)) +
  theme(rect = element_rect(fill = "transparent")) +
  geom_text(aes(label = as.integer(avg_desc_length)), vjust = -0.5, size = 4) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 16)) +
  labs(fill = "News Source") +
  theme(rect = element_rect(fill = "transparent"))


print(bar_chart)
```



```r
# save the plot with a larger size
ggsave(
  "Plots/avg_description_length.png",
  plot = bar_chart,
  height = 6,
  width = 8,
  dpi = 300,
  bg = "transparent"
)
```

4.2 Plotting top bi-gram and tri-grams in CNN

4.2.1 Plot n gram function

```r
c <- c(
  "#FF0000",
  "#FF6600",
  "#FFCC00",
  "#FF0066",
  "#CCFF00",
  "#66FF00",
  "#00FF00",
  "#00FF66",
  "#00FFCC",
  "#00CCFF",
  "#0066FF",
  "#0000FF",
  "#6600FF",
  "#CC00FF",
  "#FF00CC"
)

plot_ngram <- function(news_df, news_to_plot, n_for_ngram, fill_color) {

  if (n_for_ngram == 2) {
    n_gram_txt <- "Bigram"
  } else {
    n_gram_txt <- "Trigram"
  }

  n_gram <- news_df %>%
    filter(news_source == news_to_plot) %>%
```

```
    mutate(headline = gsub("[^[:alnum:] ]", "", headline)) %>%
    unnest_tokens(trigram, headline, token = "ngrams", n = n_for_ngram) %>%
    filter(!is.na(trigram)) %>%
    count(trigram, sort = TRUE) %>%
    head(10) %>%
    ggplot(aes(x = trigram, y = n)) +
    geom_bar(stat = "identity", fill = fill_color) +
    ggtitle(paste("Top 10", n_gram_txt, toupper(news_to_plot), "headlines")) +
    xlab(n_gram_txt) +
    ylab("Count") +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1),
          axis.text.y = element_text(size = 16),
          plot.title = element_text(hjust = 0.5, face = "bold")) +
    coord_flip()

  return(n_gram)
}
```

4.3 Looping over all the news sources and getting the n-grams and wordcloud

```
# define the list of news sources
news_sources <- c(
  "bbc-news",
  "cnn",
  "bing",
  "nytimes-news",
  "associatedpress-news",
  "reuters-news",
  "theguardian-news",
  "washingtonpost-news",
  "cnbc-news"
)

i = 1
# loop over the news sources
for (news_to_plot in news_sources) {

  cat("Iteration Number :", i)

  # plot bigram
  n_for_ngram <- 2
```

```r
fill_color <- c[i]
n_gram <- plot_ngram(news_df, news_to_plot, n_for_ngram, fill_color)
ggsave(
  paste0(
    "Plots/", news_to_plot, "_bigram_count.png"
    ),
  plot = n_gram,
  height = 6,
  width = 8,
  dpi = 300
  )

# plot trigram
n_for_ngram <- 3
fill_color <- c[i]
n_gram <- plot_ngram(news_df, news_to_plot, n_for_ngram, fill_color)

# Save trigram
ggsave(paste0("Plots/", news_to_plot, "_trigram_count.png"), plot = n_gram, height = 6

# plot wordcloud
word_freq <- news_df %>%
  filter(news_source == news_to_plot) %>%
  mutate(headline = gsub("[^[:alnum:] ]", "", headline)) %>%
  unnest_tokens(word, headline) %>%
  filter(!is.na(word)) %>%
  count(word) %>%
  top_n(n = 50, wt = n)

wc <- wordcloud2(data = word_freq, size = 1, color = "random-dark", backgroundColor =
saveWidget(wc, file = "Plots/wordcloud.html")
webshot(
  "Plots/wordcloud.html",
  file = paste0(
    "Plots/",
    news_to_plot,
    "_headline_wordcloud.png"
  ),
  vwidth = 1200,
  vheight = 800
)
Sys.sleep(3)

file.remove("Plots/wordcloud.html")
```

```
  i = i + 1

}
```

```
## Iteration Number : 1

## Iteration Number : 2

## Iteration Number : 3

## Iteration Number : 4

## Iteration Number : 5

## Iteration Number : 6

## Iteration Number : 7

## Iteration Number : 8

## Iteration Number : 9
```

5 Topic Modelling

We changed the value starting from k=5 all the way up k=12 where we observed that new topics can be added however after k=12 the topics were not clustering properly so we decided to keep k=12

