Text Analytics

Individual Assignment

Business Insight Report on Popular Newspapers:

- Wall Street Journal
- Times of India
- Hindustan Times
- The Indian Express

Sentiment Analysis of Top 20 Trending Articles on Indian National News

12 FEBRUARY 2020

Hult International Business School
Authored by: Arundhishaan Kanagaraja
Cohort: MSBA5 - Valencia

Introduction

English-language dailies and journals remain highly influential in India. The vernacular press is increasing steadily in absolute and relative importance. Overall the press functions with little government censorship, serious controls in matters of national security, in times of emergency or to avoid inflaming passions that lead to communal riots or comparable disturbances (Britannica, 2019).

"The largest-circulating dailies are The Times of India and Hindustan Times (both in English), the Hindustan and the Navbharat Times (Hindi), and the Anandabazar Patrika (Bengali)."

This proves a great opportunity of unstructured text data that is available for data analysis. This report focuses on the Sentiment Analysis of Textual data from four popular Newspapers: Wall Street Journal (WSJ), Times of India (TOI), Hindustan Times and The Indian Express. The latter three are top English newspapers in India.

Framework 1

Tokenization and bi-grams are used to initially understand the most common news by the newspaper. Further, the sentiment analysis is done using the 'nrc' dictionary in R. We would visualize the results using bar charts and word clouds.

Framework 2

Term Frequency – Inverse Document Frequency (tf-idf) is used to find the unique words from articles of these Newspapers that gives us the insight of how these Newspapers are different in their focus to pick specific News.

Data Collection

Trending top 20 Online News articles are collected from the Indian National News section of each newspaper's website which is saved in pdf format. Thus, we have obtained our unstructured data.

Data Cleaning

The pdf documents of 20 articles are imported and structured into a dataframe. Unwanted text such as ads, author names, non-insightful repetitive text, etc. are eliminated by defining custom stopwords. The column name defines the newspaper company that the articles are taken from and the rows contain the text data from the articles.

Similar dataframes are created for the remaining three newspapers.

Text Tokenization & Bi-grams

The following are the tidy format of tokenized data of different newspapers:

- *Wall Street Journal*

```
# A tibble: 3,004 x 2
   word            n
   <chr>       <int>
 1 law           195
 2 citizenship   177
 3 india's       165
 4 government    148
 5 muslims       140
 6 muslim        125
 7 protests      123
 8 delhi         120
 9 police        117
10 hindu          95
# ... with 2,994 more rows
```

```
# A tibble: 3,004 x 3
   word1       word2           n
   <chr>       <chr>       <int>
 1 citizenship law            72
 2 prime       minister       54
 3 minister    narendra       42
 4 muslim      majority       42
 5 hindu       nationalist    33
 6 bharatiya   janata         30
 7 janata      party          28
 8 rights      reserved       28
 9 citizenship amendment      26
10 internet    services       22
# ... with 2,994 more rows
```

Insights: The most common news by Wall Street Journal follows information related to Law, actions by the Indian government, social protests and views of the people. From the bi-gram, we can further understand that there is news widely spoken about citizenship amendments, people rights and the actions of the ruling party.

➢ *Times of India*

```
# A tibble: 1,946 x 2
   word          n
   <chr>     <int>
 1 delhi       573
 2 election    326
 3 2020        193
 4 assembly    132
 5 congress    129
 6 result      106
 7 aap          99
 8 bjp          99
 9 party        96
10 polls        91
# ... with 1,936 more rows
```

```
# A tibble: 1,788 x 3
   word1     word2          n
   <chr>     <chr>      <int>
 1 delhi     election     110
 2 election  result       104
 3 delhi     assembly     100
 4 assembly  election      59
 5 delhi     delhi         59
 6 delhi     polls         59
 7 2020      liveblog      58
 8 congress  candidates    52
 9 arvind    kejriwal      51
10 63        congress      49
# ... with 1,778 more rows
```

Insights: The most common news published by Times of India covers information about the 2020 Delhi elections. From the bi-gram, we can further understand that there is news widely spoken about the victory of Arvind Kejriwal as the Chief Minister of Delhi (Dutt, 2020).

➢ *Hindustan Times*

```
# A tibble: 1,896 x 2
   word          n
   <chr>     <int>
 1 court        46
 2 election     34
 3 board        33
 4 cricket      32
 5 exams        32
 6 school       32
 7 sports       32
 8 party        29
 9 people       28
10 chief        27
# ... with 1,886 more rows
```

```
# A tibble: 1,550 x 3
   word1      word2          n
   <chr>      <chr>      <int>
 1 board      exams         32
 2 omar's     detention     21
 3 chief      subhash       20
 4 cong       chief         20
 5 crushing   defeat        20
 6 party's    crushing      20
 7 1st        casualty      19
 8 defeat     psa           19
 9 hear       challenge     19
10 kejriwal's party         19
# ... with 1,540 more rows
```

Insights: The most common news by Hindustan Times follows information related to Politics, Education and Sports. From the bi-gram, we can further

understand that there is news widely spoken about higher secondary Board Examinations, elections and specific political leaders.

➢ *The Indian Express*

```
# A tibble: 1,692 x 2
   word           n
   <chr>      <int>
 1 delhi         34
 2 police        31
 3 government    28
 4 court         20
 5 minister      18
 6 rs            18
 7 women         18
 8 bjp           16
 9 party         16
10 people        16
# ... with 1,682 more rows
```

```
# A tibble: 1,327 x 3
   word1      word2          n
   <chr>      <chr>      <int>
 1 honey      mahajan        8
 2 rear       admiral        8
 3 worth      rs             8
 4 assembly   polls          7
 5 chief      minister       6
 6 farm       house          6
 7 government hospitals      6
 8 mobile     phones         6
 9 sanjay     vatsayan       6
10 admiral    sanjay         5
# ... with 1,317 more rows
```

Insights: The most common news by The Indian Express follows information related to Politics and Law. From the bi-gram, we can further understand that there is also news widely spoken about Military Personnel ('Admiral' – Commander of Naval squadron), Technology and Commerce.

Sentiment Analysis & Word Clouds

The following are the Sentiment Analysis in tidy format, bar chart & wordcloud (using nrc dictionary):

➢ *Wall Street Journal*

```
# A tibble: 344 x 3
   word          sentiment     n
   <chr>         <chr>     <int>
 1 protests      negative    123
 2 critics       negative     46
 3 attack        negative     37
 4 protest       negative     35
 5 support       positive     22
 6 opposition    negative     20
 7 illegal       negative     16
 8 supreme       positive     16
 9 controversial negative     14
10 led           positive     14
# ... with 334 more rows
```

- **bing dictionary**

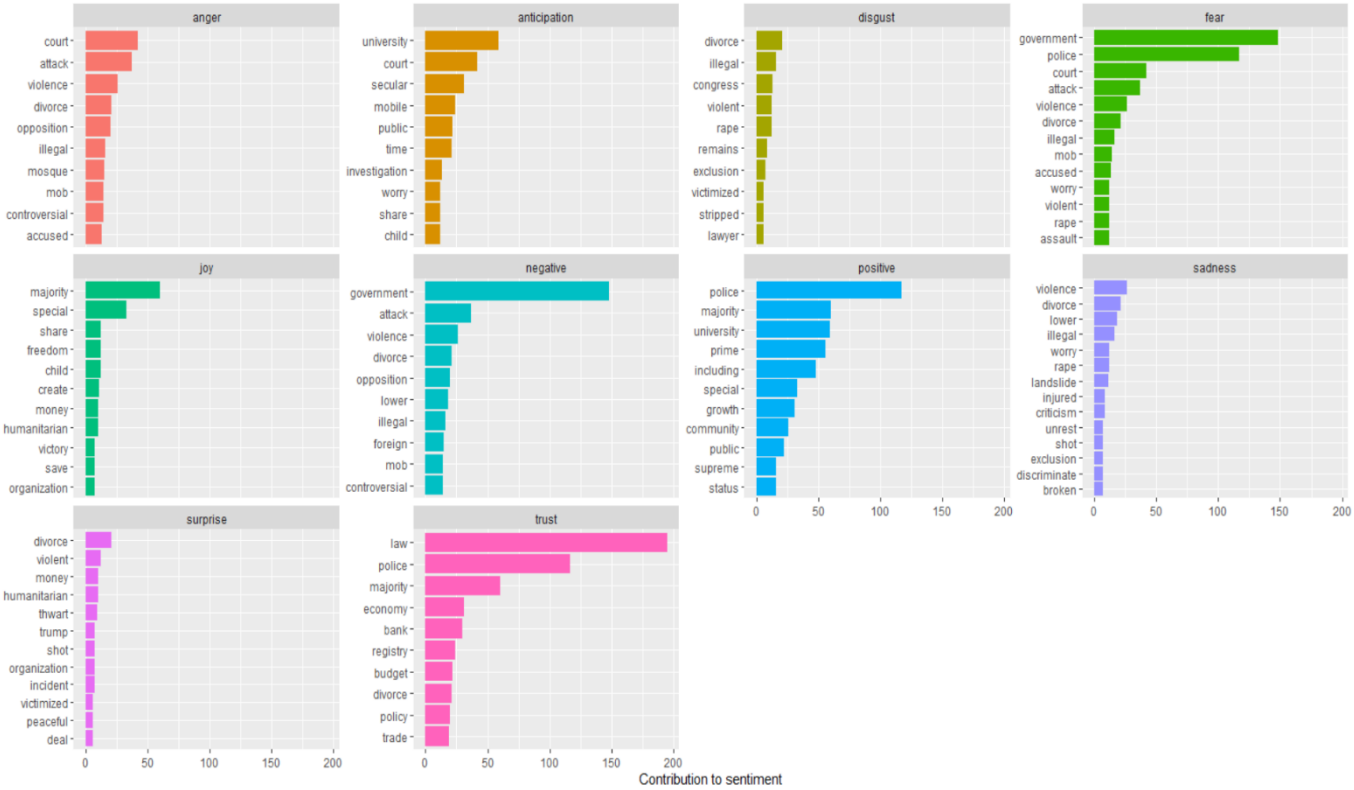
```
# A tibble: 333 x 2
   word          n
   <chr>     <int>
 1 protests    123
 2 critics      46
 3 attack       37
 4 protest      35
 5 growth       31
 6 violence     26
 7 protesters   24
 8 arrested     23
 9 banned       22
10 support      22
# ... with 323 more rows
```

