TWITTER SENTIMENT ANALYSIS

IMPORTING LIBRARIES

```
import pandas as pd
```

LOAD AND PREVIEW DATASET

```
df = pd.read_csv('twitter_dataset.csv')
print(df.info())
print(df.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   Tweet_ID   10000 non-null  int64 
 1   Username   10000 non-null  object
 2   Text       10000 non-null  object
 3   Retweets   10000 non-null  int64 
 4   Likes      10000 non-null  int64 
 5   Timestamp  10000 non-null  object
dtypes: int64(3), object(3)
memory usage: 468.9+ KB
None
   Tweet_ID        Username  \
0         1         julie81   
1         2   richardhester   
2         3  williamsjoseph   
3         4     danielsmary   
4         5      carlwarren   

                                                Text  Retweets  Likes
\
0  Party least receive say or single. Prevent pre...         2     25   

1  Hotel still Congress may member staff. Media d...        35     29   

2  Nice be her debate industry that year. Film wh...        51     25   

3  Laugh explain situation career occur serious. ...        37     18   

4  Involve sense former often approach government...        27     80   


             Timestamp  
0  2023-01-30 11:00:51  
```

```
1   2023-01-02 22:45:58
2   2023-01-18 11:25:19
3   2023-04-10 22:06:29
4   2023-01-24 07:12:21
```

Convert Timestamp to datetime format

```
df['Timestamp'] = pd.to_datetime(df['Timestamp'])
```

Function to clean text by removing URLs, mentions, hashtags, and special characters

```
import re

def clean_text(text):
    text = re.sub(r'http\S+', '', text)  # Remove URLs
    text = re.sub(r'@\w+', '', text)    # Remove mentions
    text = re.sub(r'#\w+', '', text)    # Remove hashtags
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Remove special 
characters and numbers
    text = text.lower().strip()  # Convert to lowercase and strip 
whitespace
    return text
```

Applying the cleaning function to the Text column

```
df['Cleaned_Text'] = df['Text'].apply(clean_text)
```

Checking the cleaned data

```
df[['Text', 'Cleaned_Text', 'Timestamp']].head()

                                                Text  \
0  Party least receive say or single. Prevent pre...   
1  Hotel still Congress may member staff. Media d...   
2  Nice be her debate industry that year. Film wh...   
3  Laugh explain situation career occur serious. ...   
4  Involve sense former often approach government...   

                                        Cleaned_Text  
Timestamp  
0  party least receive say or single prevent prev... 2023-01-30 
11:00:51  
1  hotel still congress may member staff media dr... 2023-01-02 
22:45:58  
2  nice be her debate industry that year film whe... 2023-01-18 
11:25:19  
3  laugh explain situation career occur serious f... 2023-04-10 
22:06:29  
```

```
4  involve sense former often approach government... 2023-01-24
07:12:21
```

```
!pip install textblob
from textblob import TextBlob
```

```
Requirement already satisfied: textblob in c:\users\user\anaconda3\
lib\site-packages (0.18.0.post0)
Requirement already satisfied: nltk>=3.8 in c:\users\user\anaconda3\
lib\site-packages (from textblob) (3.8.1)
Requirement already satisfied: click in c:\users\user\anaconda3\lib\
site-packages (from nltk>=3.8->textblob) (8.1.7)
Requirement already satisfied: joblib in c:\users\user\anaconda3\lib\
site-packages (from nltk>=3.8->textblob) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in c:\users\user\
anaconda3\lib\site-packages (from nltk>=3.8->textblob) (2023.10.3)
Requirement already satisfied: tqdm in c:\users\user\anaconda3\lib\
site-packages (from nltk>=3.8->textblob) (4.66.4)
Requirement already satisfied: colorama in c:\users\user\anaconda3\
lib\site-packages (from click->nltk>=3.8->textblob) (0.4.6)
```

Function to compute sentiment polarity

```
def get_sentiment_polarity(text):
    return TextBlob(text).sentiment.polarity
```

Applying sentiment analysis to the Cleaned_Text column

```
df['Sentiment_Polarity'] =
df['Cleaned_Text'].apply(get_sentiment_polarity)
```

Categorize sentiment as Positive, Neutral, or Negative based on polarity score

```
def categorize_sentiment(polarity):
    if polarity > 0:
        return 'Positive'
    elif polarity < 0:
        return 'Negative'
    else:
        return 'Neutral'

df['Sentiment_Category'] =
df['Sentiment_Polarity'].apply(categorize_sentiment)
```

Displaying the results

```
df[['Cleaned_Text', 'Sentiment_Polarity',
'Sentiment_Category']].head()
```