```
# Get the corpus
corpus <- Corpus(VectorSource(news_df$description))
corpus <- tm_map(corpus, stemDocument)
```

```
## Warning in tm_map.SimpleCorpus(corpus, stemDocument): transformation drops
## documents
```

```
# Create a document term matrix with term frequency weighting
dtm <- DocumentTermMatrix(corpus, control = list(weighting = weightTf))
```

```
## Warning in TermDocumentMatrix.SimpleCorpus(x, control): custom functions are
## ignored
```

```
# Get TFIDF
tfidf <- weightTfIdf(dtm)

# Run LDA on the document term matrix
k <- 12 # number of topics
lda <- LDA(dtm, k = k, method = "Gibbs", control = list(seed = 1234))
#tidy(lda)

# Find the top 15 words associated with each topic
top_words_k3 <- terms(lda, 15)

#as.data.frame(top_words_k3)
print(top_words_k3)
```

```
##       Topic 1    Topic 2   Topic 3    Topic 4  Topic 5   Topic 6
##  [1,] "secur"    "char"    "will"     "year"   "compani" "world"
##  [2,] "nation"   "minist"  "said"     "sport"  "year"    "one"
##  [3,] "ukrain"   "day"     "announc"  "game"   "report"  "new"
##  [4,] "offici"   "first"   "china"    "next"   "busi"    "year"
##  [5,] "russia"   "countri" "nation"   "team"   "million" "set"
##  [6,] "war"      "govern"  "unit"     "major"  "bank"    "use"
##  [7,] "said"     "parti"   "presid"   "leagu"  "accord"  "open"
##  [8,] "group"    "leader"  "govern"   "will"   "month"   "across"
```

```
##  [9,] "russian"  "prime"   "monday"   "season" "increas" "australia"
## [10,] "social"   "forc"    "group"    "play"   "last"    "india"
## [11,] "militari" "sinc"    "union"    "three"  "tax"     "ago"
## [12,] "includ"   "fight"   "thursday" "final"  "financi" "last"
## [13,] "latest"   "sudan"   "trade"    "time"   "number"  "place"
## [14,] "defens"   "continu" "econom"   "last"   "cut"     "just"
## [15,] "media"    "capit"   "repres"   "first"  "price"   "becom"
##       Topic 7      Topic 8   Topic 9  Topic 10  Topic 11     Topic 12
##  [1,] "state"      "citi"    "like"   "new"     "char"       "comment"
##  [2,] "school"     "said"    "can"    "york"    "former"     "share"
##  [3,] "hous"       "home"    "mani"   "work"    "presid"     "stori"
##  [4,] "bill"       "peopl"   "make"   "show"    "court"      "articl"
##  [5,] "educ"       "two"     "say"    "time"    "polit"      "gift"
##  [6,] "republican" "polic"   "way"    "give"    "trump"      "news"
##  [7,] "right"      "offic"   "one"    "televis" "feder"      "get"
##  [8,] "univers"    "charg"   "peopl"  "april"   "washington" "plan"
##  [9,] "say"        "kill"    "chang"  "book"    "biden"      "fox"
## [10,] "law"        "yearold" "time"   "live"    "donald"     "listen"
## [11,] "support"    "arrest"  "around" "star"    "campaign"   "elect"
## [12,] "health"     "man"     "see"    "look"    "hous"       "vote"
## [13,] "student"    "author"  "now"    "imag"    "investig"   "min"
## [14,] "public"     "death"   "good"   "last"    "editor"     "sign"
## [15,] "governor"   "includ"  "want"   "age"     "alleg"      "experi"
```

```
# get top 10 words for each topic
top_words_k3 <- terms(lda, 10)

print(top_words_k3)
```

```
##       Topic 1   Topic 2   Topic 3   Topic 4  Topic 5   Topic 6     Topic 7
##  [1,] "secur"   "char"    "will"    "year"   "compani" "world"     "state"
##  [2,] "nation"  "minist"  "said"    "sport"  "year"    "one"       "school"
##  [3,] "ukrain"  "day"     "announc" "game"   "report"  "new"       "hous"
##  [4,] "offici"  "first"   "china"   "next"   "busi"    "year"      "bill"
##  [5,] "russia"  "countri" "nation"  "team"   "million" "set"       "educ"
##  [6,] "war"     "govern"  "unit"    "major"  "bank"    "use"       "republican"
##  [7,] "said"    "parti"   "presid"  "leagu"  "accord"  "open"      "right"
##  [8,] "group"   "leader"  "govern"  "will"   "month"   "across"    "univers"
##  [9,] "russian" "prime"   "monday"  "season" "increas" "australia" "say"
## [10,] "social"  "forc"    "group"   "play"   "last"    "india"     "law"
##       Topic 8   Topic 9 Topic 10  Topic 11     Topic 12
##  [1,] "citi"    "like"  "new"     "char"       "comment"
##  [2,] "said"    "can"   "york"    "former"     "share"
##  [3,] "home"    "mani"  "work"    "presid"     "stori"
```