- **afinn dictionary**

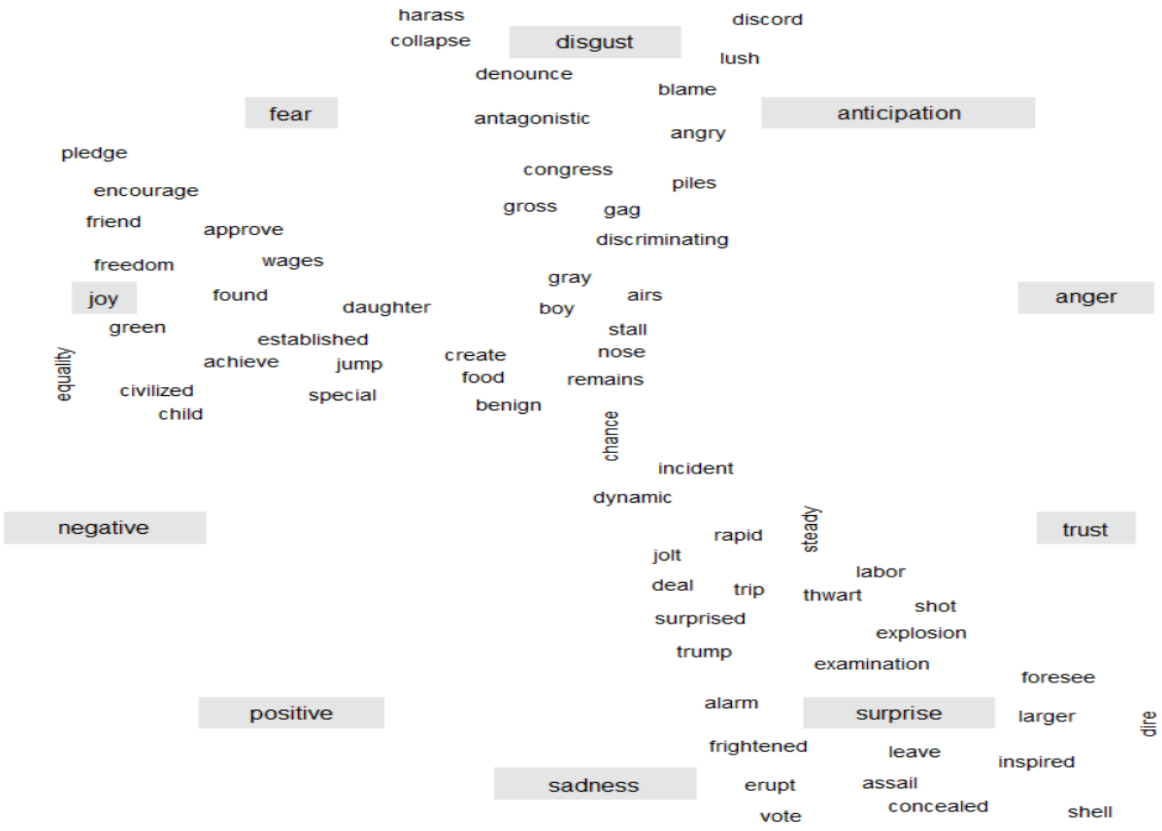
```
# A tibble: 1,409 x 3
   word       sentiment      n
   <chr>      <chr>      <int>
 1 law        trust        195
 2 government fear         148
 3 government negative     148
 4 police     fear         117
 5 police     positive     117
 6 police     trust        117
 7 majority   joy           60
 8 majority   positive      60
 9 majority   trust         60
10 university anticipation  59
# ... with 1,399 more rows
```

- **nrc dictionary**

Bar Chart (Using nrc dictionary showing words' contribution to sentiment):



Word Cloud (Using nrc dictionary):



Insights: From the above charts, we can understand that Wall Street Journal articles have a strong use of words that relate to the sentiments, surprise and joy. When they publish political News, they also focus more on the views of the people which can be seen through the usage of the word 'protest' and other words used closer to the 'disgust' sentiment in the word cloud.

➢ *Times of India*

```
# A tibble: 170 x 3
   word       sentiment     n
   <chr>      <chr>     <int>
 1 lose       negative     71
 2 trump      positive     65
 3 victory    positive     63
 4 fans       positive     60
 5 tortured   negative     60
 6 lost       negative     50
 7 top        positive     48
 8 breaking   negative     40
 9 excitement positive     40
10 losing     negative     36
# ... with 160 more rows
```

- **bing dictionary**

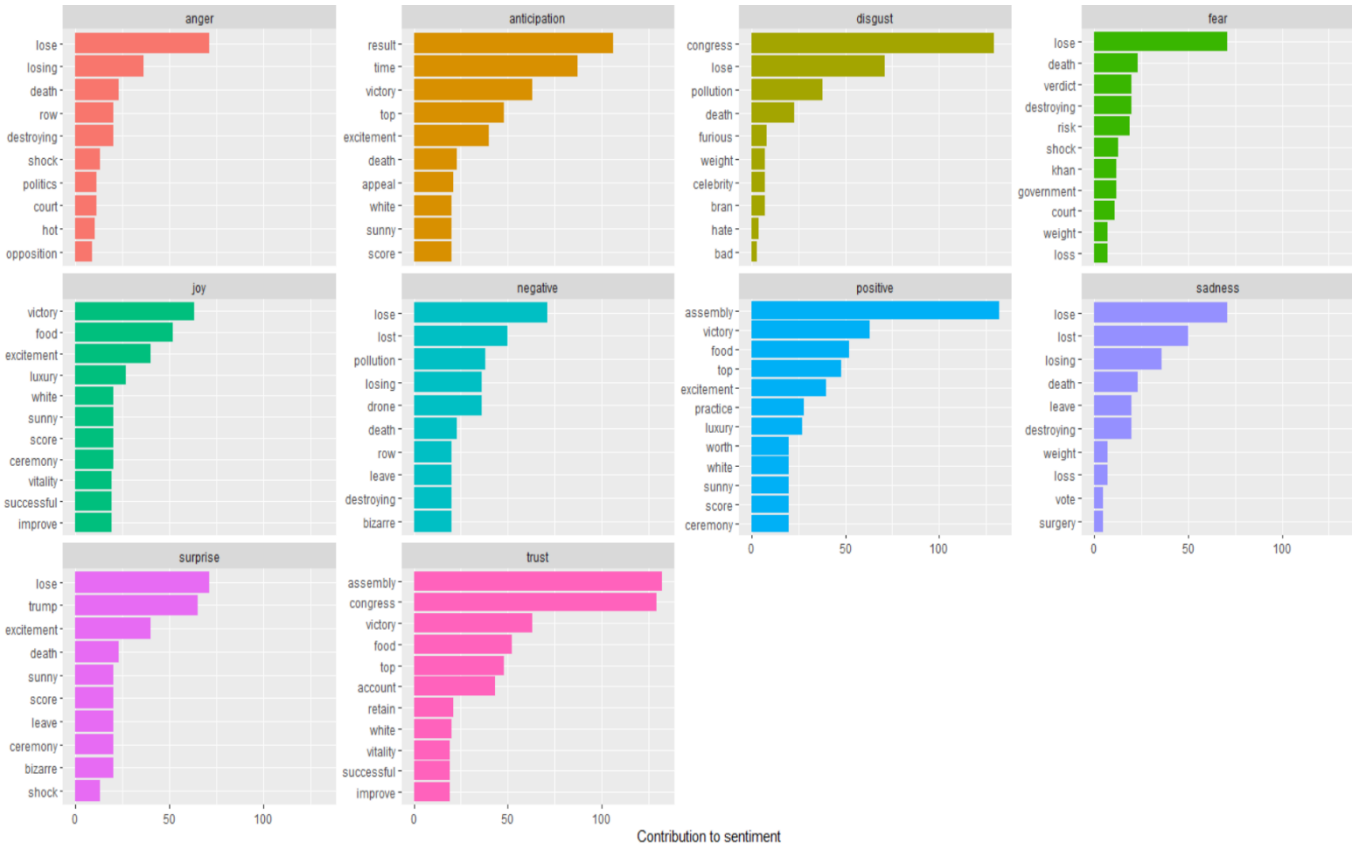
```
# A tibble: 150 x 2
   word          n
   <chr>     <int>
 1 stop         63
 2 tortured     60
 3 lost         50
 4 top          48
 5 excitement   40
 6 shares       40
 7 losing       36
 8 treasures    30
 9 free         29
10 popular      26
# ... with 140 more rows
```

- **afinn dictionary**

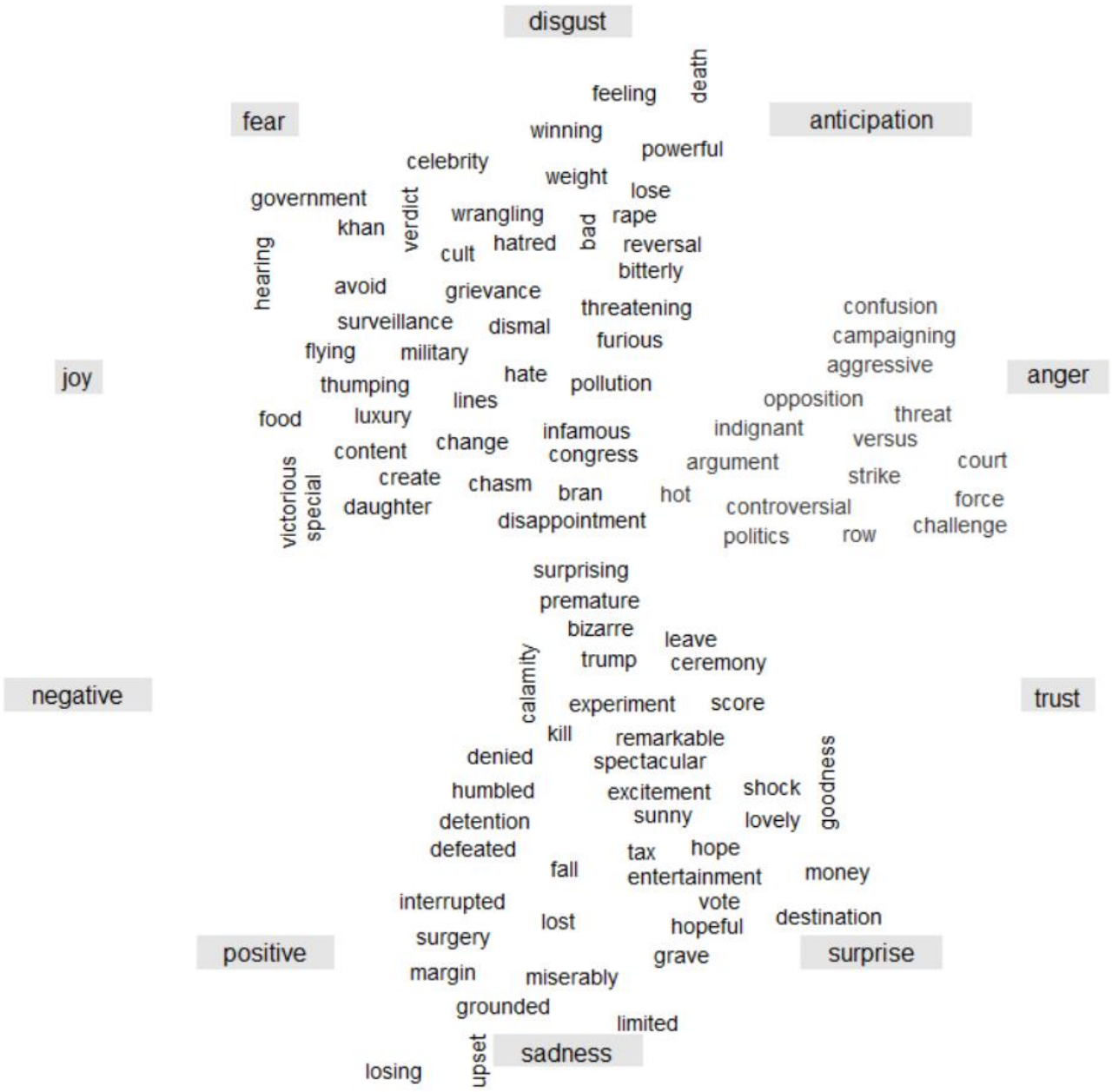
```
# A tibble: 662 x 3
   word     sentiment        n
   <chr>    <chr>        <int>
 1 assembly positive       132
 2 assembly trust          132
 3 congress disgust        129
 4 congress trust          129
 5 result   anticipation   106
 6 time     anticipation    87
 7 lose     anger           71
 8 lose     disgust         71
 9 lose     fear            71
10 lose     negative        71
# ... with 652 more rows
```

- **nrc dictionary**

Bar Chart (Using nrc dictionary showing words' contribution to sentiment):



Word Cloud (Using nrc dictionary):



Insights: From the above charts, we can understand that Times of India articles have a strong use of words that relate to the sentiments, fear, disgust, sadness and anger. The word 'lose' has been used to drive different sentiments which is related to covering the Delhi Elections. The words 'assembly' and 'congress' also indicate that they are publishing many stories about political parties actively.