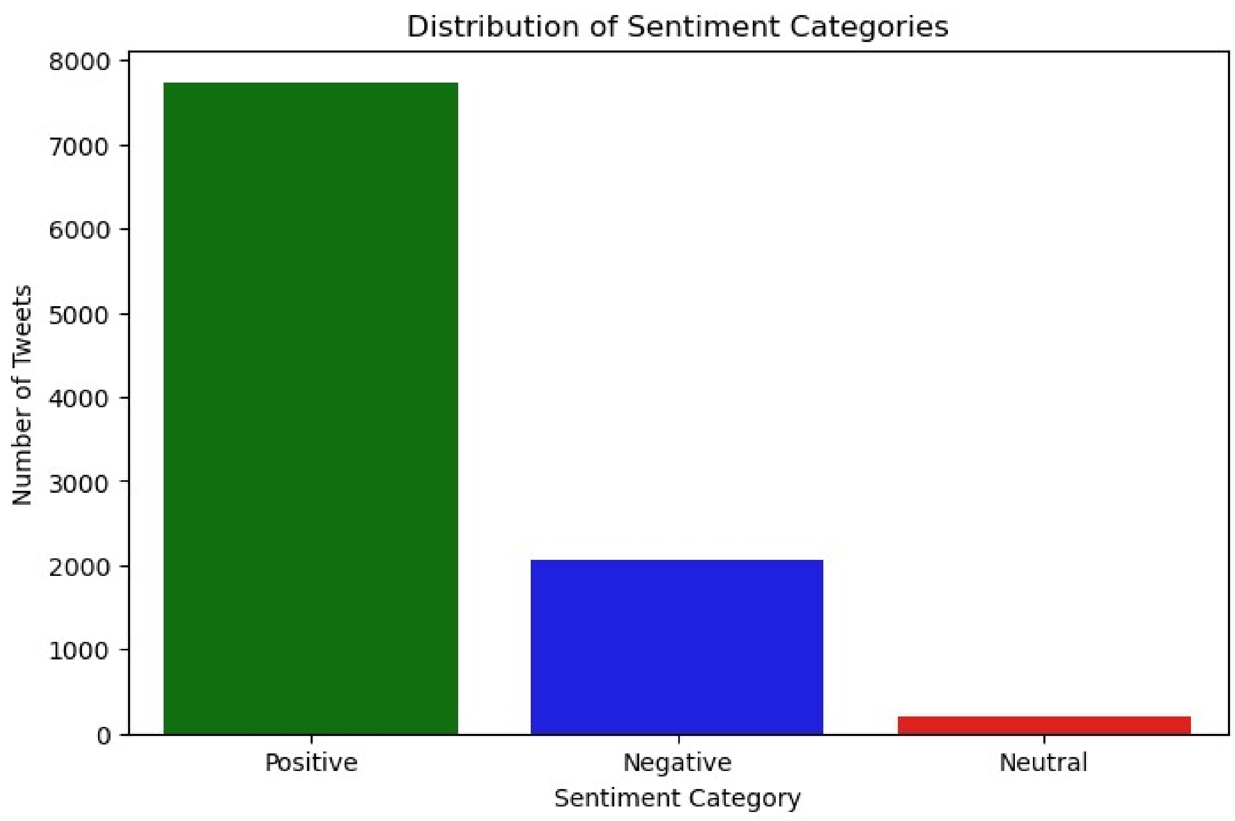
```
                                        Cleaned_Text  
Sentiment_Polarity  \
0  party least receive say or single prevent prev...            
0.115714   
1  hotel still congress may member staff media dr...            
0.308333   
2  nice be her debate industry that year film whe...            
0.220000   
3  laugh explain situation career occur serious f...            
0.054762   
4  involve sense former often approach government...            
0.033333   

  Sentiment_Category  
0           Positive  
1           Positive  
2           Positive  
3           Positive  
4           Positive  

import matplotlib.pyplot as plt
import seaborn as sns
```

Count the occurrences of each sentiment category

```
sentiment_counts = df['Sentiment_Category'].value_counts()
```

Plot the distribution of sentiment categories

```
plt.figure(figsize=(8, 5))
sns.barplot(x=sentiment_counts.index, y=sentiment_counts.values, 
palette={'red', 'blue', 'green'})
plt.title('Distribution of Sentiment Categories')
plt.xlabel('Sentiment Category')
plt.ylabel('Number of Tweets')
plt.savefig('Distribution of Sentiment Categories.png')
plt.show()

C:\Users\User\AppData\Local\Temp\ipykernel_13424\641027253.py:2: 
FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be 
removed in v0.14.0. Assign the `x` variable to `hue` and set 
`legend=False` for the same effect.

  sns.barplot(x=sentiment_counts.index, y=sentiment_counts.values, 
palette={'red', 'blue', 'green'})
```

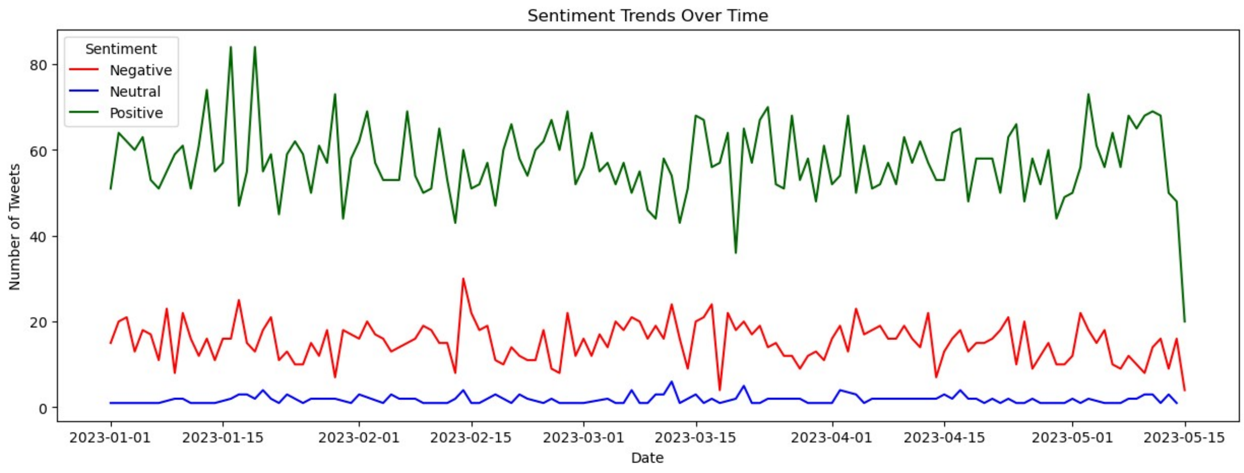


Grouping the data by date and sentiment category

```
df['Date'] = df['Timestamp'].dt.date
sentiment_trends = df.groupby(['Date',
'Sentiment_Category']).size().reset_index(name='Count')
```

