SentimentAnalysisProject_group(asenjo, elizalde, barrientos

Asenjo, Elizalde, Barrientos

2024-12-10

Trend Analysis

```
library(dplyr)
```

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

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

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

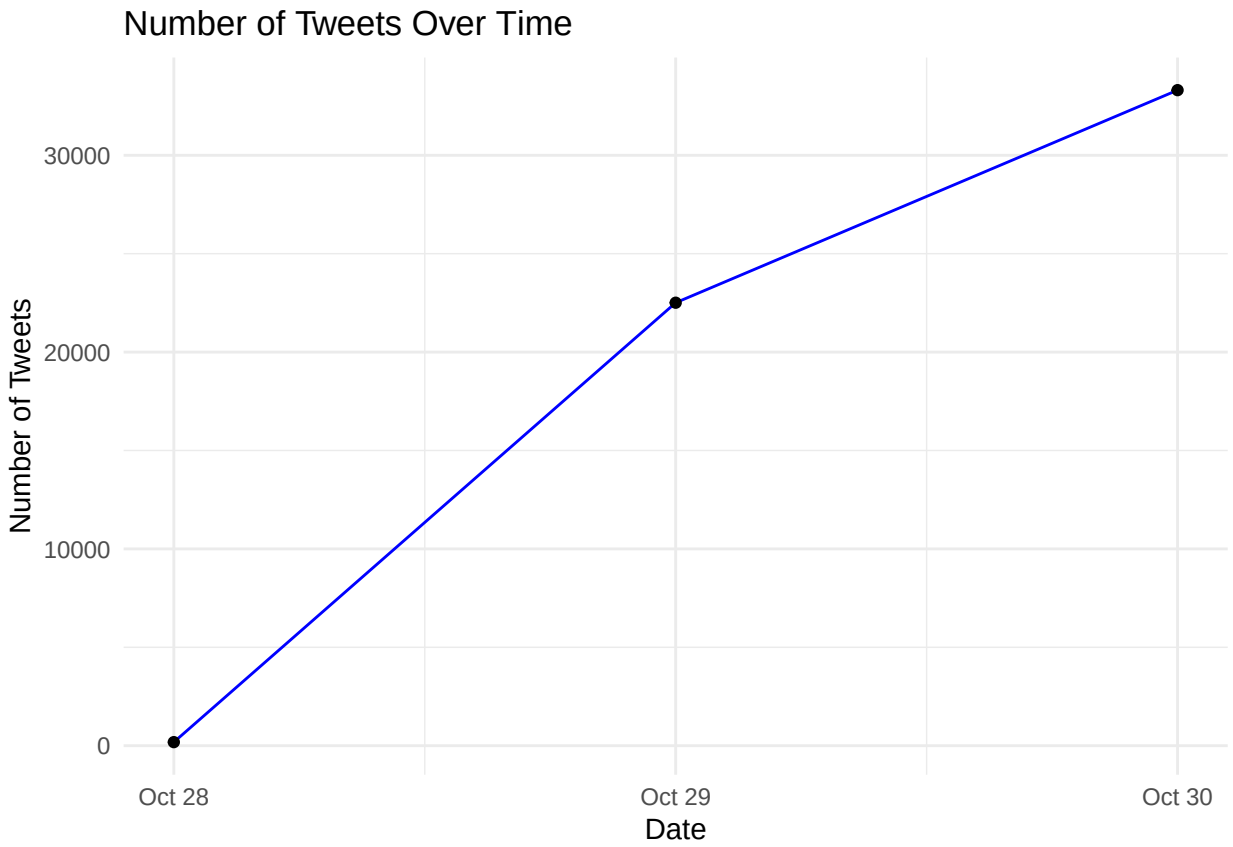
```
library(ggplot2)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

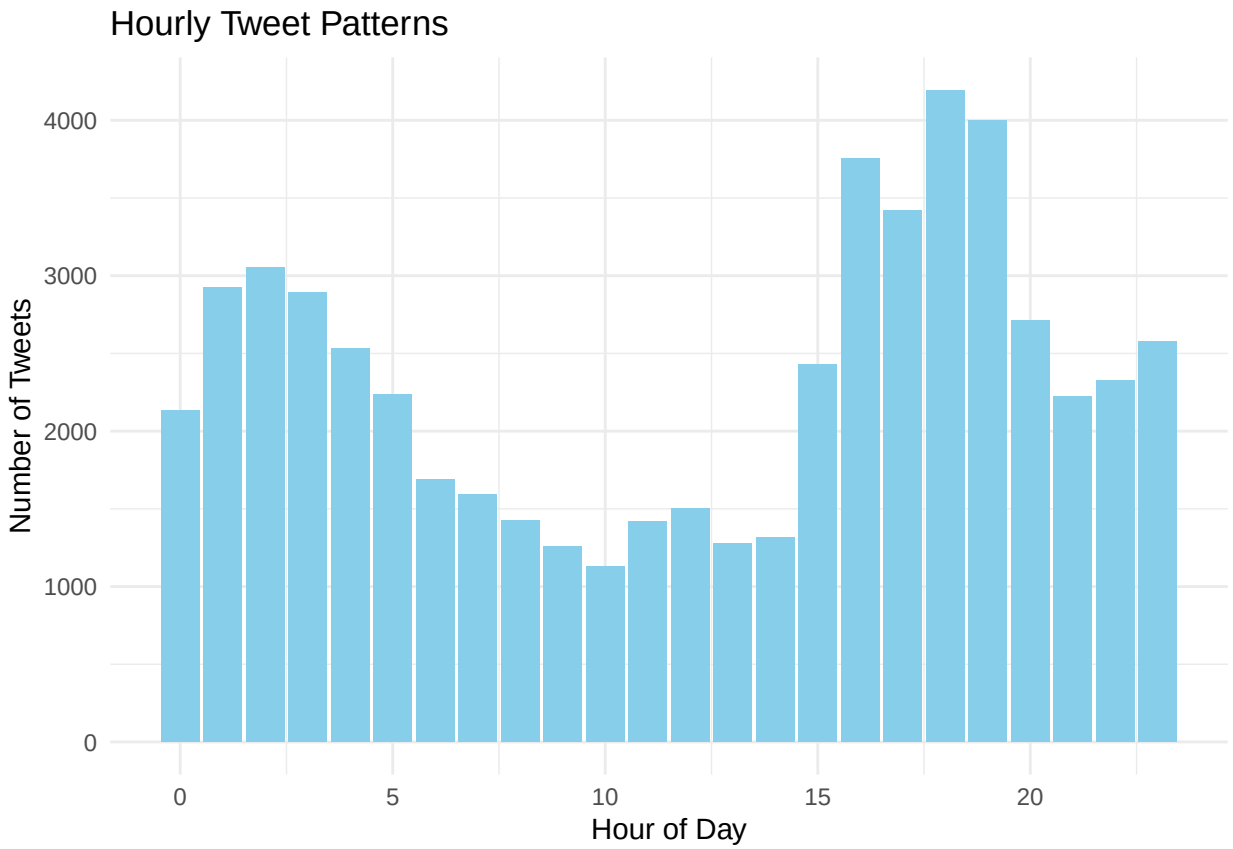
```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(stringr)