➢ *Hindustan Times*

```
# A tibble: 189 x 3
   word      sentiment     n
   <chr>     <chr>     <int>
 1 defeat    positive     24
 2 crushing  negative     20
 3 victory   positive     20
 4 wins      positive     20
 5 casualty  negative     19
 6 limited   negative     19
 7 promises  positive     19
 8 fall      negative     18
 9 shine     positive     16
10 rape      negative     11
# ... with 179 more rows
```

- **bing dictionary**

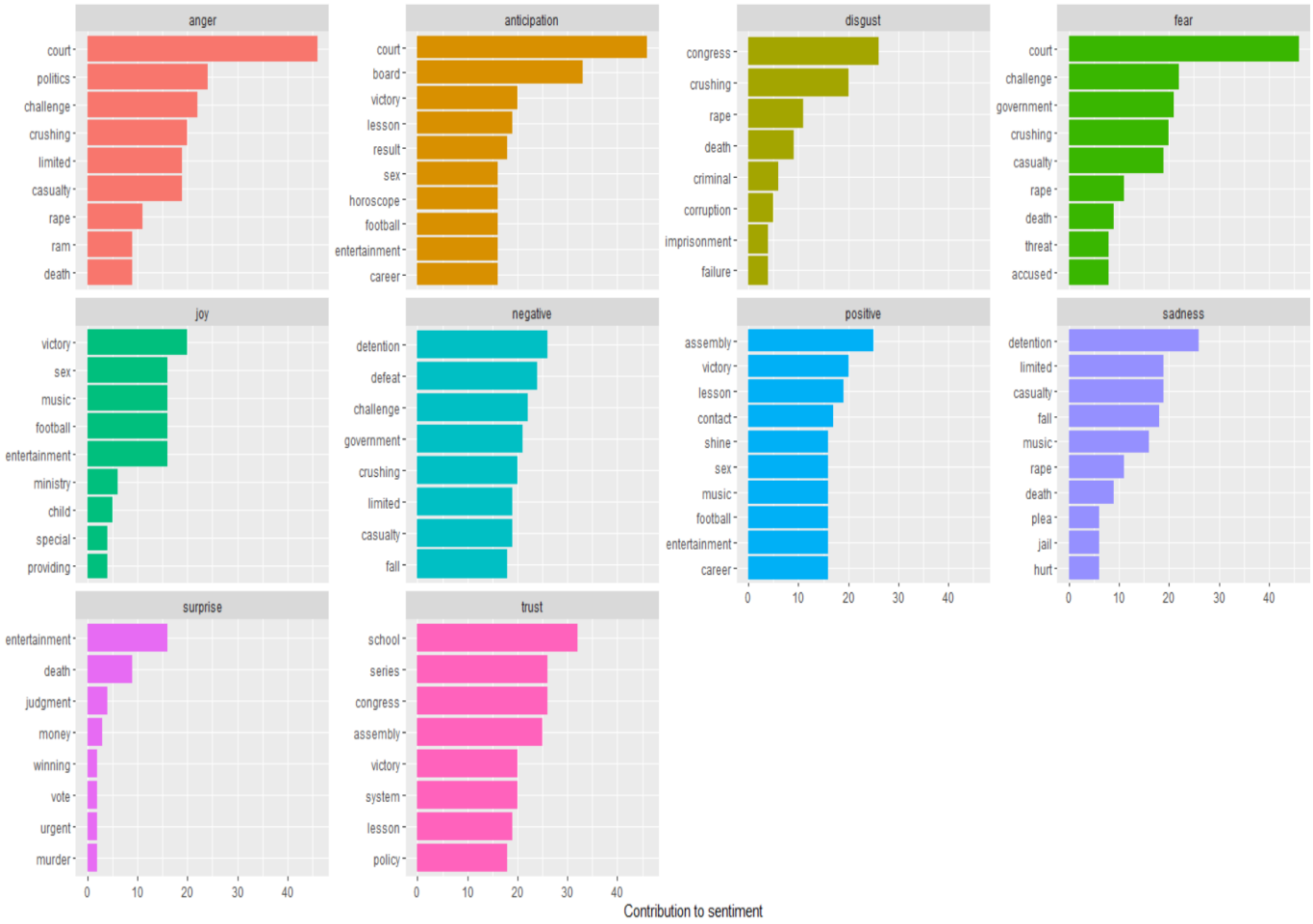
```
# A tibble: 170 x 2
   word          n
   <chr>     <int>
 1 detention    26
 2 challenge    22
 3 crushing     20
 4 wins         20
 5 casualty     19
 6 limited      19
 7 promises     19
 8 rape         11
 9 death         9
10 accused       8
# ... with 160 more rows
```

- **afinn dictionary**

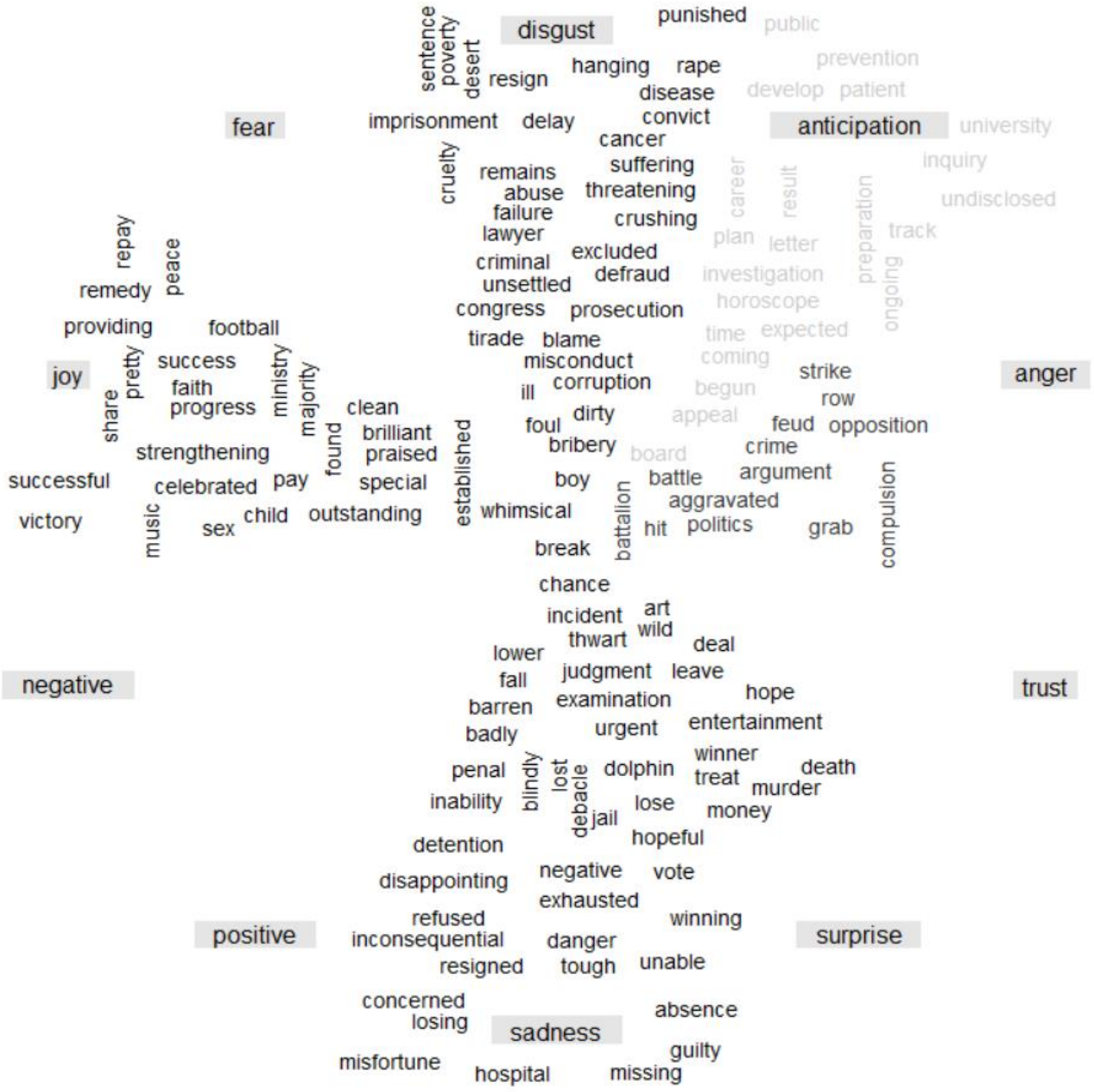
```
# A tibble: 834 x 3
   word      sentiment       n
   <chr>     <chr>       <int>
 1 court     anger          46
 2 court     anticipation   46
 3 court     fear           46
 4 board     anticipation   33
 5 school    trust          32
 6 congress  disgust        26
 7 congress  trust          26
 8 detention negative       26
 9 detention sadness        26
10 series    trust          26
# ... with 824 more rows
```

- **nrc dictionary**

Bar Chart (Using nrc dictionary showing words' contribution to sentiment):



Word Cloud (Using nrc dictionary):



Insights: From the above charts, we can understand that Hindustan Times articles have a strong use of words that relate to the sentiments, anticipation, sadness, disgust and joy. The word ‘court’ has been used to drive different sentiments like anger, anticipation and fear. This relates to News on Law cases where readers anticipate the story. Similar words like ‘criminal’, ‘punished’, ‘rape’, etc. support the same sentiments.

➢ *The Indian Express*

```
# A tibble: 161 x 3
   word       sentiment     n
   <chr>      <chr>     <int>
 1 attack     negative      8
 2 protest    negative      8
 3 worth      positive      8
 4 protesting negative      7
 5 appeal     positive      5
 6 rejected   negative      5
 7 forged     negative      4
 8 mercy      positive      4
 9 plea       negative      4
10 rape       negative      4
# ... with 151 more rows
```

- **bing dictionary**

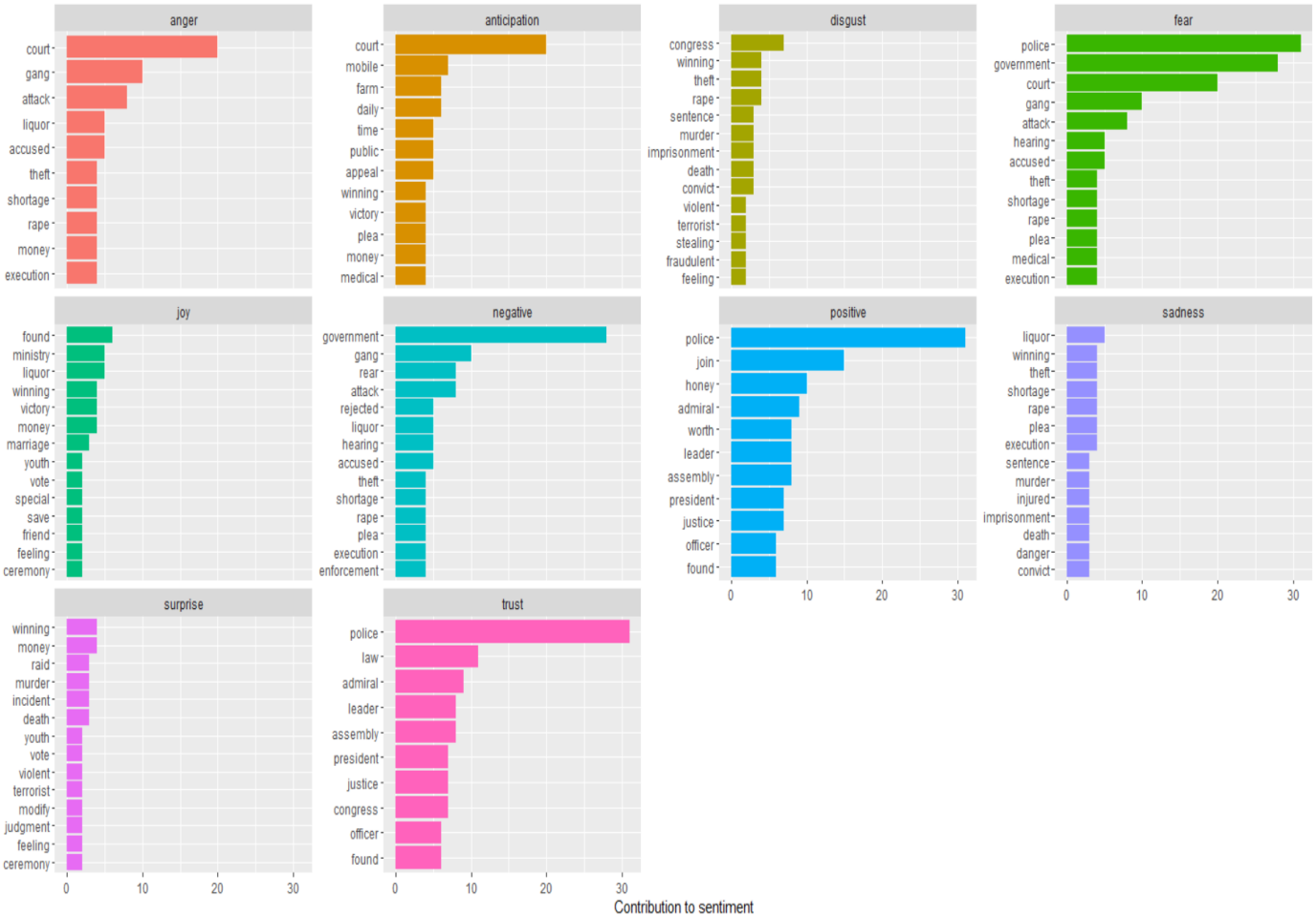
```
# A tibble: 133 x 2
   word          n
   <chr>     <int>
 1 join         15
 2 attack        8
 3 protest       8
 4 worth         8
 5 justice       7
 6 protesting    7
 7 accused       5
 8 rejected      5
 9 arrested      4
10 mercy         4
# ... with 123 more rows
```