```
##  [4,] "peopl"   "make"  "show"    "court"      "articl"
##  [5,] "two"     "say"   "time"    "polit"      "gift"
##  [6,] "polic"   "way"   "give"    "trump"      "news"
##  [7,] "offic"   "one"   "televis" "feder"      "get"
##  [8,] "charg"   "peopl" "april"   "washington" "plan"
##  [9,] "kill"    "chang" "book"    "biden"      "fox"
## [10,] "yearold" "time"  "live"    "donald"     "listen"
```

1 - International Conflicts 2 - Government and political 3 - Economy news 4 - Sports news 5 - Corporate news 6 - World news 7 - Education news 8 - Local news 9 - Lifestyle news 10 - Entertainment news 11 - Legal news 12 - Miscellaneous news

5.1 Get the topic on the data & remap the topic number

```
# get the topics and their terms
topics <- topics(lda)
terms <- terms(lda)

# assign topics to documents
doc_topics <- as.data.frame(lda@gamma)
doc_topics$topic <- apply(doc_topics, 1, which.max)

# add the topics to the news_df
news_df$topic <- doc_topics$topic

news_df %>% group_by(category, topic) %>% summarise(n = n(), .groups = 'drop')
```

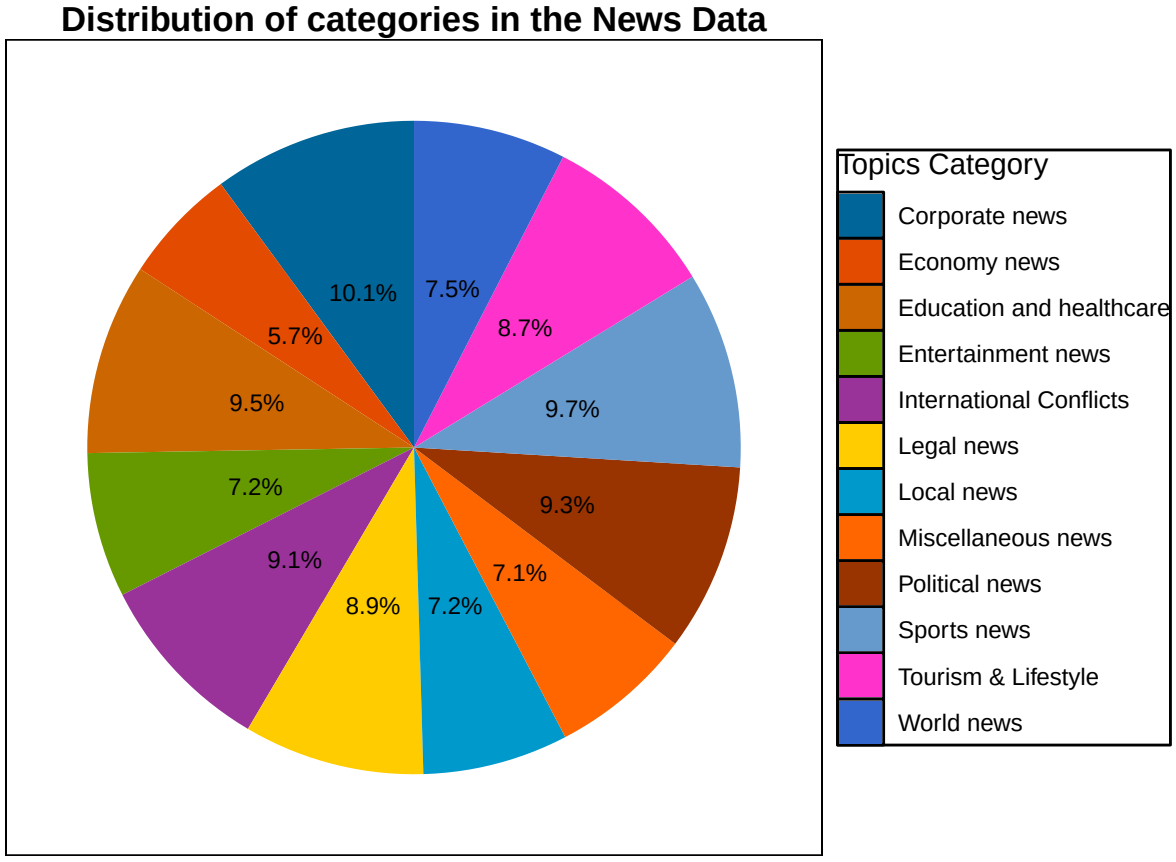
```
## # A tibble: 192 x 3
##    category topic     n
##    <chr>    <int> <int>
##  1 business     1    34
##  2 business     2    28
##  3 business     3    51
##  4 business     4    40
##  5 business     5   141
##  6 business     6    46
##  7 business     7    33
##  8 business     8    28
##  9 business     9    56
## 10 business    10    41
## # ... with 182 more rows
```