tweets_df <- read.csv("/cloud/project/Sentiment_Analysis_Project/tweetsDF.csv")

tweets_df_cleaned <- tweets_df %>%
  select(-c( statusSource, Created_At_Round)) %>%
  mutate(created = as.POSIXct(created, format = "%Y-%m-%d %H:%M:%S"),
         date = as.Date(created),
         hour = hour(created),
         day_of_week = weekdays(created)) %>%
  distinct(text, .keep_all = TRUE)

# Trend 1: Tweet Volume Over Time
tweet_trend <- tweets_df_cleaned %>%
  group_by(date) %>%
  summarise(tweet_count = n())

ggplot(tweet_trend, aes(x = date, y = tweet_count)) +
  geom_line(color = "blue") +
  geom_point() +
```

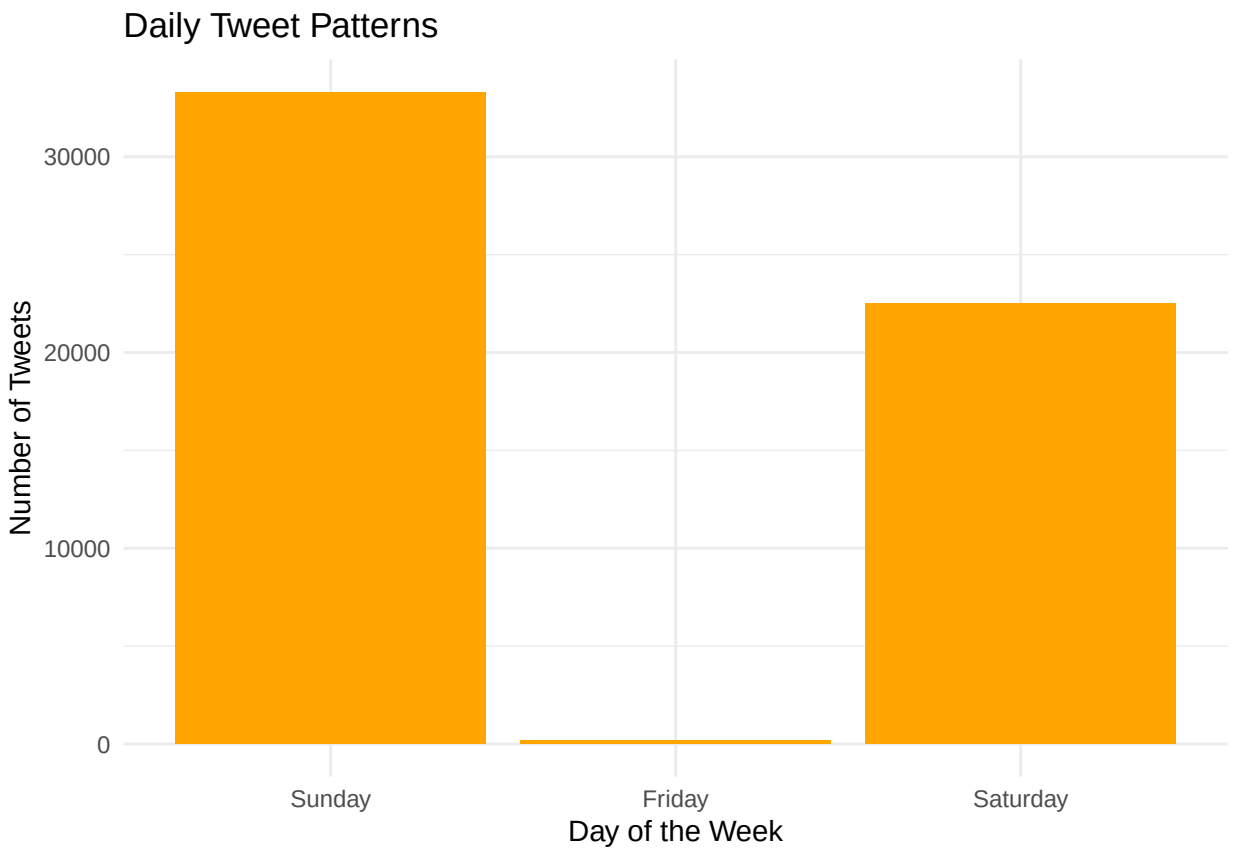
```
  theme_minimal() +
  labs(title = "Number of Tweets Over Time",
       x = "Date",
       y = "Number of Tweets")
```



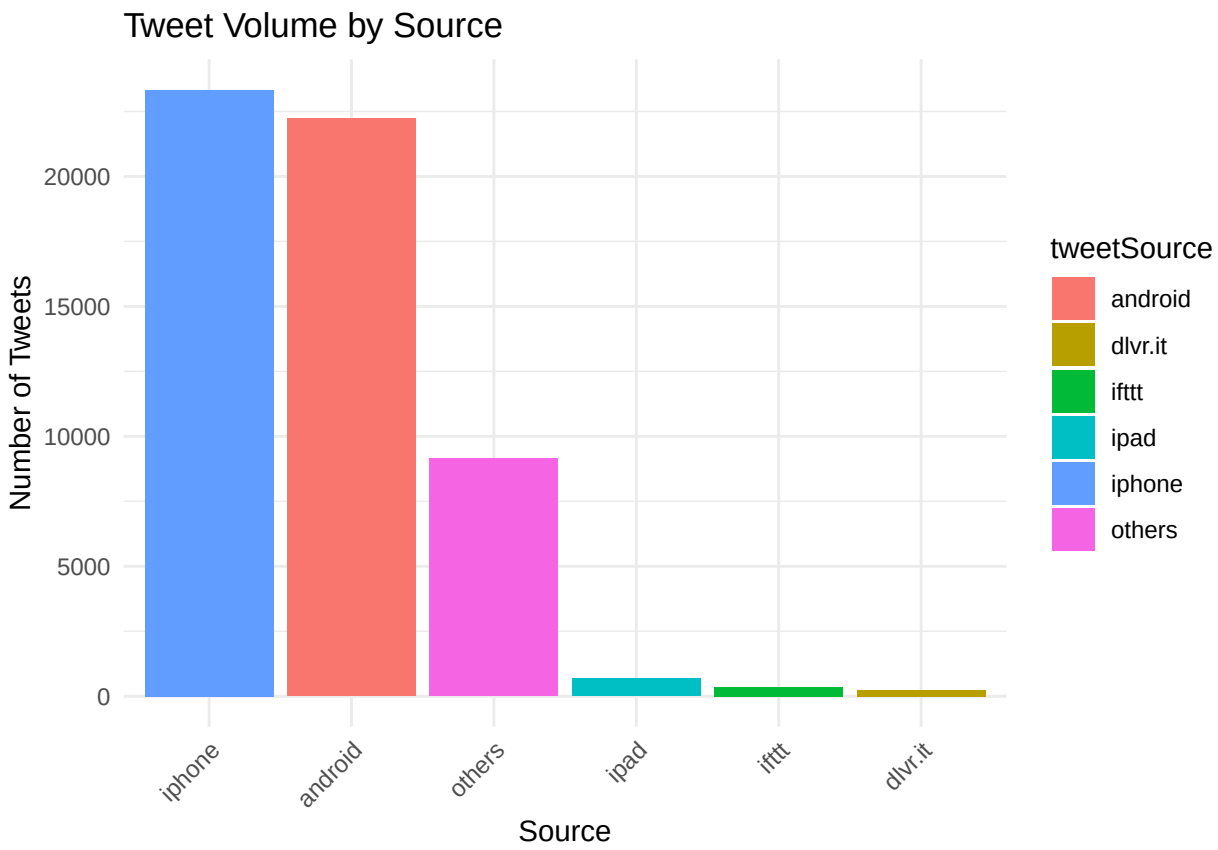