- **afinn dictionary**

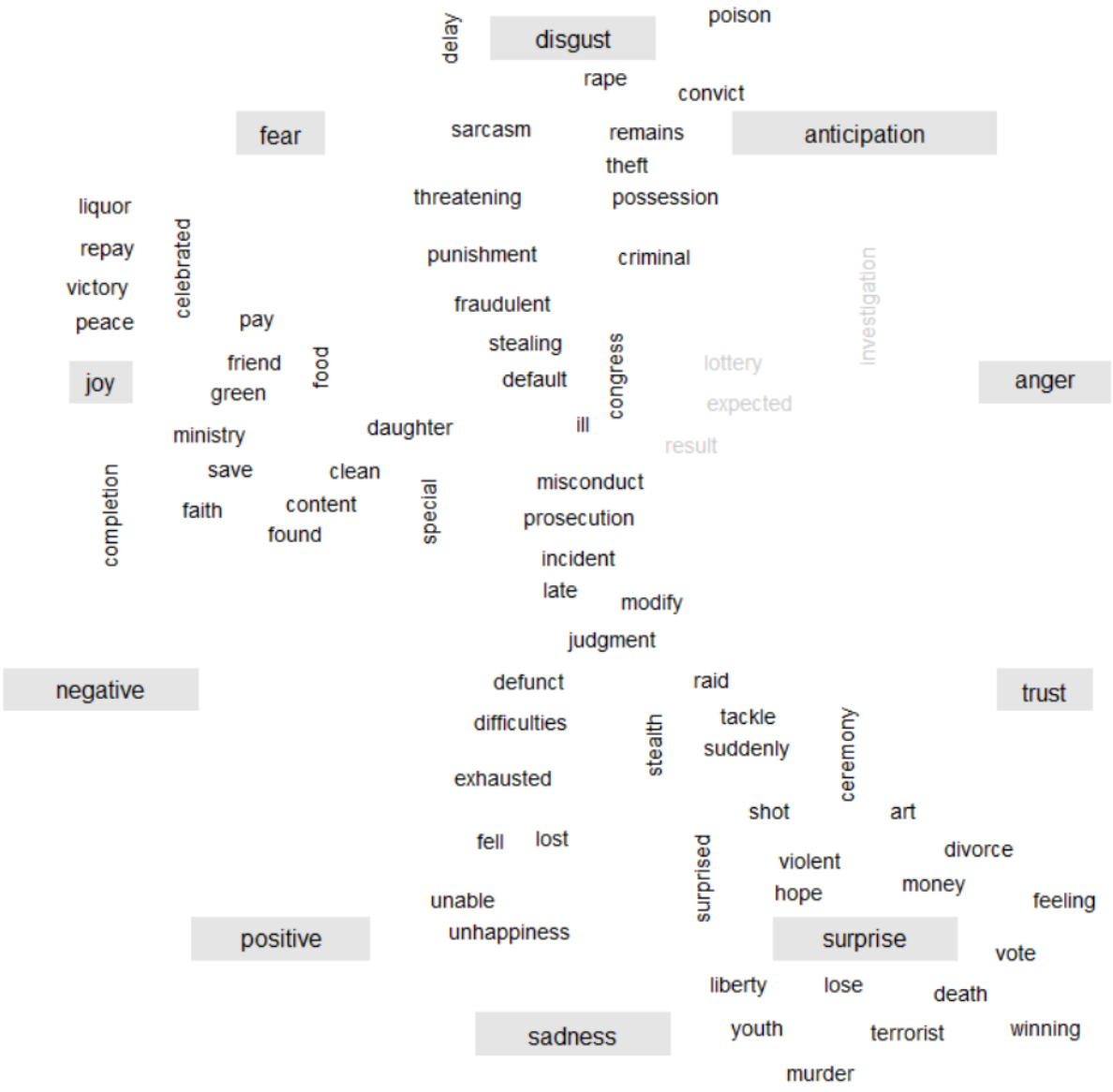
```
# A tibble: 706 x 3
   word       sentiment     n
   <chr>      <chr>     <int>
 1 police     fear         31
 2 police     positive     31
 3 police     trust        31
 4 government fear         28
 5 government negative     28
 6 court      anger        20
 7 court      anticipation 20
 8 court      fear         20
 9 join       positive     15
10 law        trust        11
# ... with 696 more rows
```

- **nrc dictionary**

Bar Chart (Using nrc dictionary showing words' contribution to sentiment):



Word Cloud (Using nrc dictionary):

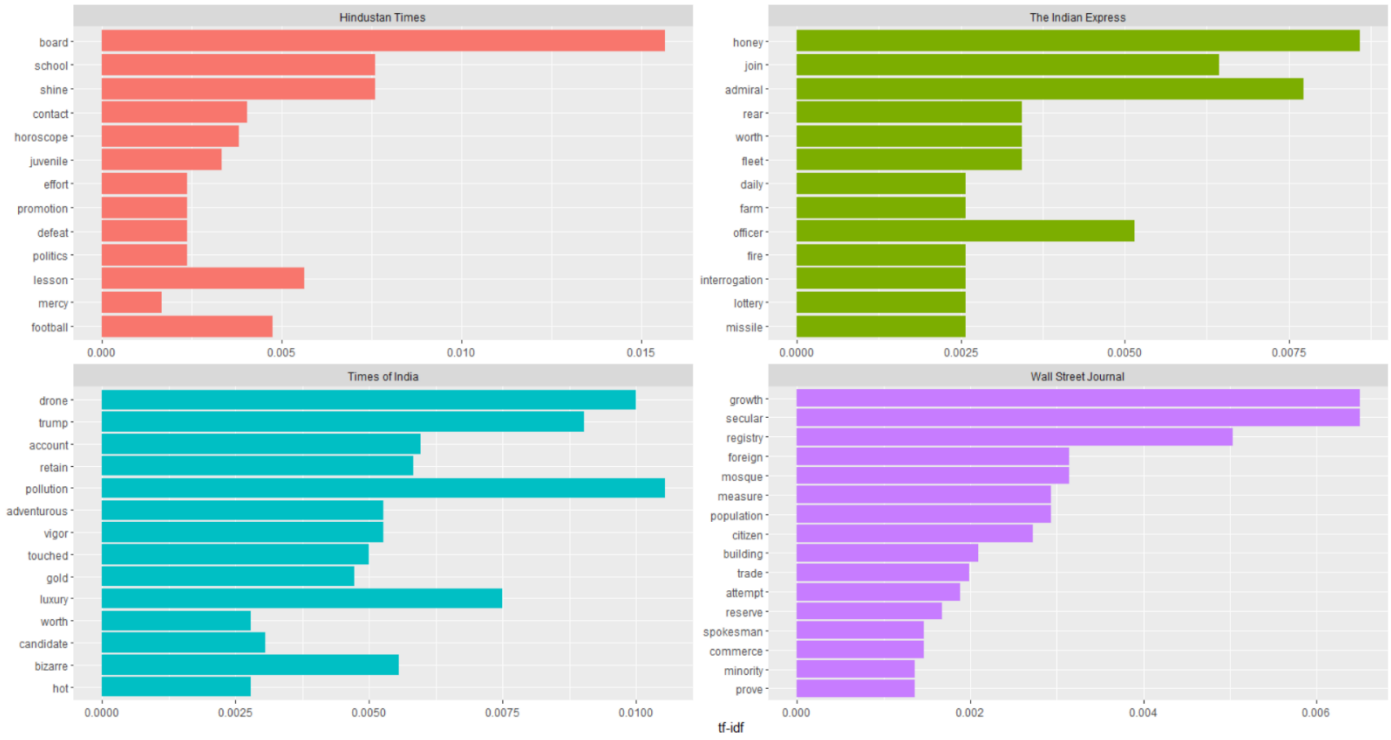


Insights: From the above charts, we can understand that The Indian Express articles have a strong use of words that relate to the sentiments, surprise, disgust and joy. The word ‘police’, ‘government’ and ‘court’ are widely used which contribute to different sentiments which involves News on several Law cases. The words ‘money’, and ‘winning’ contribute to the sentiment surprise, and joy indicating commerce or social News.

Finding Unique Trending News

The term frequency - inverse document frequency (tf-idf) is used to identify unique words from the articles of each newspaper. Though there are very common trending News, using tf-idf we can identify how newspapers differentiate themselves to their customers.

Bar graph using tf-idf:



Insights:

- Wall Street Journal focuses on a very broader scale of News comprising growth, people's rights, culture, trade and commerce.
- Times of India focuses on luxury, gold, technology and environmental News targeting youth, women and activists.
- Hindustan Times focuses on education and sports news targeting youth and entertainment sector.
- The Indian Express focuses more on specific Law cases, military news and other anticipatory news like lottery, social news, etc.

Conclusion

By using Text analytics techniques, unstructured trending News data on Indian National News of different Newspapers is transformed into structural data for data analysis.

The Sentiment analysis shows that different Newspapers express various sentiments using different style of writing to sell their News to their customers.

- Wall Street Journal sets a standard perspective to share quality information to students and professional class of people.
- Newspapers may focus on every aspect of a specific topic. For example, Times of India focuses on covering the Delhi Elections.
- Newspapers may also create awareness on pollution, new technology, education, etc.
- Newspapers may focus on articles that create surprise, joy and anticipation. For example, The Indian Express focuses on Law cases, social events, lottery, etc.

Finally, we found some unique news, sectors and customers that each Newspaper is targeting.