```r
# Map the topics
news_df <- news_df %>%
  mutate(topic_category = case_when(
    topic == 1 ~ "International Conflicts",
    topic == 2 ~ "Political news",
    topic == 3 ~ "Economy news",
    topic == 4 ~ "Sports news",
    topic == 5 ~ "Corporate news",
    topic == 6 ~ "World news",
    topic == 7 ~ "Education and healthcare",
    topic == 8 ~ "Local news",
    topic == 9 ~ "Tourism & Lifestyle",
    topic == 10 ~ "Entertainment news",
    topic == 11 ~ "Legal news",
    topic == 12 ~ "Miscellaneous news",
    TRUE ~ NA_character_
  ))
```

5.2 Distribution of News category in the Data

```r
# Define custom color palette
news_colors <- c(
  "#006699",
  "#E34B00",
  "#CC6600",
  "#669900",
  "#993399",
  "#FFCC00",
  "#0099CC",
  "#FF6600",
  "#993300",
  "#6699CC",
  "#FF33CC",
  "#3366CC"
)

news_cat_pie <- news_df %>%
            count(topic_category) %>%
            mutate(perc = n / sum(n)) %>%
            ggplot(aes(
              x = "",
              y = n,
```

```
              fill = topic_category,
              label = scales::percent(perc, accuracy = 0.1)
              )
            ) +
            geom_bar(stat = "identity", width = 1) +
            coord_polar("y", start = 0) +
            ggtitle("Distribution of categories in the News Data") +
            scale_fill_manual(values = news_colors) +
            labels(fill = "Topics Category", label = NULL) +
            theme_void() +
            theme(plot.title = element_text(hjust = 0.5, face = "bold")) +
            theme(rect = element_rect(fill = "transparent")) +
            geom_text(position = position_stack(vjust = 0.5), size = 3)