Plot sentiment trends

```
palette = {
    'Positive': 'darkgreen',  # Positive gets green
    'Negative': 'red',       # Negative gets red
    'Neutral': 'blue'   # Neutral gets blue
}

plt.figure(figsize=(15, 5))
sns.lineplot(data=sentiment_trends, x='Date', y='Count',
hue='Sentiment_Category', palette=palette)
plt.title('Sentiment Trends Over Time')
plt.xlabel('Date')
plt.ylabel('Number of Tweets')
plt.legend(title='Sentiment')
plt.savefig('Sentiment Trends Over Time.png')
plt.show()
```



```
!pip install wordcloud
from wordcloud import WordCloud
```

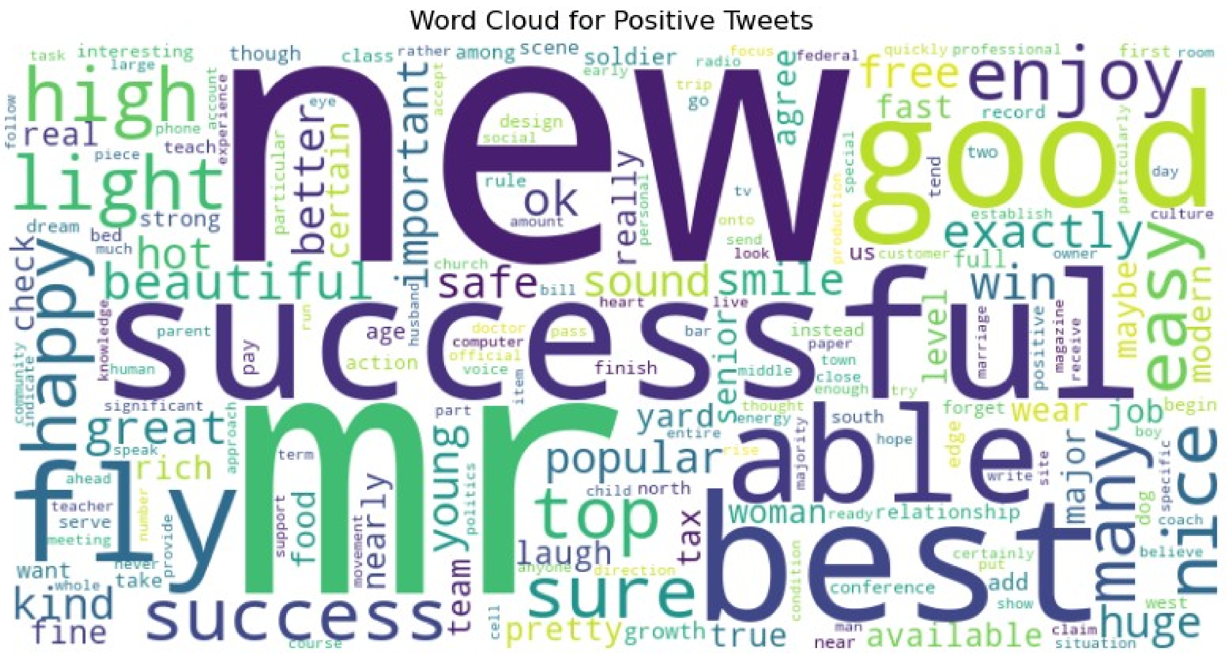
```
Requirement already satisfied: wordcloud in c:\users\user\anaconda3\
lib\site-packages (1.9.4)
Requirement already satisfied: numpy>=1.6.1 in c:\users\user\
anaconda3\lib\site-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in c:\users\user\anaconda3\lib\
site-packages (from wordcloud) (10.3.0)
Requirement already satisfied: matplotlib in c:\users\user\anaconda3\
lib\site-packages (from wordcloud) (3.8.4)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\user\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\user\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\user\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\user\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\user\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (23.2)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\user\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\user\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\users\user\anaconda3\
lib\site-packages (from python-dateutil>=2.7->matplotlib->wordcloud)
(1.16.0)
```

Function to create a word cloud for a specific sentiment

```
def create_wordcloud(sentiment):
    text = ' '.join(df[df['Sentiment_Category'] == sentiment]
['Cleaned_Text'])
    wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(text)
    plt.figure(figsize=(10, 5))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.title(f'Word Cloud for {sentiment} Tweets')
    plt.savefig(f'Word Cloud for {sentiment} Tweets.png')
    plt.show()
```

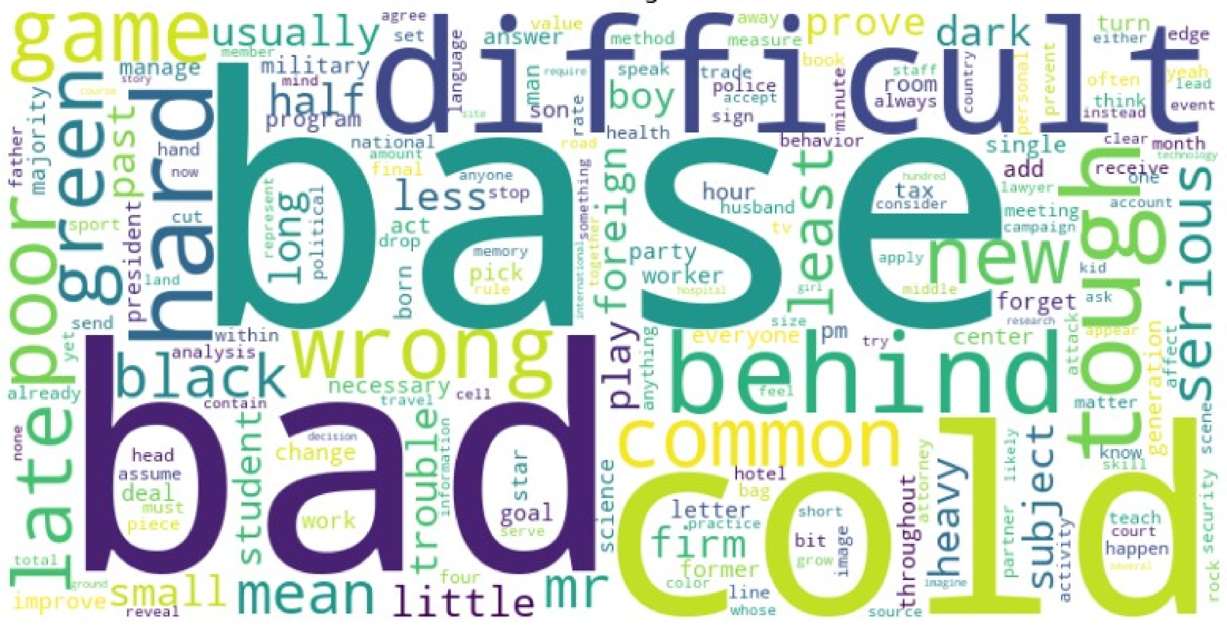
Generate word clouds

```
for sentiment in ['Positive']:
    create_wordcloud(sentiment)
```