Reference

The Editors of Encyclopaedia Britannica. (2019, February 4). India. Retrieved from https://www.britannica.com/place/India/Cultural-institutions#ref487381

Dutt, B. (2020, February 11). Opinion | How this Delhi leader fought and defeated Modi's BJP - and why it matters. Retrieved from https://www.washingtonpost.com/opinions/2020/02/11/how-this-delhi-leader-fought-defeated-modis-bjp-why-it-matters/

Appendix

R Code

```
# Calling libraries

library(dplyr)

library(tidytext)

library(tidyr)

library(ggplot2)

library(pdftools)

library(wordcloud)

library(reshape2)

#####################################################################

Sentiment Analysis on Trending 20 Wall Street Journal Articles about Indian National News

#####################################################################

# Importing all PDF files from the same folder

setwd("C:/Users/arund/OneDrive/Desktop/Masters in Business Analytics/Text Analytics/Assignments/Individual/pdf_wsj")

nm <- list.files(path="C:/Users/arund/OneDrive/Desktop/Masters in Business Analytics/Text Analytics/Assignments/Individual/pdf_wsj")

my_pdf_text <- do.call(rbind, lapply(nm, function(x) pdf_text(x)))

my_pdf_text <- as.data.frame(my_pdf_text)


# Merge all columns

my_df <- unite(my_pdf_text, col = wsj, sep = " ")
```

Custom stopwords

```
custom <- data_frame(word = c("http", "rt", "https", "t.io", "india", "india's",
"articles", "copy","copies", "personal", "visit", "ready", "commercial",
"www.djreprints.com", “www.wsj.com", "vibhuti", “agarwal", "krishna", "pokharel",

"2020", "copyright", "dow", "jones", "colleagues", "clients"),
lexicon=rep("custom", each=25))

new_stopwords <- bind_rows(custom, stop_words)


# Tokenization

wsj_tokenized <- my_df %>%

  unnest_tokens(word, wsj) %>%

  anti_join(new_stopwords) %>% #here's where we remove tokens

  count(word, sort=TRUE)

print(wsj_tokenized) # This is Tidy Format
```

```
# A tibble: 3,004 x 2
   word            n
   <chr>       <int>
 1 law           195
 2 citizenship   177
 3 india’s       165
 4 government    148
 5 muslims       140
 6 muslim        125
 7 protests      123
 8 delhi         120
 9 police        117
10 hindu          95
# ... with 2,994 more rows
```

Bi-grams

```
my_bigrams <- my_df %>%

  unnest_tokens(bigram, wsj, token = "ngrams", n=2)
```

```
bigrams_separated <- my_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")
bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% new_stopwords$word) %>%
  filter(!word2 %in% new_stopwords$word)
bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)
print(bigram_counts)
```

```
# A tibble: 3,004 x 3
   word1       word2           n
   <chr>       <chr>       <int>
 1 citizenship law            72
 2 prime       minister       54
 3 minister    narendra       42
 4 muslim      majority       42
 5 hindu       nationalist    33
 6 bharatiya   janata         30
 7 janata      party          28
 8 rights      reserved       28
 9 citizenship amendment      26
10 internet    services       22
# ... with 2,994 more rows
```

```
# Get Sentiments
# Using bing dictionary
wsj_sentiments <- my_df %>%
  unnest_tokens(word, wsj) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()
```

print(wsj_sentiments)

```
# A tibble: 344 x 3
   word          sentiment     n
   <chr>         <chr>     <int>
 1 protests      negative    123
 2 critics       negative     46
 3 attack        negative     37
 4 protest       negative     35
 5 support       positive     22
 6 opposition    negative     20
 7 illegal       negative     16
 8 supreme       positive     16
 9 controversial negative     14
10 led           positive     14
# ... with 334 more rows
```

Using afinn dictionary

```
wsj_sentiments <- my_df %>%
  unnest_tokens(word, wsj) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("afinn")) %>%
  count(word, sort=T) %>%
  ungroup()
print(wsj_sentiments)
```

```
# A tibble: 333 x 2
   word           n
   <chr>      <int>
 1 protests     123
 2 critics       46
 3 attack        37
 4 protest       35
 5 growth        31
 6 violence      26
 7 protesters    24
 8 arrested      23
 9 banned        22
10 support       22
# ... with 323 more rows
```

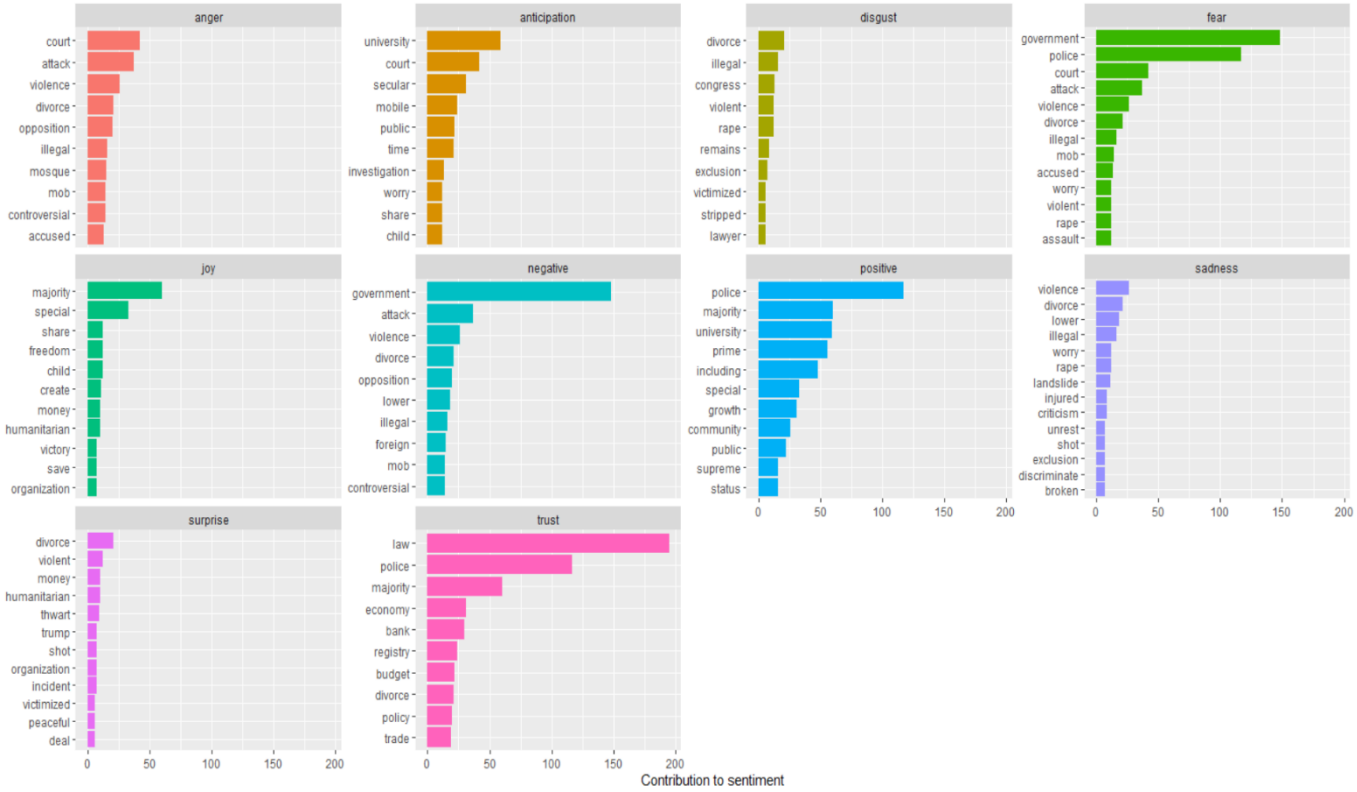
```
# Using nrc dictionary
wsj_sentiments <- my_df %>%
  unnest_tokens(word, wsj) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

print(wsj_sentiments)
```

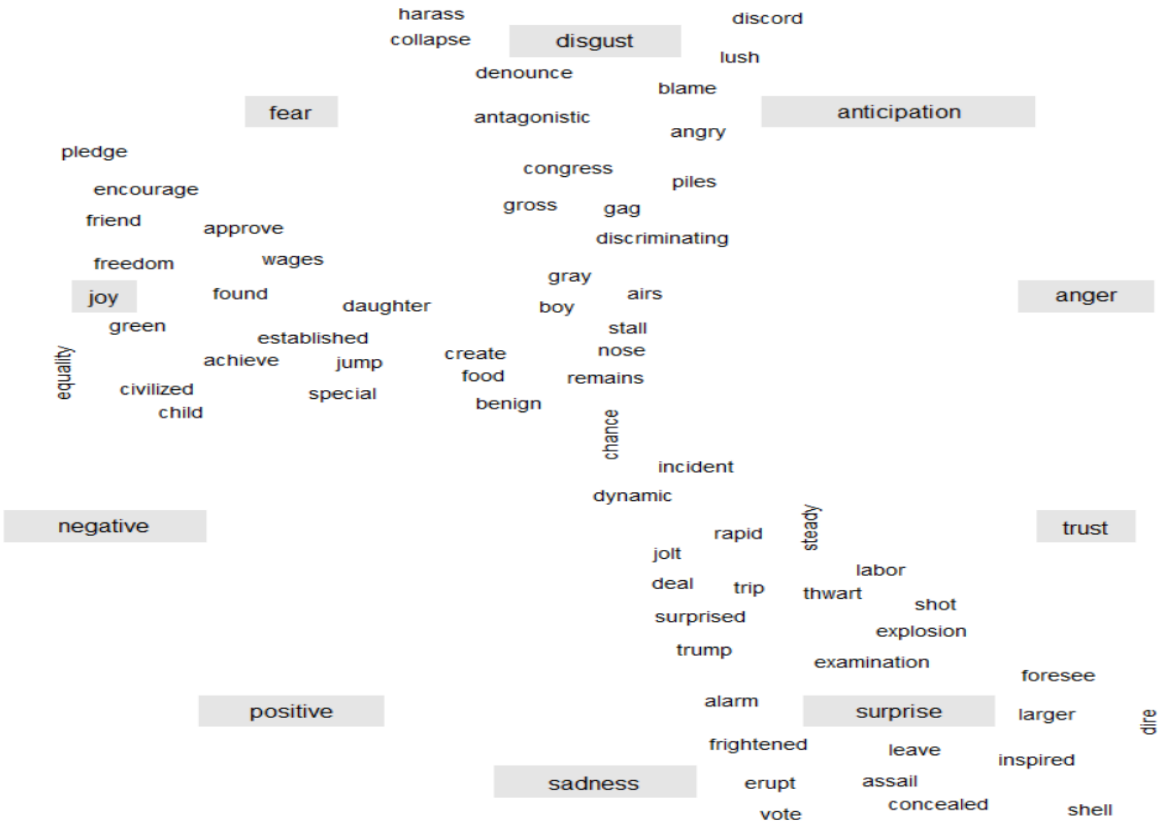
```
# A tibble: 1,409 x 3
   word       sentiment        n
   <chr>      <chr>        <int>
 1 law        trust          195
 2 government fear           148
 3 government negative       148
 4 police     fear           117
 5 police     positive       117
 6 police     trust          117
 7 majority   joy             60
 8 majority   positive        60
 9 majority   trust           60
10 university anticipation    59
# ... with 1,399 more rows
```

```
# Sentiment Analysis using the nrc dictionary:
wsj_sentiments %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  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()
```




```
# Wordcloud based on nrc dictionary on Wall Street Journal Articles
wsj_sentiments %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=150, scale = c(0.9,0.9),
                   fixed.asp = TRUE,
                   title.size = 1)
```



##

Sentiment Analysis on Trending 20 Times of India Articles about India National News