print(news_cat_pie)
```



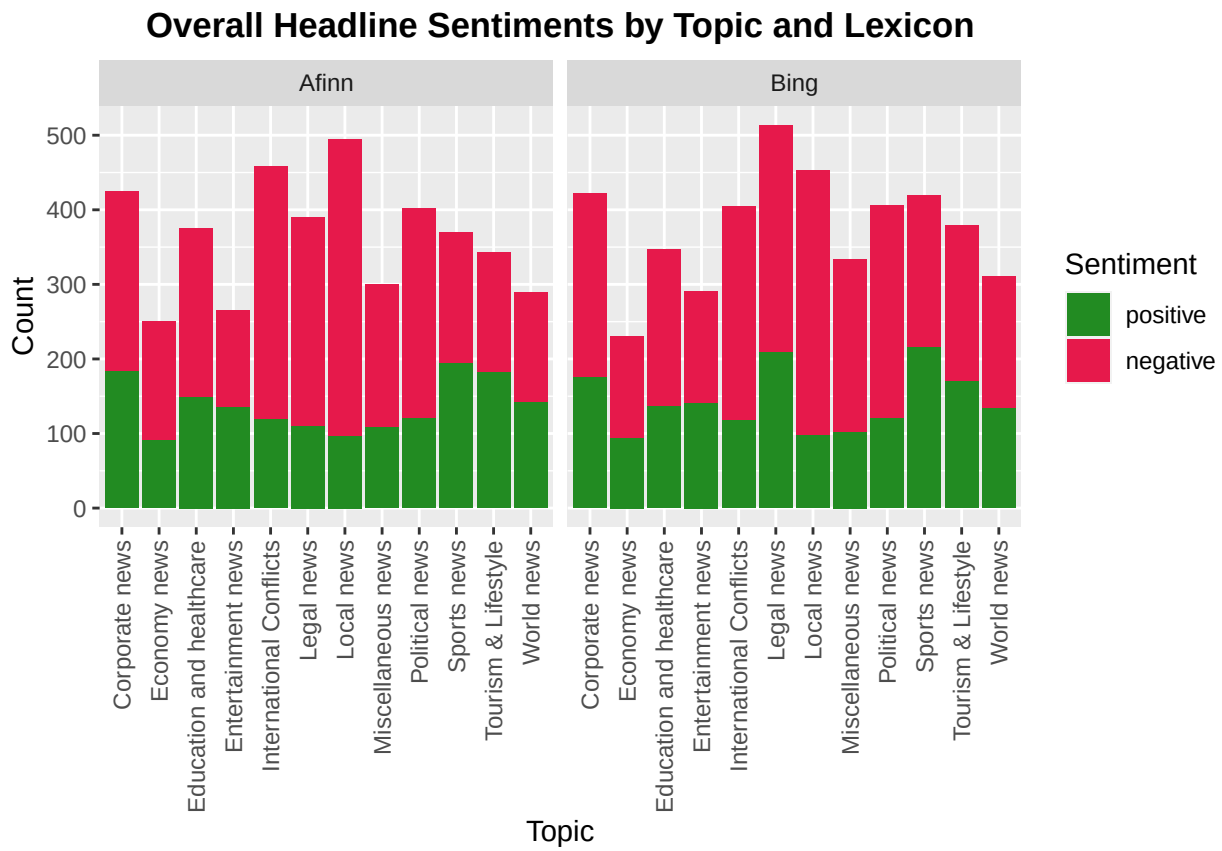

```
# save the plot with a larger size
ggsave(
  "Plots/news_category_afterLDA_count.png",
  plot = news_cat_pie,
  height = 6,
```

```
  width = 8,
  dpi = 300,
  bg = "transparent"
)
```

6 Sentiment Analysis

6.1 By Topic Category

```
df <- news_df %>%
     select(topic_category, headline)

# Perform sentiment analysis with Afinn lexicon
afinn_sentiments <- df %>%
  unnest_tokens(word, headline) %>%
  inner_join(get_sentiments("afinn"), by = "word") %>%
  group_by(topic_category, value) %>%
  summarize(count = n(), .groups = "drop") %>%
  mutate(sentiment = ifelse(value > 0, "positive", ifelse(value==0, "neutral", "negative


# Perform sentiment analysis with Bing lexicon
bing_sentiments <- df %>%
  unnest_tokens(word, headline) %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  group_by(topic_category, sentiment) %>%
  summarize(count = n(), .groups = "drop") %>%
  mutate(lexicon = "Bing")

# Combine the results from both lexicons
sentiments <- bind_rows(afinn_sentiments, bing_sentiments)

# Plot the sentiments
sentiment_split <-ggplot(sentiments, aes(x = topic_category, y = count, fill = sentiment
                geom_col(position = "stack") +
                facet_wrap(~ lexicon, ncol = 2) +
                labs(title = "Overall Headline Sentiments by Topic and Lexicon",
                     x = "Topic",
                     y = "Count",
                     fill = "Sentiment") +
                scale_fill_manual(values = c("positive" = "forestgreen", "negative" = "#
                theme(plot.title = element_text(hjust = 0.5, face = "bold")) +
                theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))