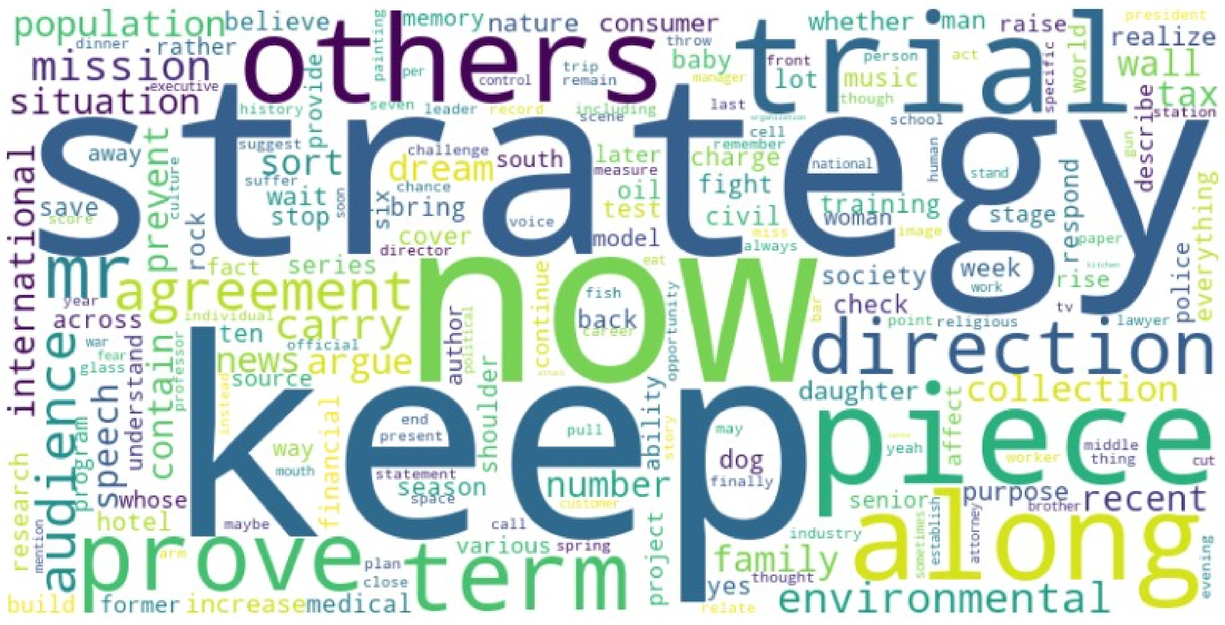
```
for sentiment in ['Negative']:
    create_wordcloud(sentiment)
```

Word Cloud for Negative Tweets



```python
for sentiment in ['Neutral']:
    create_wordcloud(sentiment)
```

Word Cloud for Neutral Tweets

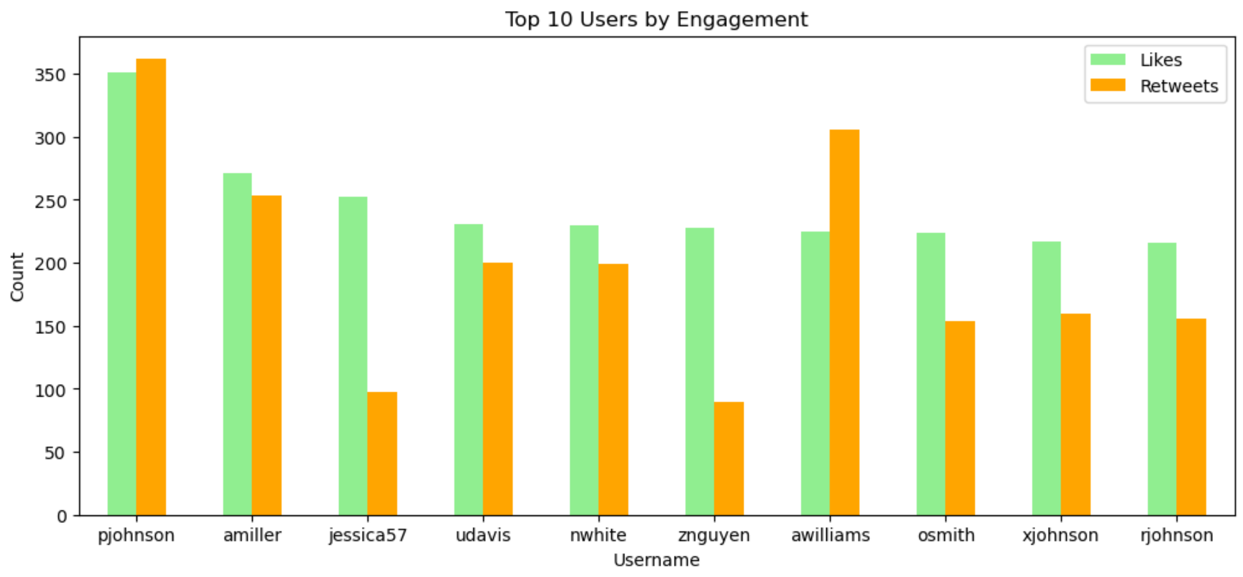


Aggregate likes and retweets by username

```python
top_users = df.groupby('Username')[['Likes',
'Retweets']].sum().sort_values(by='Likes', ascending=False).head(10)
```