##

Importing all PDF files from the same folder

setwd("C:/Users/arund/OneDrive/Desktop/Masters in Business Analytics/Text Analytics/Assignments/Individual/pdf_toi")

nm <- list.files(path="C:/Users/arund/OneDrive/Desktop/Masters in Business Analytics/Text Analytics/Assignments/Individual/pdf_toi")

my_pdf_text <- do.call(rbind, lapply(nm, function(x) pdf_text(x)))

my_pdf_text <- as.data.frame(my_pdf_text)

my_pdf_text <- my_pdf_text[,c(2:6)]

```
# Merge all columns

my_df <- unite(my_pdf_text, col = toi, sep = " ")

# Custom stopwords

custom <- data_frame(word = c("http","rt","https","t.io","india","india's","articles",
"copy","copies", "personal","visit","ready",

 "commercial","timesofindia.indiatimes.com","timesofindia.indiatimes.com",
"cms","ad","articleshow",


"news","times","live","results","elections","videos","diva","videoshow","updates","
miss", "shefali","sood",


"daughter's","cancer","9","lakh","rs","treatment","simple","trick","disappointing","
74072501","can;t","afford"),lexicon=rep("custom", each=42))

new_stopwords <- bind_rows(custom, stop_words)


# Tokenization

toi_tokenized <- my_df %>%

  unnest_tokens(word, toi) %>%

  anti_join(new_stopwords) %>% #here's where we remove tokens

  count(word, sort=TRUE)


print(toi_tokenized) # This is Tidy Format
```

```
# A tibble: 1,946 x 2
   word          n
   <chr>     <int>
 1 delhi       573
 2 election    326
 3 2020        193
 4 assembly    132
 5 congress    129
 6 result      106
 7 aap          99
 8 bjp          99
 9 party        96
10 polls        91
# ... with 1,936 more rows
```

```
# Bi-grams
my_bigrams <- my_df %>%
  unnest_tokens(bigram, toi, token = "ngrams", n=2)

bigrams_separated <- my_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")

bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% new_stopwords$word) %>%
  filter(!word2 %in% new_stopwords$word)

bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

print(bigram_counts)
```

```
# A tibble: 1,788 x 3
   word1     word2          n
   <chr>     <chr>      <int>
 1 delhi     election     110
 2 election  result       104
 3 delhi     assembly     100
 4 assembly  election      59
 5 delhi     delhi         59
 6 delhi     polls         59
 7 2020      liveblog      58
 8 congress  candidates    52
 9 arvind    kejriwal      51
10 63        congress      49
# ... with 1,778 more rows
```

```
# Get Sentiments

# Using bing dictionary

toi_sentiments <- my_df %>%

  unnest_tokens(word, toi) %>%

  anti_join(new_stopwords) %>% #here's where we remove tokens

  inner_join(get_sentiments("bing")) %>%

  count(word, sentiment, sort=T) %>%

  ungroup()

print(toi_sentiments)
```

```
# A tibble: 170 x 3
   word       sentiment     n
   <chr>      <chr>     <int>
 1 lose       negative     71
 2 trump      positive     65
 3 victory    positive     63
 4 fans       positive     60
 5 tortured   negative     60
 6 lost       negative     50
 7 top        positive     48
 8 breaking   negative     40
 9 excitement positive     40
10 losing     negative     36
# ... with 160 more rows
```

```
# Using afinn dictionary
toi_sentiments <- my_df %>%
  unnest_tokens(word, toi) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("afinn")) %>%
  count(word, sort=T) %>%
  ungroup()

print(toi_sentiments)
```

```
# A tibble: 150 x 2
   word           n
   <chr>      <int>
 1 stop          63
 2 tortured      60
 3 lost          50
 4 top           48
 5 excitement    40
 6 shares        40
 7 losing        36
 8 treasures     30
 9 free          29
10 popular       26
# ... with 140 more rows
```

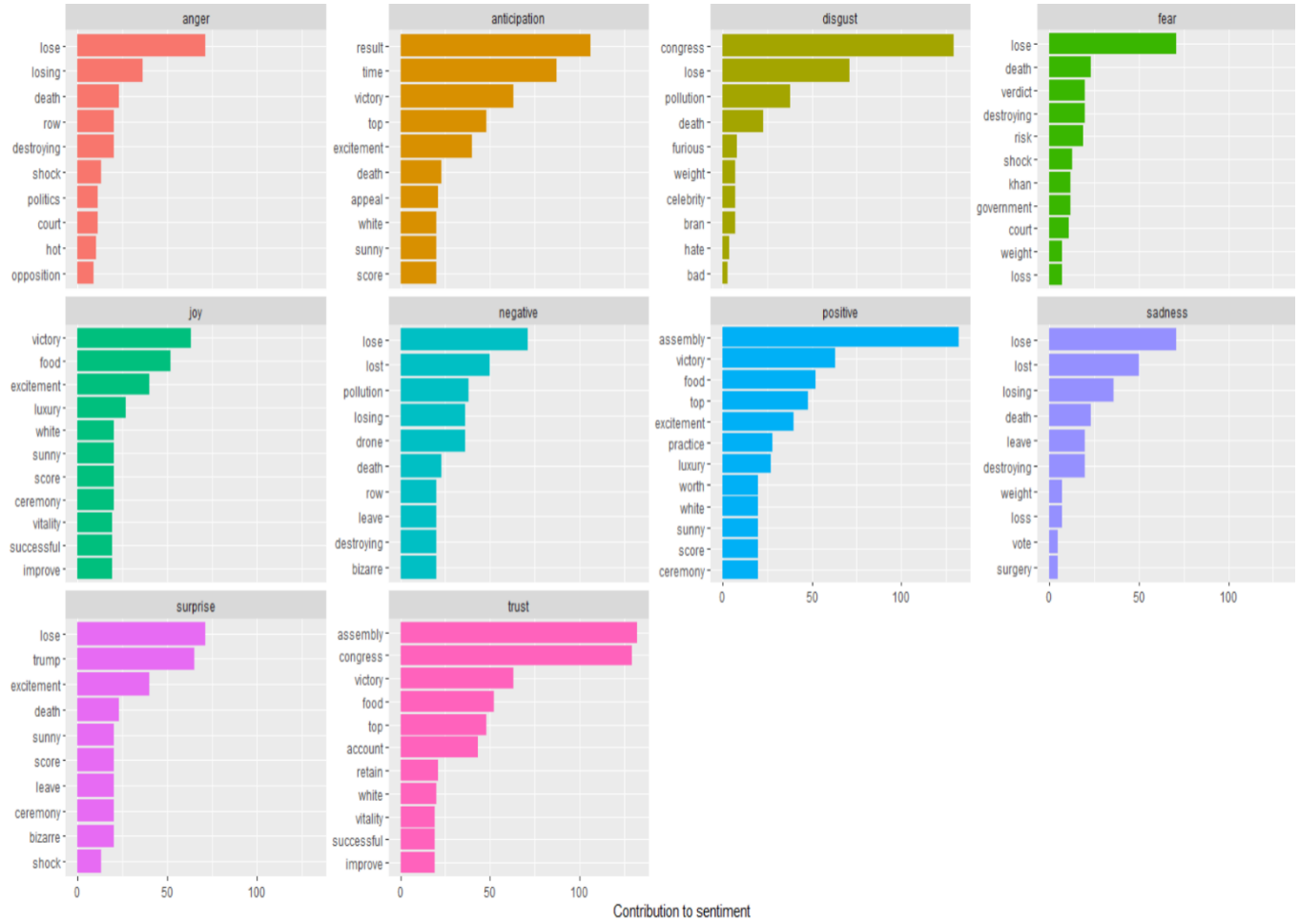
```
# Using nrc dictionary
toi_sentiments <- my_df %>%
  unnest_tokens(word, toi) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()
```

print(toi_sentiments)

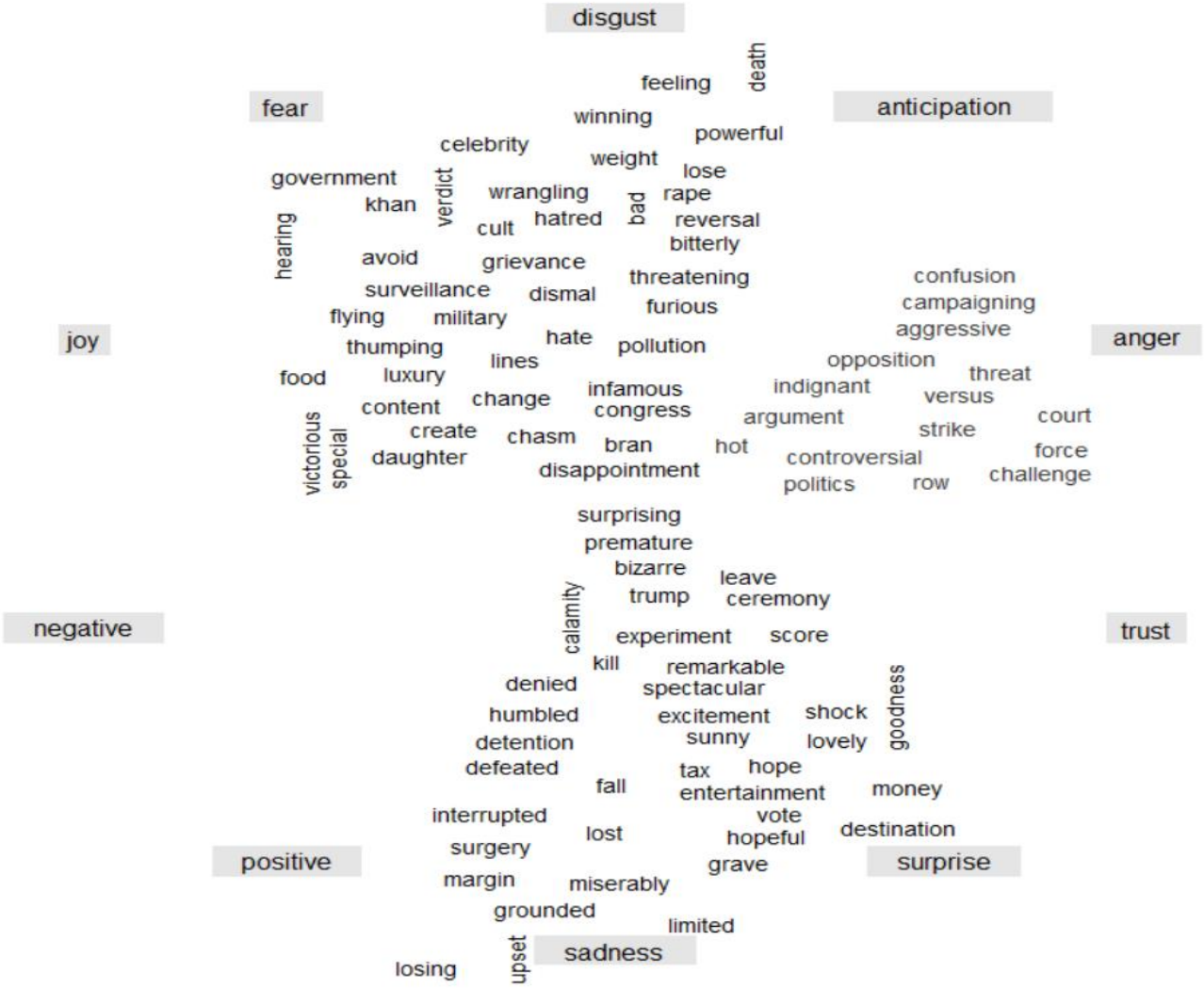
```
# A tibble: 662 x 3
   word     sentiment        n
   <chr>    <chr>        <int>
 1 assembly positive       132
 2 assembly trust          132
 3 congress disgust        129
 4 congress trust          129
 5 result   anticipation   106
 6 time     anticipation    87
 7 lose     anger           71
 8 lose     disgust         71
 9 lose     fear            71
10 lose     negative        71
# ... with 652 more rows
```

Sentiment Analysis using the nrc dictionary:

```
toi_sentiments %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  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()
```




```
# Wordcloud based on nrc dictionary on Wall Street Journal Articles
toi_sentiments %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100, scale = c(0.9,0.9),
                   fixed.asp = TRUE,
                   title.size = 1)
```



##

Sentiment Analysis on Trending 20 Hindustan Times Articles about India National News