print(sentiment_split)
```

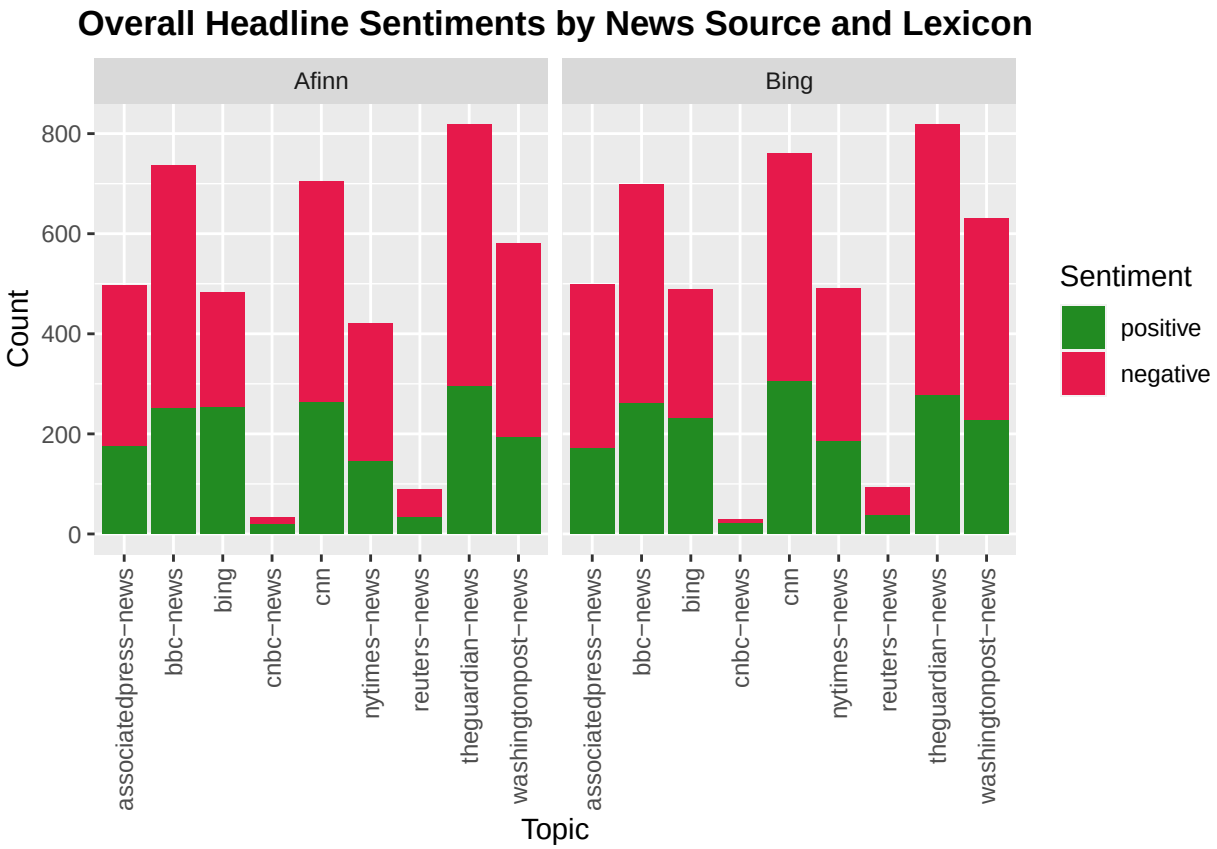



```
# save the plot with a larger size
ggsave(
  "Plots/sentiment_bytopic_count.png",
  plot = sentiment_split,
  height = 6,
  width = 8,
  dpi = 300
)
```

6.2 By Source

```
df <- news_df %>%
     select(news_source, headline)

# Perform sentiment analysis with Afinn lexicon
afinn_sentiments <- df %>%
  unnest_tokens(word, headline) %>%
  inner_join(get_sentiments("afinn"), by = "word") %>%
  group_by(news_source, value) %>%
```

```r
  summarize(count = n(), .groups = "drop") %>%
  mutate(sentiment = ifelse(value > 0, "positive", ifelse(value==0, "neutral", "negative


# Perform sentiment analysis with Bing lexicon
bing_sentiments <- df %>%
  unnest_tokens(word, headline) %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  group_by(news_source, sentiment) %>%
  summarize(count = n(), .groups = "drop") %>%
  mutate(lexicon = "Bing")

# Combine the results from both lexicons
sentiments <- bind_rows(afinn_sentiments, bing_sentiments)

# Plot the sentiments
sentiment_split <-ggplot(sentiments, aes(x = news_source, y = count, fill = sentiment))
                geom_col(position = "stack") +
                facet_wrap(~ lexicon, ncol = 2) +
                labs(title = "Overall Headline Sentiments by News Source and Lexicon",
                     x = "Topic",
                     y = "Count",
                     fill = "Sentiment") +
                scale_fill_manual(values = c("positive" = "forestgreen", "negative" = "#
                theme(plot.title = element_text(hjust = 0.5, face = "bold")) +
                theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))

print(sentiment_split)
```




```
# save the plot with a larger size
ggsave(
  "Plots/sentiment_bysource_count.png",
  plot = sentiment_split,
  height = 6,
  width = 8,
  dpi = 300,
  bg = "transparent"
)
```