Plot top users

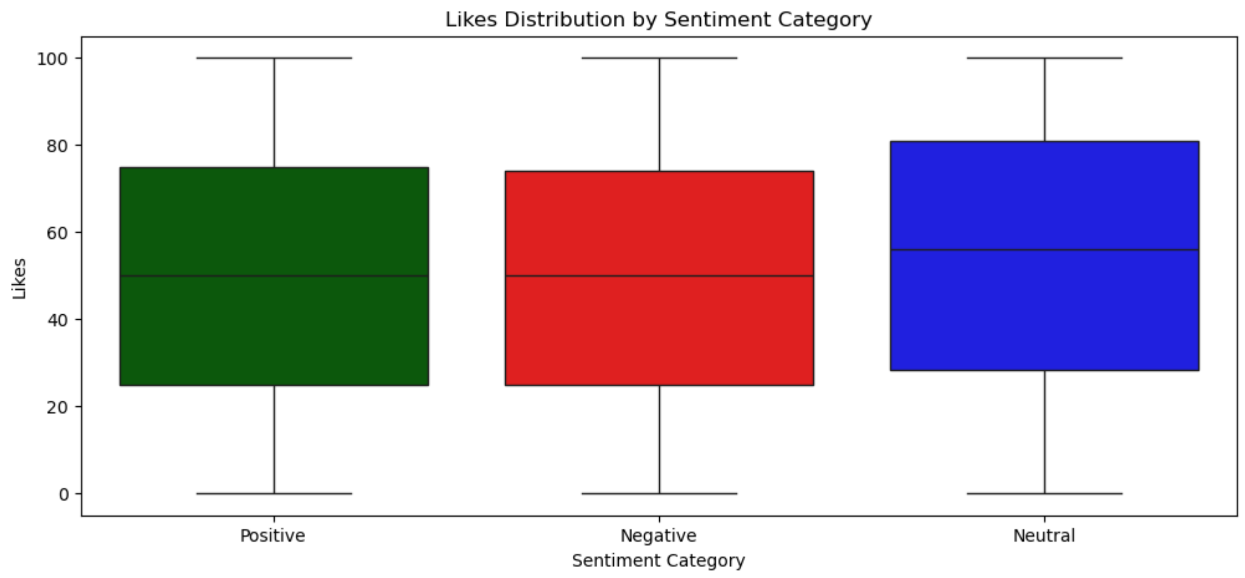
```
top_users.plot(kind='bar', figsize=(12, 5), color=['lightgreen',
'orange'])
plt.title('Top 10 Users by Engagement')
plt.xlabel('Username')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.savefig('Top 10 Users by Engagement.png')
plt.show()
```



Boxplot of Likes and Retweets by Sentiment Category

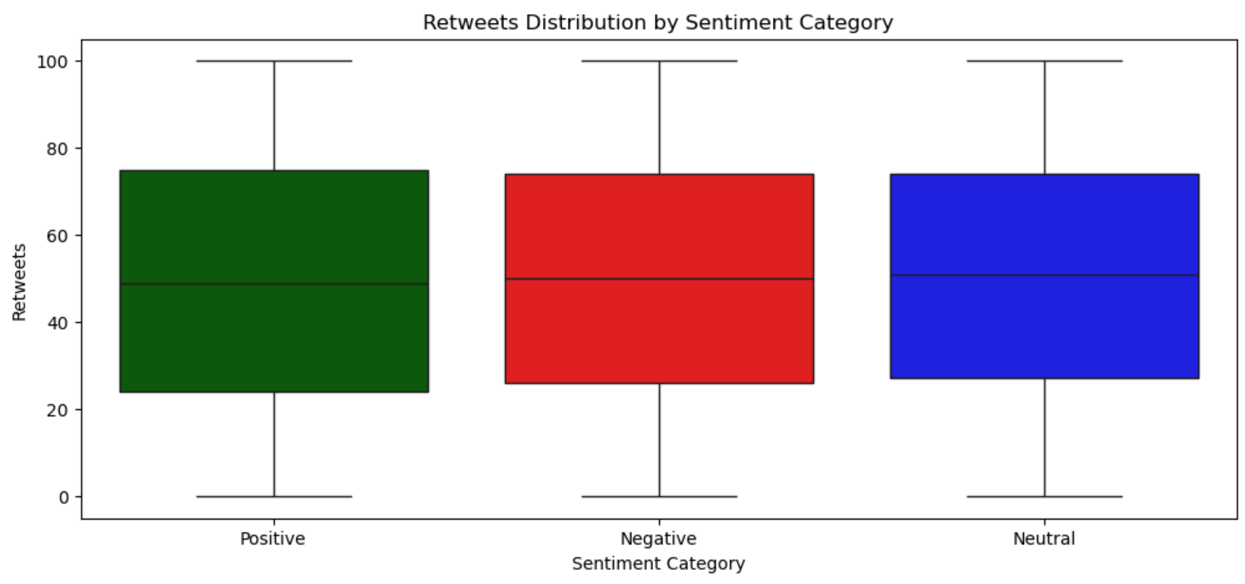
```
plt.figure(figsize=(12, 5))
sns.boxplot(data=df, x='Sentiment_Category', y='Likes',
palette=palette)
plt.title('Likes Distribution by Sentiment Category')
plt.xlabel('Sentiment Category')
plt.ylabel('Likes')
plt.savefig('Likes Distribution by Sentiment Category.png')
plt.show()
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_13424\631421549.py:2:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.boxplot(data=df, x='Sentiment_Category', y='Likes',
palette=palette)
```



```
plt.figure(figsize=(12, 5))
sns.boxplot(data=df, x='Sentiment_Category', y='Retweets',
palette=palette)
plt.title('Retweets Distribution by Sentiment Category')
plt.xlabel('Sentiment Category')
plt.ylabel('Retweets')
plt.savefig('Retweets Distribution by Sentiment Category.png')
plt.show()
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_13424\1465585623.py:2:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.boxplot(data=df, x='Sentiment_Category', y='Retweets',
palette=palette)
```