##

Importing all PDF files from the same folder

setwd("C:/Users/arund/OneDrive/Desktop/Masters in Business Analytics/Text Analytics/Assignments/Individual/pdf_ht")

nm <- list.files(path="C:/Users/arund/OneDrive/Desktop/Masters in Business Analytics/Text Analytics/Assignments/Individual/pdf_ht")

my_pdf_text <- do.call(rbind, lapply(nm, function(x) pdf_text(x)))

```
my_pdf_text <- as.data.frame(my_pdf_text)

my_pdf_text <- my_pdf_text[,c(2,3)]


# Merge all columns

my_df <- unite(my_pdf_text, col = ht, sep = " ")


# Custom stopwords

custom <- data_frame(word = c("http","rt","https","t.io","india","india's","articles",
"copy","copies", "personal","visit","ready",


"commercial","bjp","aap","ht","top","news","delhi","2020","opinion","inte","city","w
orld","don't","bra","live","trends"),lexicon=rep("custom", each=28))

new_stopwords <- bind_rows(custom, stop_words)


# Tokenization

ht_tokenized <- my_df %>%

  unnest_tokens(word, ht) %>%

  anti_join(new_stopwords) %>% #here's where we remove tokens

  count(word, sort=TRUE)

print(ht_tokenized) # This is Tidy Format
```

```
# A tibble: 1,896 x 2
   word          n
   <chr>     <int>
 1 court        46
 2 election     34
 3 board        33
 4 cricket      32
 5 exams        32
 6 school       32
 7 sports       32
 8 party        29
 9 people       28
10 chief        27
# ... with 1,886 more rows
```

```
# Bi-grams
my_bigrams <- my_df %>%
  unnest_tokens(bigram, ht, token = "ngrams", n=2)

bigrams_separated <- my_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")

bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% new_stopwords$word) %>%
  filter(!word2 %in% new_stopwords$word)

bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

print(bigram_counts)
```

```
# A tibble: 1,550 x 3
   word1      word2          n
   <chr>      <chr>      <int>
 1 board      exams         32
 2 omar's     detention     21
 3 chief      subhash       20
 4 cong       chief         20
 5 crushing   defeat        20
 6 party's    crushing      20
 7 1st        casualty      19
 8 defeat     psa           19
 9 hear       challenge     19
10 kejriwal's party         19
# ... with 1,540 more rows
```

```
# Get Sentiments
# Using bing dictionary
ht_sentiments <- my_df %>%
```

```
  unnest_tokens(word, ht) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

print(ht_sentiments)
```

```
# A tibble: 189 x 3
   word     sentiment     n
   <chr>    <chr>     <int>
 1 defeat   positive     24
 2 crushing negative     20
 3 victory  positive     20
 4 wins     positive     20
 5 casualty negative     19
 6 limited  negative     19
 7 promises positive     19
 8 fall     negative     18
 9 shine    positive     16
10 rape     negative     11
# ... with 179 more rows
```

```
# Using afinn dictionary
ht_sentiments <- my_df %>%
  unnest_tokens(word, ht) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("afinn")) %>%
  count(word, sort=T) %>%
  ungroup()

print(ht_sentiments)
```

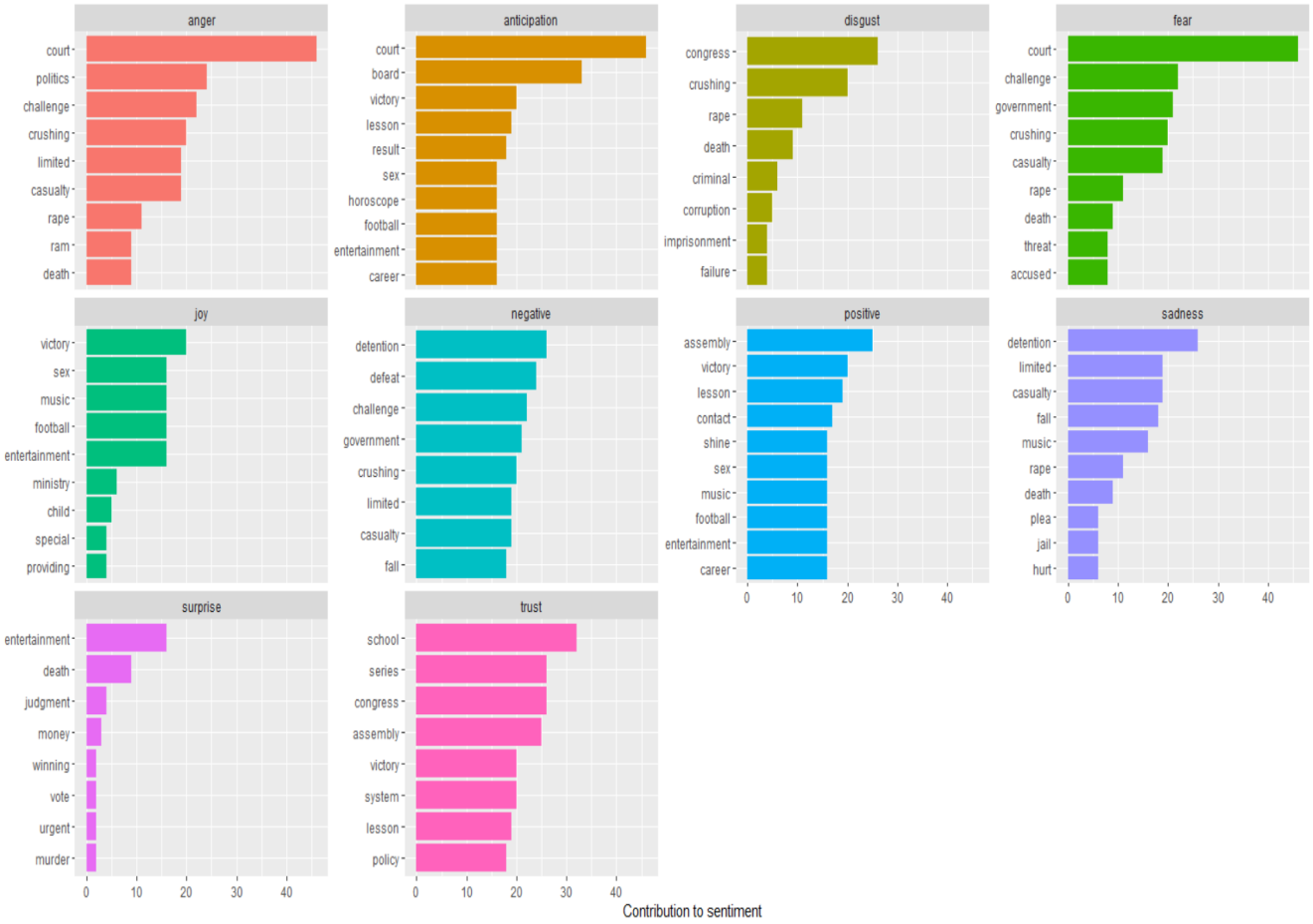
```
# A tibble: 170 x 2
   word          n
   <chr>     <int>
 1 detention    26
 2 challenge    22
 3 crushing     20
 4 wins         20
 5 casualty     19
 6 limited      19
 7 promises     19
 8 rape         11
 9 death         9
10 accused       8
# ... with 160 more rows
```

```
# Using nrc dictionary
ht_sentiments <- my_df %>%
  unnest_tokens(word, ht) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()
print(ht_sentiments)
```

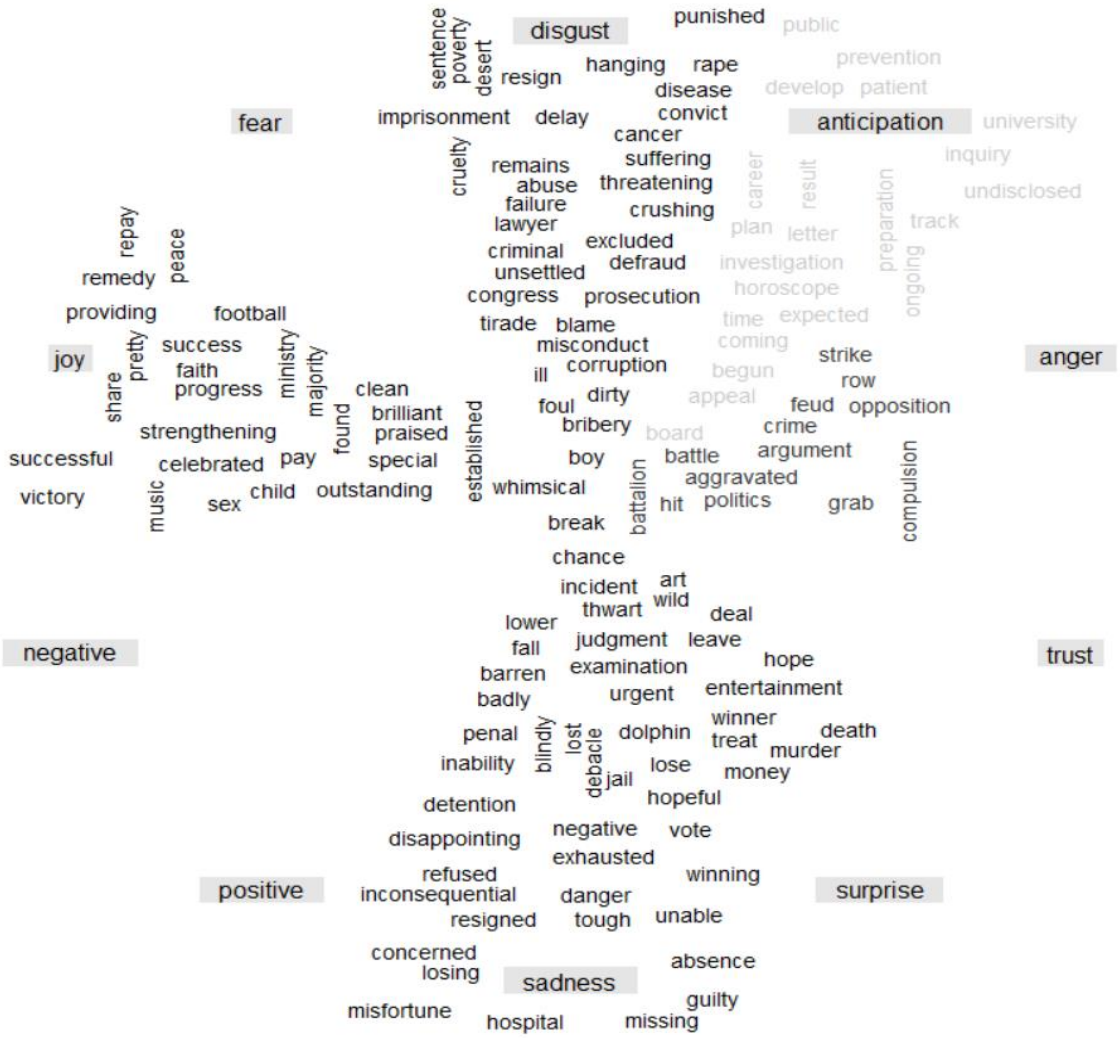
```
# A tibble: 834 x 3
   word      sentiment        n
   <chr>     <chr>        <int>
 1 court     anger           46
 2 court     anticipation    46
 3 court     fear            46
 4 board     anticipation    33
 5 school    trust           32
 6 congress  disgust         26
 7 congress  trust           26
 8 detention negative        26
 9 detention sadness         26
10 series    trust           26
# ... with 824 more rows
```

