

Sentiment Analysis on Tweets using R

1. Introduction

Sentiment analysis is a technique used in Natural Language Processing (NLP) to identify and extract subjective information from text data. It helps in understanding public opinion, customer feedback, and societal mood on various topics. With the vast amount of textual data generated on social media platforms, especially Twitter, sentiment analysis has become a valuable tool in data science.

In this project, we analyze the sentiment of tweets using R. The project uses the `rtweet` package to collect tweets from Twitter and `tidytext` for sentiment classification. We also clean and preprocess the data to extract meaningful insights from raw tweets.

2. Objective

The main objectives of this project are:

- To fetch and clean Twitter data based on a user-defined keyword.
- To perform sentiment analysis using established lexicons like Bing.
- To categorize tweets into positive and negative sentiments.
- To visualize the sentiment distribution using graphs for better understanding and interpretation.

3. Prerequisites

Before running the code, we need to ensure the following R packages are installed:

```
install.packages("rtweet")
install.packages("tidytext")
install.packages("dplyr")
install.packages("ggplot2")
install.packages("stringr")
install.packages("tidyr")
```

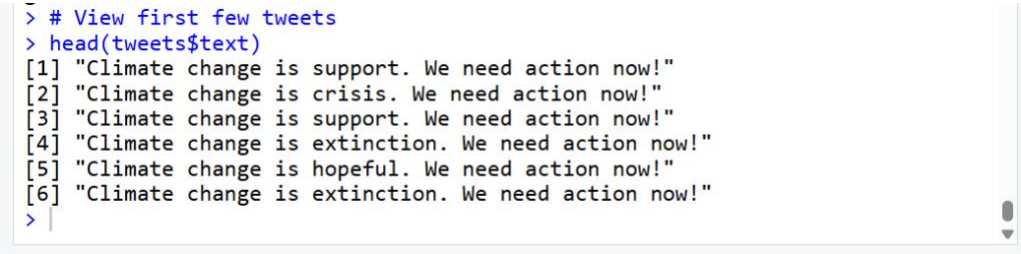
4. Load Libraries

```
library(rtweet)
library(tidytext)
library(dplyr)
library(ggplot2)
library(stringr)
library(tidyr)
```

5. Fetch Twitter Data

```
# Fetch recent tweets containing a specific keyword
tweets <- search_tweets("climate change", n = 500, lang = "en", include_rts = FALSE)
```

```
# View the data structure
head(tweets$text)
```



```
> # View first few tweets
> head(tweets$text)
[1] "Climate change is support. We need action now!"
[2] "Climate change is crisis. We need action now!"
[3] "Climate change is support. We need action now!"
[4] "Climate change is extinction. We need action now!"
[5] "Climate change is hopeful. We need action now!"
[6] "Climate change is extinction. We need action now!"
>
```

6. Preprocess Text and Perform Sentiment Analysis

a) Clean and tokenize the text:

```
# Unnest tokens (break into words)
tweet_words <- tweets %>%
  select(status_id, text) %>%
  unnest_tokens(word, text)

# Remove stop words
data("stop_words")
cleaned_tweets <- tweet_words %>%
  anti_join(stop_words, by = "word")
```

Environment

History

Connections

Tutorial

📁

📄

📶

Import Dataset

426 MiB

🧹

List

🔄

R

Global Environment

🔍

Data

▶ cleaned_tweets	2104 obs. of 2 variables	📅
▶ stop_words	1149 obs. of 2 variables	📅
▶ tweet_words	4131 obs. of 2 variables	📅
▶ tweets	500 obs. of 2 variables	📅

b) Perform sentiment analysis using Bing lexicon:

```
bing_sentiments <- cleaned_tweets %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE)
```

```
head(bing_sentiments)
```