```
# Trend 2: Hourly Tweet Patterns
hourly_trend <- tweets_df_cleaned %>%
  group_by(hour) %>%
  summarise(tweet_count = n())

ggplot(hourly_trend, aes(x = hour, y = tweet_count)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  theme_minimal() +
  labs(title = "Hourly Tweet Patterns",
       x = "Hour of Day",
       y = "Number of Tweets")
```




```r
# Trend 3: Daily Tweet Patterns
daily_trend <- tweets_df_cleaned %>%
  group_by(day_of_week) %>%
  summarise(tweet_count = n()) %>%
  mutate(day_of_week = factor(day_of_week,
                              levels = c("Sunday", "Monday", "Tuesday", "Wednesday",
                                         "Thursday", "Friday", "Saturday")))

ggplot(daily_trend, aes(x = day_of_week, y = tweet_count)) +
  geom_bar(stat = "identity", fill = "orange") +
  theme_minimal() +
  labs(title = "Daily Tweet Patterns",
       x = "Day of the Week",
       y = "Number of Tweets")
```




```
# Trend 4: Trends by Source
source_trend <- tweets_df_cleaned %>%
  group_by(tweetSource) %>%
  summarise(tweet_count = n())

ggplot(source_trend, aes(x = reorder(tweetSource, -tweet_count), y = tweet_count, fill = tweetSource)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Tweet Volume by Source",
       x = "Source",
       y = "Number of Tweets") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
tweet_trend
```

```
## # A tibble: 3 x 2
##   date       tweet_count
##   <date>           <int>
## 1 2022-10-28         181
## 2 2022-10-29       22509
## 3 2022-10-30       33310
```

```
hourly_trend
```

```
## # A tibble: 24 x 2
##     hour tweet_count
##    <int>       <int>
##  1     0        2131
##  2     1        2922
##  3     2        3050
##  4     3        2892
##  5     4        2529
##  6     5        2237
##  7     6        1688
##  8     7        1592
##  9     8        1428
## 10     9        1256
## # i 14 more rows
```

```
daily_trend
```

```
## # A tibble: 3 x 2
```

```
##   day_of_week tweet_count
##   <fct>             <int>
## 1 Friday              181
## 2 Saturday          22509
## 3 Sunday            33310
```