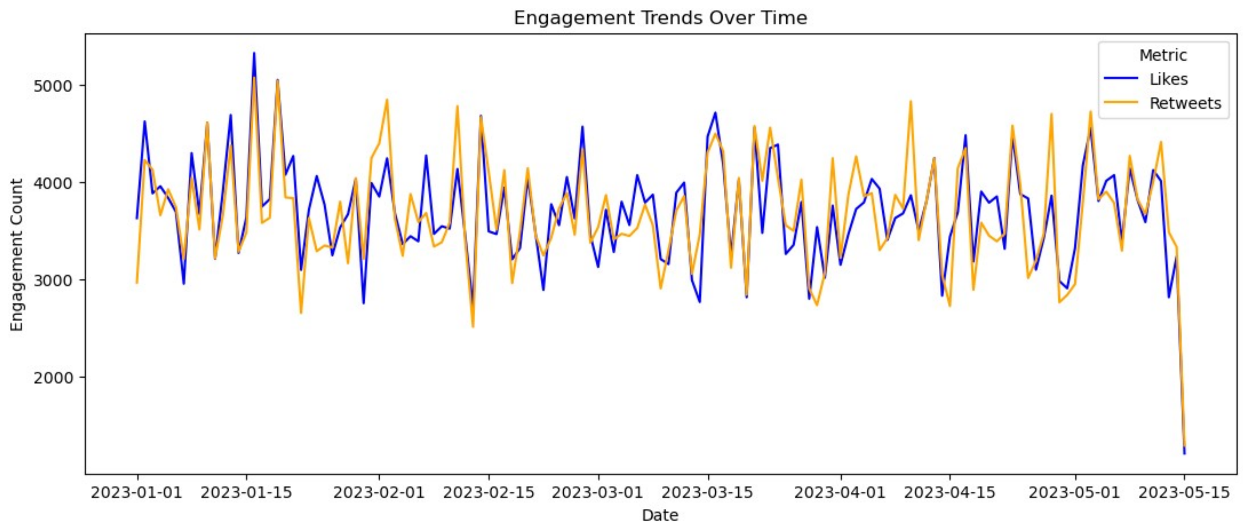


Aggregate likes and retweets by date

```
engagement_trends = df.groupby(df['Timestamp'].dt.date)[['Likes',
'Retweets']].sum().reset_index()
```

Plot engagement trends

```
plt.figure(figsize=(13, 5))
sns.lineplot(data=engagement_trends, x='Timestamp', y='Likes',
label='Likes', color='blue')
sns.lineplot(data=engagement_trends, x='Timestamp', y='Retweets',
label='Retweets', color='orange')
plt.title('Engagement Trends Over Time')
plt.xlabel('Date')
plt.ylabel('Engagement Count')
plt.legend(title='Metric')
plt.savefig('Engagement Trends Over Time.png')
plt.show()
```



Importing Counter

```
from collections import Counter
```

Combine all words in the Cleaned_Text column

```
all_words = ' '.join(df['Cleaned_Text']).split()
```

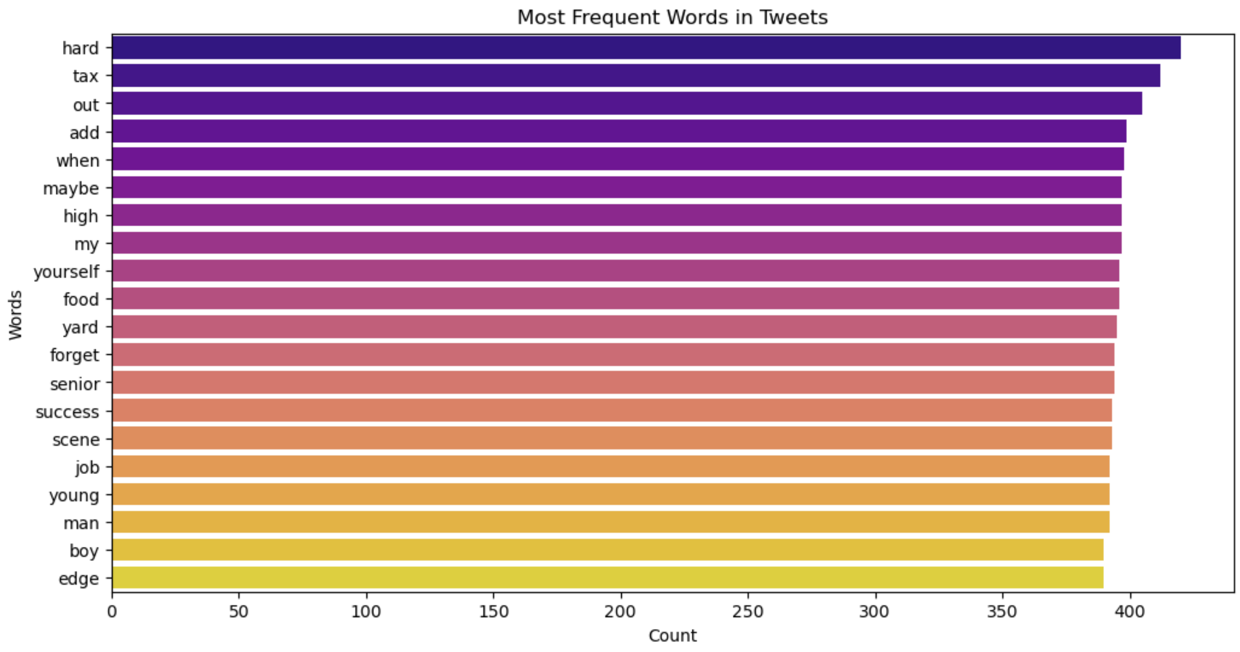
Count word frequencies

```
word_counts = Counter(all_words).most_common(20)
words, counts = zip(*word_counts)
```