Sentiment Analysis using the nrc dictionary:

```
ht_sentiments %>%
  group_by(sentiment) %>%
  top_n(8) %>%
  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()
```



```
# Wordcloud based on nrc dictionary on Wall Street Journal Articles
ht_sentiments %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=150, scale = c(0.9,0.9),
                   fixed.asp = TRUE,
                   title.size = 1)
```



```
#####################################################################

Sentiment Analysis on Trending 20 The Indian Express Articles about India National News

#####################################################################

# Importing all PDF files from the same folder

setwd("C:/Users/arund/OneDrive/Desktop/Masters in Business Analytics/Text Analytics/Assignments/Individual/pdf_iex")

nm <- list.files(path="C:/Users/arund/OneDrive/Desktop/Masters in Business Analytics/Text Analytics/Assignments/Individual/pdf_iex")

my_pdf_text <- do.call(rbind, lapply(nm, function(x) pdf_text(x)))

my_pdf_text <- as.data.frame(my_pdf_text)

my_pdf_text <- my_pdf_text[,c(2,3)]


# Merge all columns

my_df <- unite(my_pdf_text, col = iex, sep = " ")


# Custom stopwords

custom <- data_frame(word = c("http","rt","https","t.io","india","india's","articles", "copy","copies", "personal","visit","ready",


"commercial","indian","express","news","click","channel","indianexpress","stay","updated","indianexpress.com",


"post","comment","live","blog","cleared","manually"),lexicon=rep("custom", each=28))

new_stopwords <- bind_rows(custom, stop_words)
```

```
# Tokenization
iex_tokenized <- my_df %>%
  unnest_tokens(word, iex) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  count(word, sort=TRUE)

print(iex_tokenized) # This is Tidy Format
```

```
# A tibble: 1,692 x 2
   word            n
   <chr>       <int>
 1 delhi          34
 2 police         31
 3 government     28
 4 court          20
 5 minister       18
 6 rs             18
 7 women          18
 8 bjp            16
 9 party          16
10 people         16
# ... with 1,682 more rows
```

```
# Bi-grams
my_bigrams <- my_df %>%
  unnest_tokens(bigram, iex, token = "ngrams", n=2)

bigrams_separated <- my_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")

bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% new_stopwords$word) %>%
  filter(!word2 %in% new_stopwords$word)
```

```
bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

print(bigram_counts)
```

```
# A tibble: 1,327 x 3
   word1      word2          n
   <chr>      <chr>      <int>
 1 honey      mahajan        8
 2 rear       admiral        8
 3 worth      rs             8
 4 assembly   polls          7
 5 chief      minister       6
 6 farm       house          6
 7 government hospitals      6
 8 mobile     phones         6
 9 sanjay     vatsayan       6
10 admiral    sanjay         5
# ... with 1,317 more rows
```

```
# Get Sentiments
# Using bing dictionary
iex_sentiments <- my_df %>%
  unnest_tokens(word, iex) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

print(iex_sentiments)
```

```
# A tibble: 161 x 3
   word        sentiment     n
   <chr>       <chr>     <int>
 1 attack      negative      8
 2 protest     negative      8
 3 worth       positive      8
 4 protesting  negative      7
 5 appeal      positive      5
 6 rejected    negative      5
 7 forged      negative      4
 8 mercy       positive      4
 9 plea        negative      4
10 rape        negative      4
# ... with 151 more rows
```

```
# Using afinn dictionary
iex_sentiments <- my_df %>%
  unnest_tokens(word, iex) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("afinn")) %>%
  count(word, sort=T) %>%
  ungroup()
print(iex_sentiments)
```

```
# A tibble: 133 x 2
   word            n
   <chr>       <int>
 1 join           15
 2 attack          8
 3 protest         8
 4 worth           8
 5 justice         7
 6 protesting      7
 7 accused         5
 8 rejected        5
 9 arrested        4
10 mercy           4
# ... with 123 more rows
```

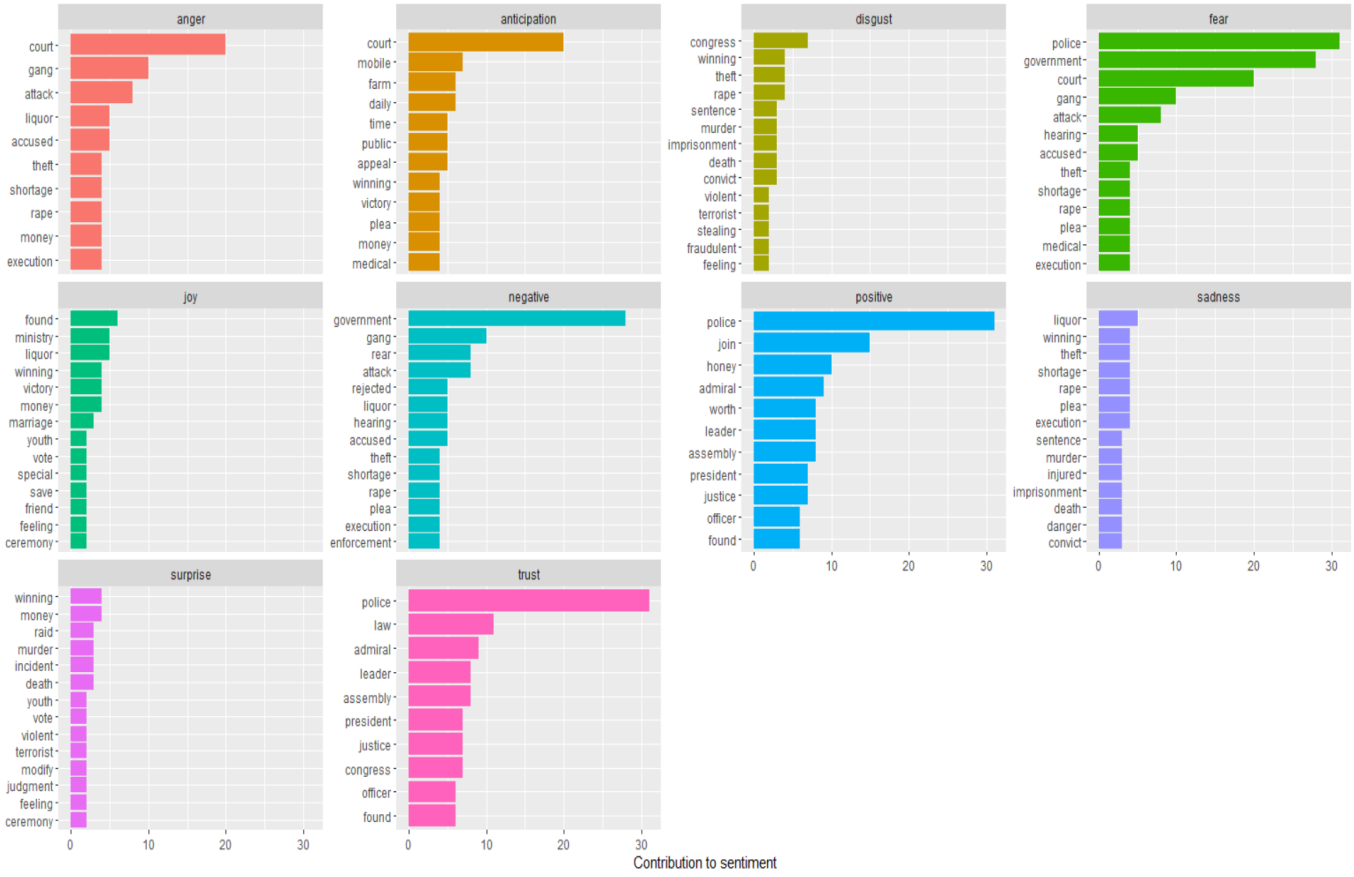
```
# Using nrc dictionary
iex_sentiments <- my_df %>%
  unnest_tokens(word, iex) %>%
  anti_join(new_stopwords) %>% #here's where we remove tokens
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

print(iex_sentiments)
```

```
# A tibble: 706 x 3
   word       sentiment        n
   <chr>      <chr>        <int>
 1 police     fear            31
 2 police     positive        31
 3 police     trust           31
 4 government fear            28
 5 government negative        28
 6 court      anger           20
 7 court      anticipation    20
 8 court      fear            20
 9 join       positive        15
10 law        trust           11
# ... with 696 more rows
```

```
# Sentiment Analysis using the nrc dictionary:
iex_sentiments %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  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()
```

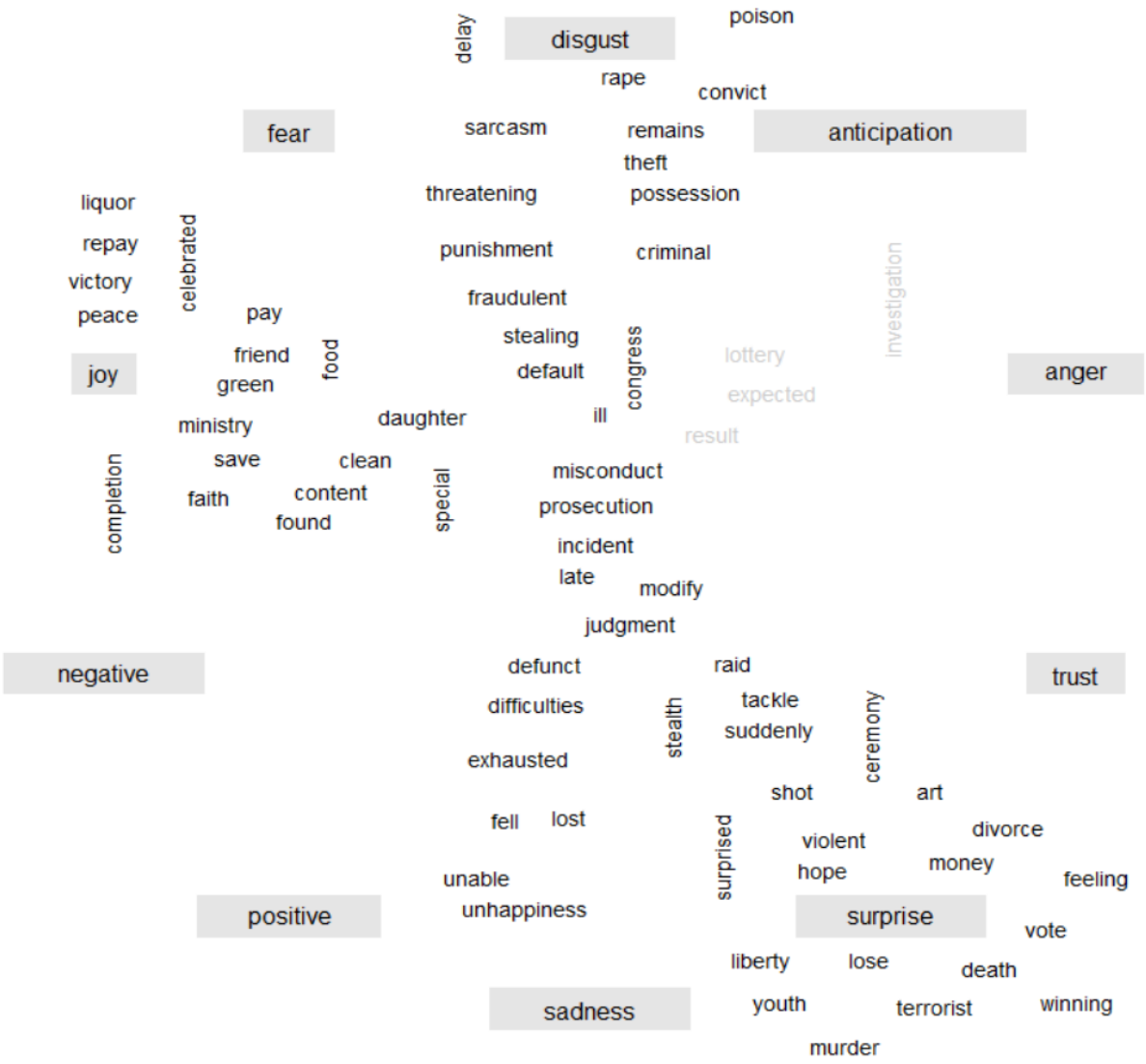



```
# Wordcloud based on nrc dictionary on Wall Street Journal Articles
iex_sentiments %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
```

max.words=80, scale = c(0.9,0.9),

fixed.asp = TRUE,

title.size = 1)



```
#####################################################################
## Finding Unique News using tf-idf ##
#####################################################################
# Combine data of all 4 newspapers
my_df <- bind_rows(
  mutate(wsj_sentiments, newspaper = "Wall Street Journal"),
  mutate(toi_sentiments, newspaper = "Times of India"),
  mutate(ht_sentiments, newspaper = "Hindustan Times"),
  mutate(iex_sentiments, newspaper = "The Indian Express"))

my_df <- my_df %>%
  bind_tf_idf(word, newspaper, n)

my_df %>%
  arrange(desc(tf_idf))

my_df %>%
  arrange(desc(tf_idf)) %>%
  mutate(word=factor(word, levels=rev(unique(word)))) %>%
  group_by(newspaper) %>%
  top_n(15) %>%
  ungroup %>%
  ggplot(aes(word, tf_idf, fill=newspaper))+
  geom_col(show.legend=FALSE)+
```

```
labs(x=NULL, y="tf-idf")+
facet_wrap(~newspaper, ncol=2, scales="free")+
coord_flip()
```