```
source_trend
```

```
## # A tibble: 6 x 2
##   tweetSource tweet_count
##   <chr>             <int>
## 1 android           22227
## 2 dlvr.it             241
## 3 ifttt               364
## 4 ipad                685
## 5 iphone            23336
## 6 others             9147
```

1. Tweet Volume Over Time

Peaks in the graph indicate days with high tweet activity, which might correspond to specific events or trending topics. Periods with low activity could indicate times of general inactivity or reduced interest in the topic. Long-term trends (upward or downward) reveal growing or waning interest over time.

2. Hourly Tweet Patterns

Highlights peak tweeting hours, which could indicate when users are most active. A sharp increase in tweets during certain hours (e.g., evenings) may align with user habits like after-work or leisure time. Hours with low activity might reflect times when users are less engaged, such as early mornings.

3. Daily Tweet Patterns

Helps identify the most active days for tweeting. For example, weekends might see a spike in activity if users are more active on social media during their free time. Lower activity on specific weekdays could reflect work-related distractions or other commitments. Patterns can inform scheduling strategies for social media engagement.

4. Trends by Source Identifies the most commonly used platforms for tweeting. For example, a higher number of tweets from Android or iOS suggests mobile dominance. Variations in source usage might reflect demographic differences (e.g., tech preferences by region or age). Uncommon platforms could indicate niche users or automated posting tools.

General Insights from Trends: Temporal Engagement: Understanding when (daily, hourly) users are most active helps target specific times for tweets or campaigns.

Event Identification: Peaks in tweet volume can be mapped to real-world events, providing context to spikes.

User Behavior Analysis: Patterns across days, hours, and platforms reveal user preferences and habits.

Platform Optimization: Insights from source trends can guide platform-specific optimizations, such as improving mobile app experiences.

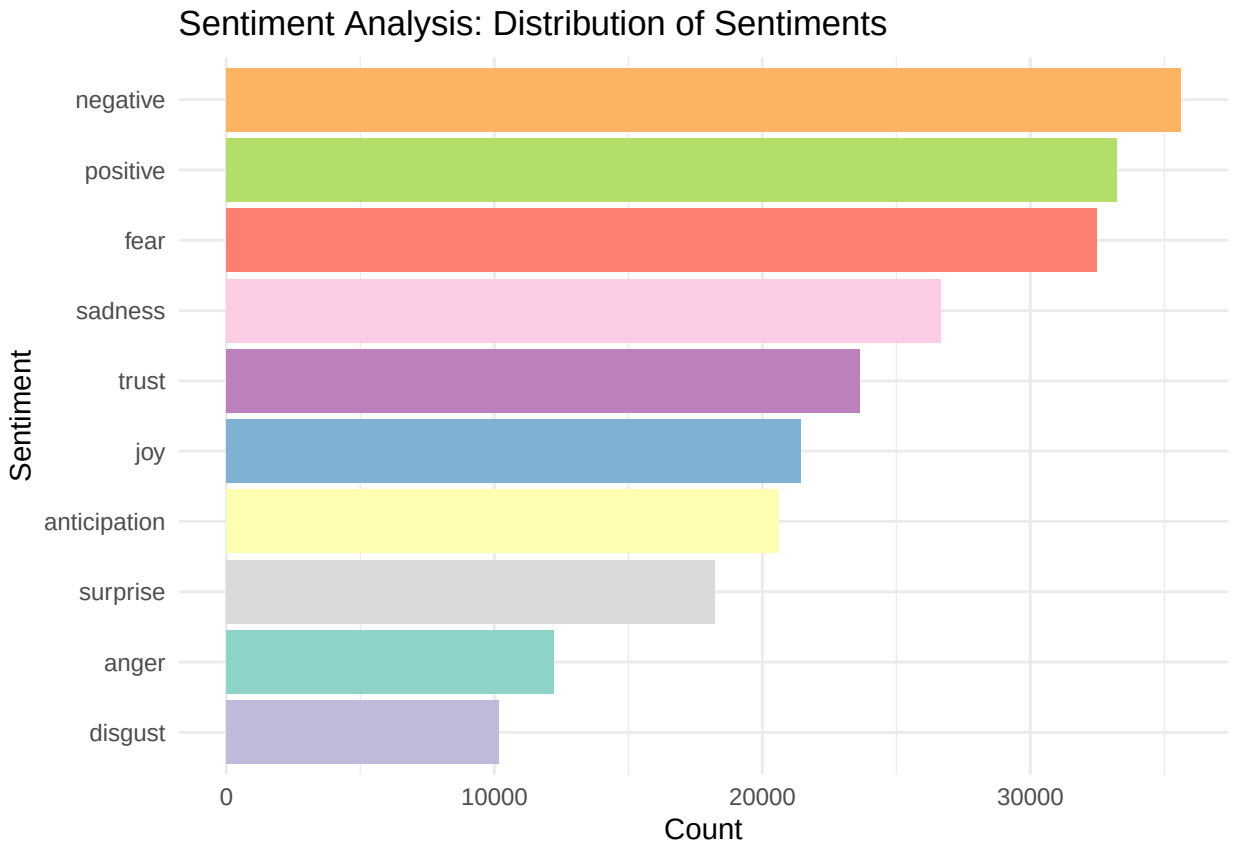
Sentiment Analysis