Plot the most frequent words

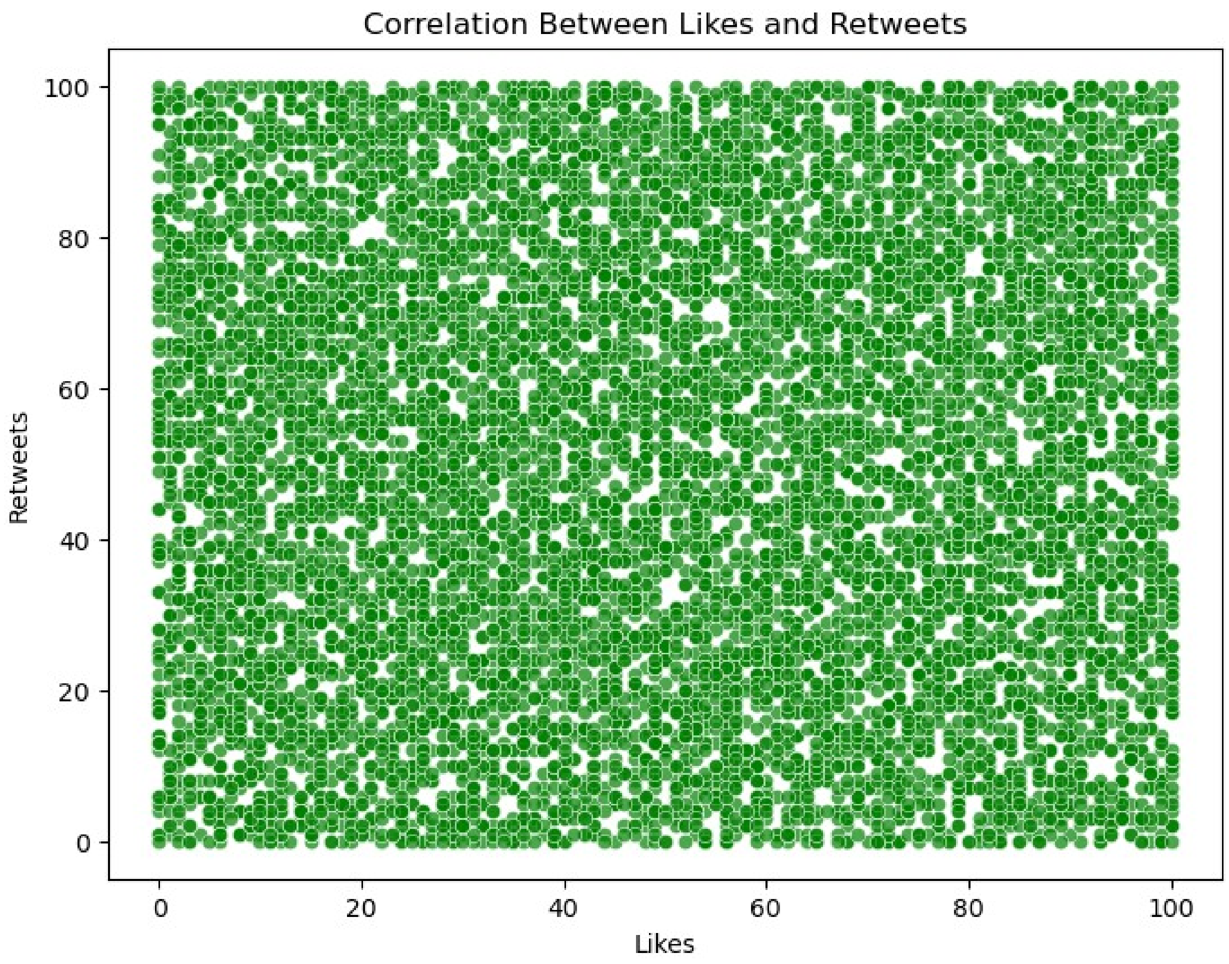
```
plt.figure(figsize=(12, 6))
sns.barplot(x=list(counts), y=list(words), palette='plasma')
plt.title('Most Frequent Words in Tweets')
plt.xlabel('Count')
plt.ylabel('Words')
plt.savefig('Most Frequent Words in Tweets.png')
plt.show()
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_13424\97703641.py:2:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `y` variable to `hue` and set
`legend=False` for the same effect.

  sns.barplot(x=list(counts), y=list(words), palette='plasma')
```



Scatter plot of Likes vs. Retweets

```
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='Likes', y='Retweets', alpha=0.7,
color='green')
plt.title('Correlation Between Likes and Retweets')
plt.xlabel('Likes')
plt.ylabel('Retweets')
plt.savefig('Correlation Between Likes and Retweets.png')
plt.show()
```



Calculate correlation coefficient

```
correlation = df[['Likes', 'Retweets']].corr()
print("Correlation between Likes and Retweets:\n", correlation)

Correlation between Likes and Retweets:
              Likes  Retweets
Likes     1.000000  0.012798
Retweets  0.012798  1.000000
```