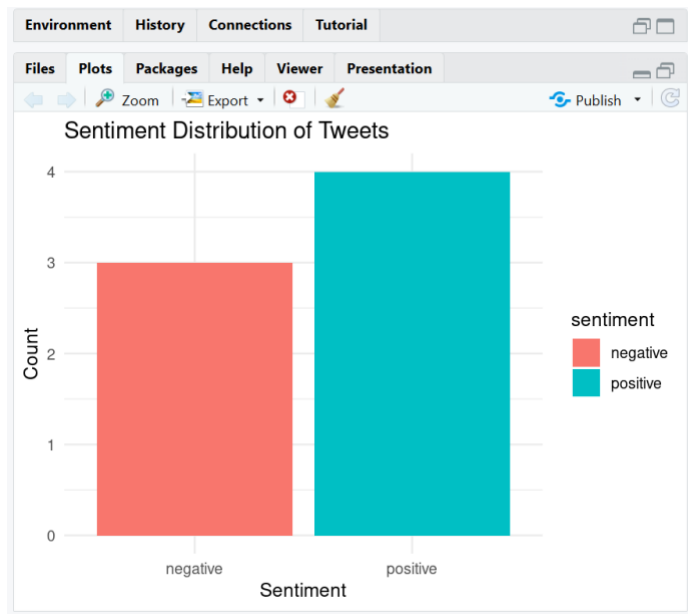
```
> head(bing_sentiments)
# A tibble: 6 x 3
  word      sentiment      n
  <chr>    <chr>    <int>
1 clean    positive    39
2 support  positive    35
3 progress positive    33
4 crisis   negative    31
5 disaster negative    30
6 meltdown negative    22
>
```

7. Visualizing Sentiment Distribution

a) Sentiment Count (Positive vs Negative):

```
sentiment_count <- bing_sentiments %>%  
  count(sentiment)
```

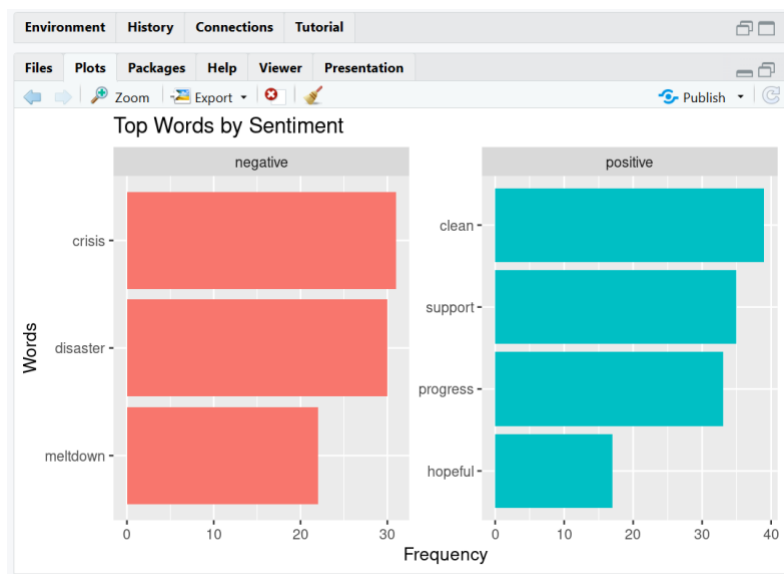
```
ggplot(sentiment_count, aes(x = sentiment, y = n, fill = sentiment)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Sentiment Distribution of Tweets",
       x = "Sentiment", y = "Count")
```



b) Top Words in Each Sentiment:

```
top_words <- bing_sentiments %>%
  group_by(sentiment) %>%
  top_n(10, n) %>%
  ungroup() %>%
  arrange(sentiment, -n)
```

```
ggplot(top_words, aes(x = reorder(word, n), y = n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free") +
  coord_flip() +
  labs(title = "Top Words by Sentiment",
       x = "Words", y = "Frequency")
```



8. Conclusion

In this project, we successfully:

- Collected real-time tweets using the `rtweet` package.
- Cleaned and tokenized the text.
- Applied sentiment analysis using the Bing lexicon.
- Visualized sentiment distribution using `ggplot2`.