```
library(dplyr)
library(tidytext)
library(ggplot2)
library(textdata)

tweets_df <- read.csv("/cloud/project/Sentiment_Analysis_Project/tweetsDF.csv")
```

```r
tweets_df_cleaned <- tweets_df %>%
  select(text) %>%
  distinct(text, .keep_all = TRUE)

tweet_words <- tweets_df_cleaned %>%
  unnest_tokens(word, text)

data("stop_words")
tweet_words <- tweet_words %>%
  anti_join(stop_words, by = "word")

nrc_sentiments <- get_sentiments("nrc")

tweet_sentiment <- tweet_words %>%
  inner_join(nrc_sentiments, by = "word") %>%
  count(sentiment, sort = TRUE)
```

```
## Warning in inner_join(., nrc_sentiments, by = "word"): Detected an unexpected many-to-many relationsh
## i Row 1 of `x` matches multiple rows in `y`.
## i Row 1995 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```r
ggplot(tweet_sentiment, aes(x = reorder(sentiment, n), y = n, fill = sentiment)) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  coord_flip() +
  theme_minimal() +
  labs(title = "Sentiment Analysis: Distribution of Sentiments",
       x = "Sentiment",
       y = "Count") +
  scale_fill_brewer(palette = "Set3")
```




```
tweet_sentiment
```

```
##       sentiment     n
## 1      negative 35597
## 2      positive 33241
## 3          fear 32481
## 4       sadness 26655
## 5         trust 23628
## 6           joy 21429
## 7  anticipation 20600
## 8      surprise 18234
## 9         anger 12221
## 10      disgust 10166
```

Graph Insights: Overall Sentiment Distribution: The bar plot provides insights into the dominant emotional tones expressed in the tweets.

Positive/Negative Sentiment: Indicates overall sentiment, e.g., a predominance of positive tweets may suggest general optimism. Emotions like Joy, Anger, or Trust: Can provide deeper insights into user feelings and responses to specific events. Event Detection: Spikes in certain sentiments (like Anger or Joy) may correlate with specific events or public reactions.

Actionable Insights:

Positive Sentiment: Can be used to reinforce positive campaigns or messages.

Negative Sentiment: Suggests areas for improvement or customer concerns that need attention.

Trust and Anticipation: Indicates user confidence and interest, which could be leveraged for engagement.

Potential Applications: Brand Monitoring: Track how users feel about a product or service.

Event-driven Sentiment: Analyze reactions to specific events, such as a product launch or news event.

User Engagement: Adjust engagement strategies based on emotional responses from users.

This sentiment analysis provides a clear understanding of the emotional tone of the tweets, enabling you to make data-driven decisions for improving user experience, brand perception, and content strategy.

Use Case: Social Media Sentiment Analysis for Brand Monitoring

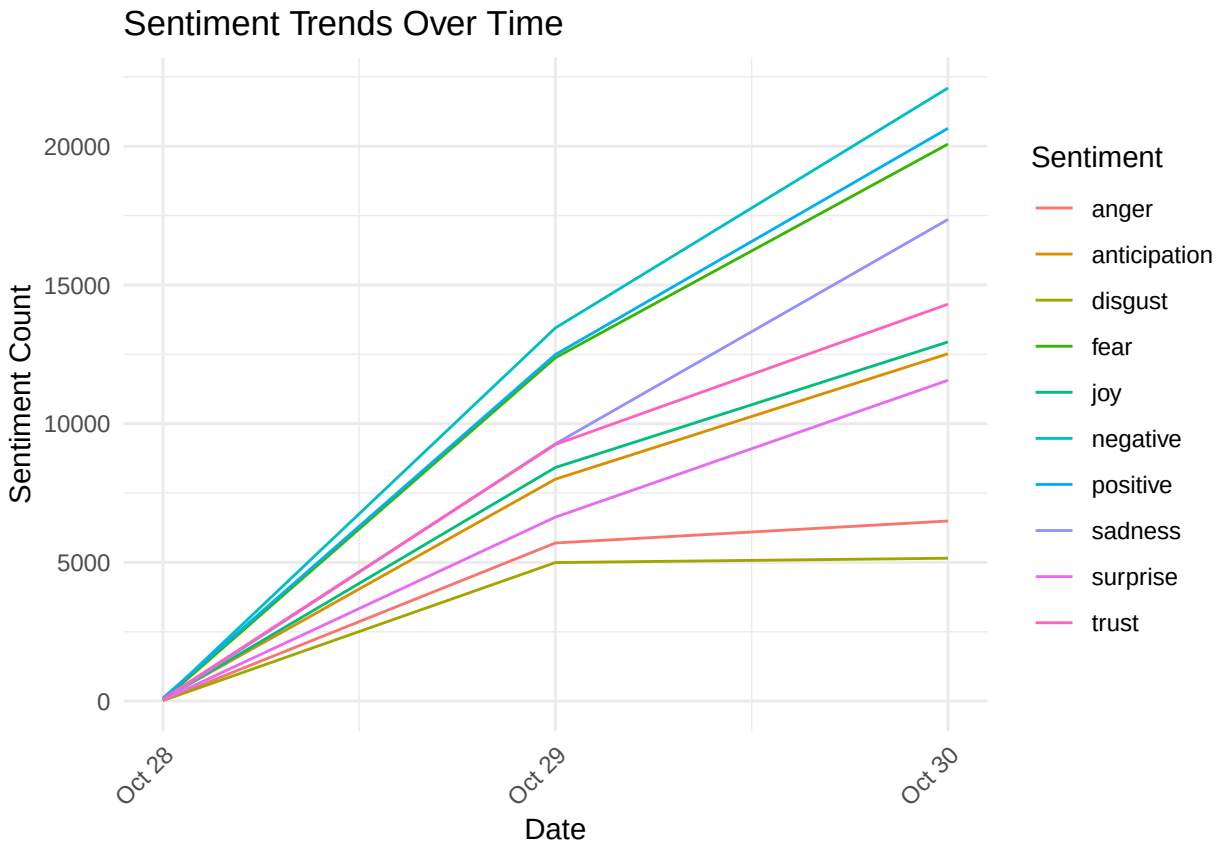
```
library(dplyr)
library(tidytext)
library(ggplot2)
library(lubridate)