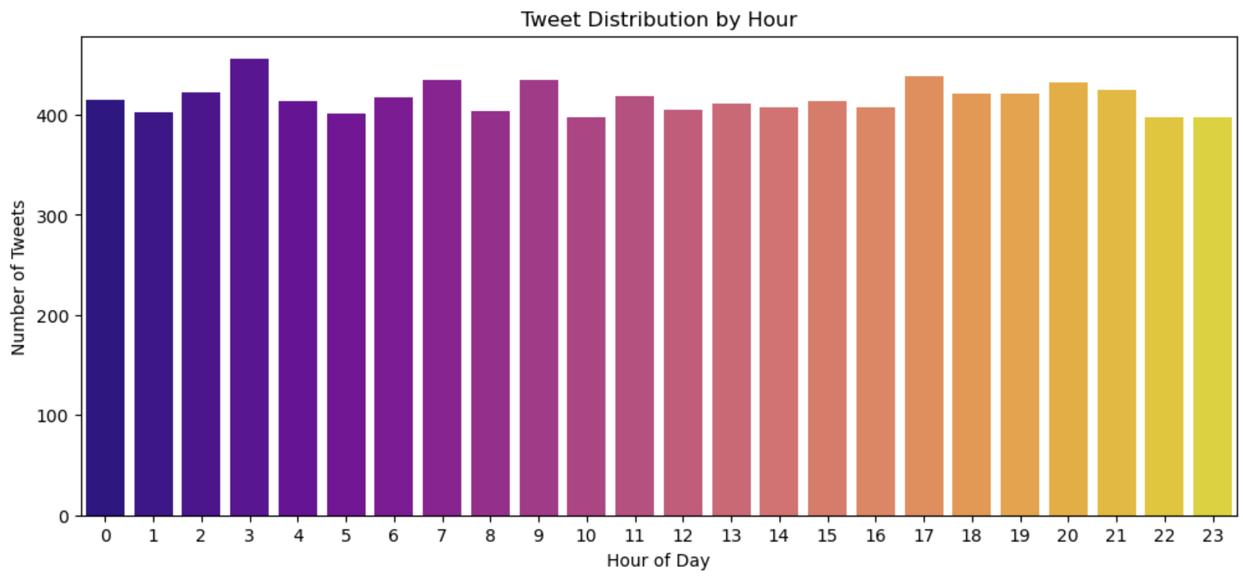
Extract hour from the Timestamp

```
df['Hour'] = df['Timestamp'].dt.hour
```

Plot the distribution of tweets by hour

```
plt.figure(figsize=(12, 5))
sns.countplot(x='Hour', data=df, palette='plasma')
plt.title('Tweet Distribution by Hour')
plt.xlabel('Hour of Day')
plt.ylabel('Number of Tweets')
plt.savefig('Tweet Distribution by Hour.png')
plt.show()
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_13424\4020942131.py:2:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.countplot(x='Hour', data=df, palette='plasma')
```



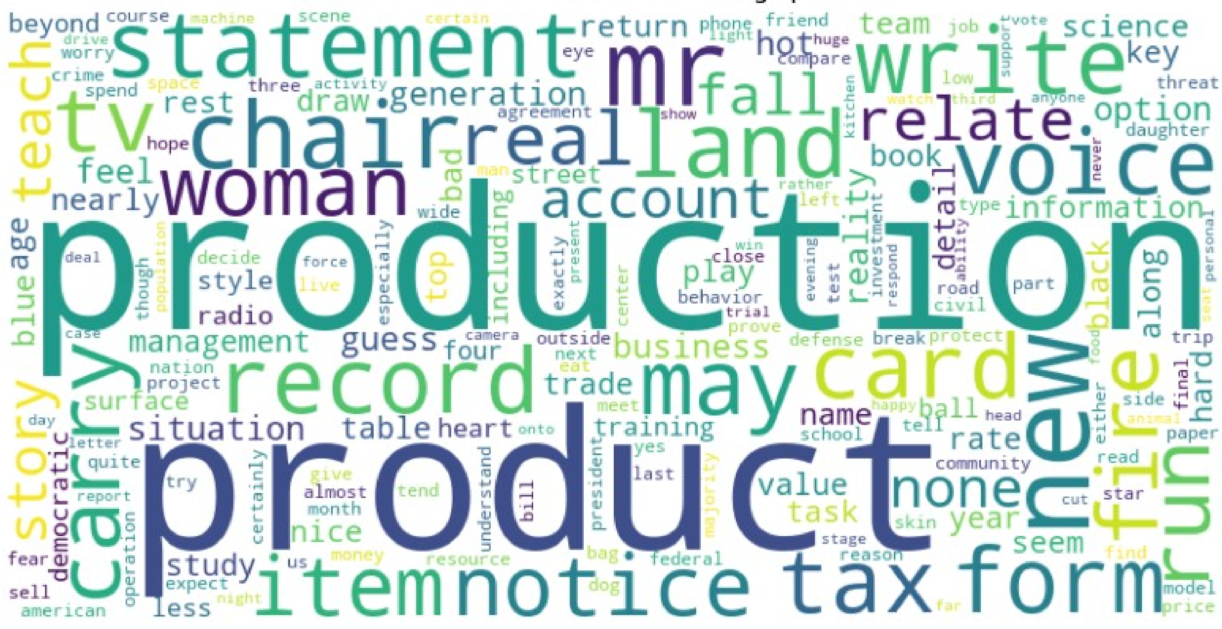
Filter tweets containing a specific keyword

```
keyword = 'product'  # Example keyword
filtered_tweets = ' '.join(df[df['Cleaned_Text'].str.contains(keyword,
na=False)]['Cleaned_Text'])
```

Generate word cloud for the filtered tweets

```
wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(filtered_tweets)
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title(f'Word Cloud for Tweets Containing "{keyword}"')
plt.savefig(f'Word Cloud for Tweets Containing {keyword}.png')
plt.show()
```

Word Cloud for Tweets Containing "product"


production
product
statement
mr
write
chair
real
land
voice
tv
woman
carry
record
may
card
new
fire
run
item
notice
tax
form
account
relate
fall
teach
story
none
information
generation
management
situation
business
science
option
team
return
guess
table
heart
training
value
task
year
seem
study
nice
reality
detail
black
along
hard
rate
name
ball
trade
four
surface
radio
style
play
top
including
bad
street
book
hot
key
draw
rest
feel
nearly
blue
age
beyond
democratic