tweets_df <- read.csv("/cloud/project/Sentiment_Analysis_Project/tweetsDF.csv")

tweets_df_cleaned <- tweets_df %>%
  select(created, text) %>%
  distinct(text, .keep_all = TRUE) %>%
  filter(!is.na(text))

tweets_df_cleaned$created <- as.Date(tweets_df_cleaned$created)

tweet_words <- tweets_df_cleaned %>%
  unnest_tokens(word, text)

data("stop_words")
tweet_words <- tweet_words %>%
  anti_join(stop_words, by = "word")

nrc_sentiments <- get_sentiments("nrc")
tweet_sentiment <- tweet_words %>%
  inner_join(nrc_sentiments, by = "word") %>%
  count(created, sentiment, sort = TRUE)
```

```
## Warning in inner_join(., nrc_sentiments, by = "word"): Detected an unexpected many-to-many relationsl
## i Row 1 of `x` matches multiple rows in `y`.
## i Row 1995 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

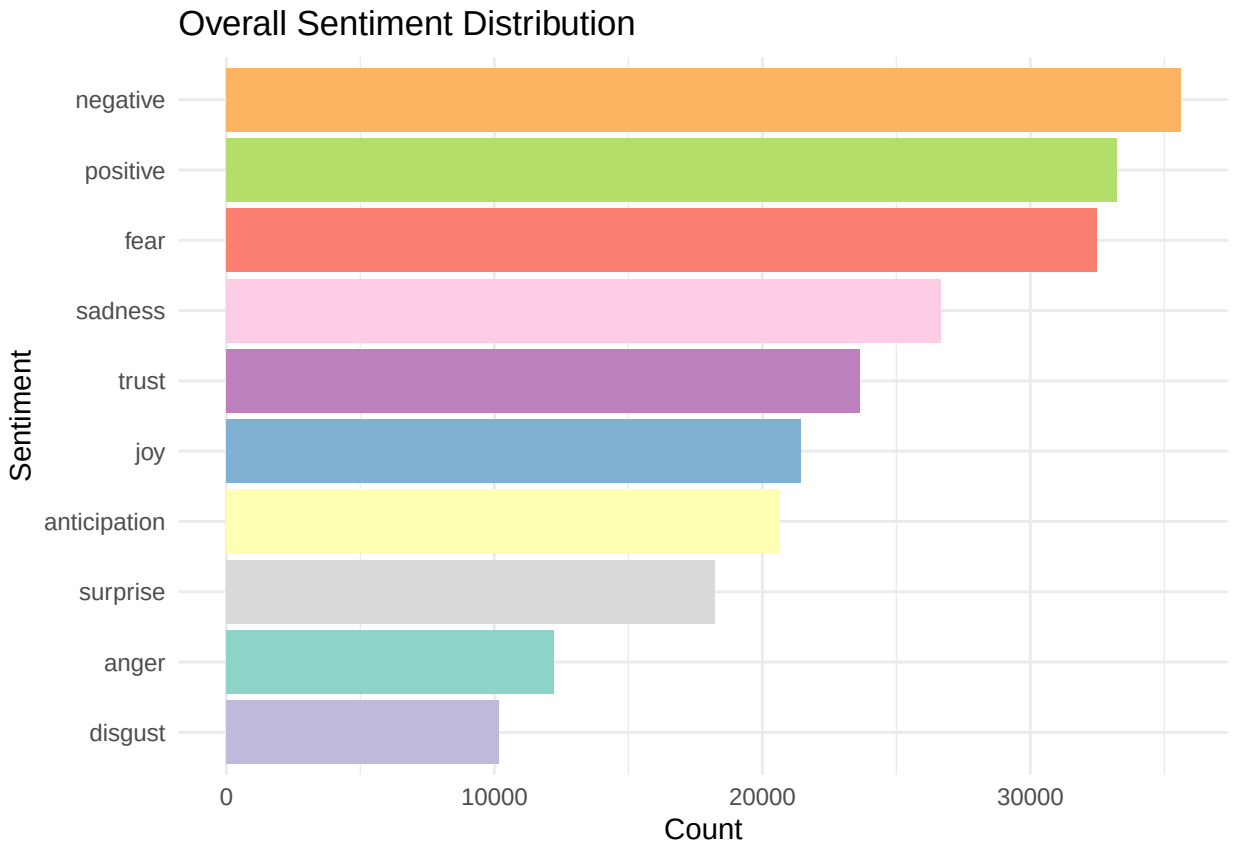
```
sentiment_trends <- tweet_sentiment %>%
  group_by(created, sentiment) %>%
  summarise(daily_sentiment_count = sum(n)) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'created'. You can override using the
## `.groups` argument.
```

```
ggplot(sentiment_trends, aes(x = created, y = daily_sentiment_count, color = sentiment)) +
  geom_line() +
  theme_minimal() +
  labs(title = "Sentiment Trends Over Time",
       x = "Date",
       y = "Sentiment Count",
       color = "Sentiment") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
sentiment_distribution <- tweet_sentiment %>%
  group_by(sentiment) %>%
  summarise(sentiment_count = sum(n)) %>%
  ungroup()

ggplot(sentiment_distribution, aes(x = reorder(sentiment, sentiment_count), y = sentiment_count, fill =
  geom_bar(stat = "identity", show.legend = FALSE) +
  coord_flip() +
  theme_minimal() +
  labs(title = "Overall Sentiment Distribution",
       x = "Sentiment",
       y = "Count") +
  scale_fill_brewer(palette = "Set3")
```




```
positive_tweets <- tweet_sentiment %>%
  filter(sentiment == "positive") %>%
  summarise(positive_tweet_count = sum(n))

negative_tweets <- tweet_sentiment %>%
  filter(sentiment == "negative") %>%
  summarise(negative_tweet_count = sum(n))

print(paste("Number of Positive Tweets: ", positive_tweets$positive_tweet_count))
```

```
## [1] "Number of Positive Tweets:  33241"
```

```
print(paste("Number of Negative Tweets: ", negative_tweets$negative_tweet_count))
```

```
## [1] "Number of Negative Tweets:  35597"
```

```
if (negative_tweets$negative_tweet_count > positive_tweets$positive_tweet_count) {
  print("Warning: High number of negative sentiments. Immediate action may be required.")
} else {
  print("Brand is receiving positive feedback overall. Consider enhancing positive campaigns.")
}
```

```
## [1] "Warning: High number of negative sentiments. Immediate action may be required."
```

Use Case: Social Media Sentiment Analysis for Brand Monitoring The goal is to analyze tweets to understand public sentiment toward a brand, product, or event. By performing sentiment analysis, businesses can:

Identify Sentiments: Determine if tweets are positive, negative, or neutral, and analyze emotions like anger, joy, trust, or fear.

Track Trends: Monitor sentiment trends over time to detect shifts in customer satisfaction or reactions to events.

Targeted Marketing: Engage with satisfied customers (positive sentiment) for campaigns and address dissatisfied customers (negative sentiment) to resolve issues.

Product Improvement: Use feedback from sentiment analysis to identify areas for product or service enhancement.

Crisis Management: Identify negative sentiment spikes early to mitigate potential PR crises.

Brand Positioning: Use positive sentiment to reinforce the brand's strengths in marketing and customer engagement.