Upload and load your CSV file

Start coding or generate with AI.

```
!pip install pandas matplotlib seaborn

#Import required libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import re
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.11/dist-packages (0.13.2)
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.58.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.2.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.3)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

```
#Upload the dataset
from google.colab import files
uploaded = files.upload()
```

Choose Files Social_Sen…nt_Data.csv

- **Social_Sentiment_Data.csv**(text/csv) - 160733 bytes, last modified: 5/27/2025 - 100% done

Saving Social_Sentiment_Data.csv to Social_Sentiment_Data.csv

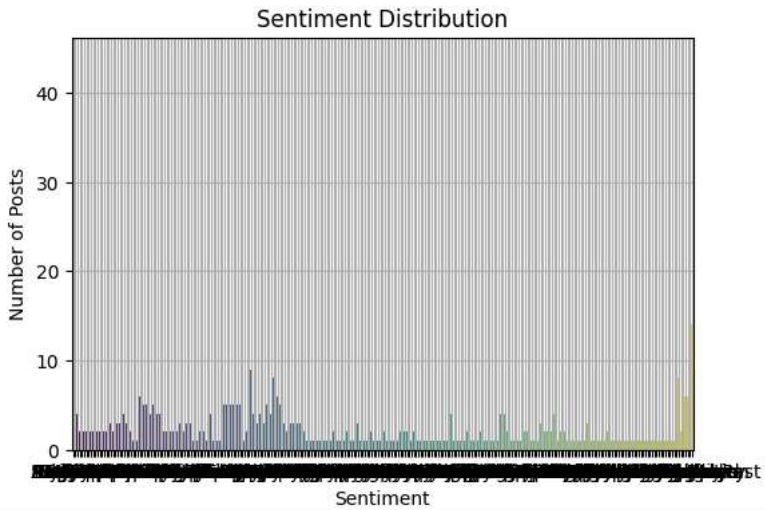
```
# Load the uploaded CSV
import io
df = pd.read_csv(io.BytesIO(uploaded['Social_Sentiment_Data.csv']))  # Replace with your actual filename
```

```
#Clean the text data
def clean_text(text):
    text = str(text).lower()
    text = re.sub(r"http\S+|www\S+|https\S+", '', text)  # Remove URLs
    text = re.sub(r'@\w+|\#', '', text)  # Remove mentions and hashtags
    text = re.sub(r'[^a-z\s]', '', text)  # Remove punctuation and numbers
    text = re.sub(r'\s+', ' ', text).strip()  # Remove extra whitespace
    return text
df['Clean_Text'] = df['Text'].apply(clean_text)
```

Visualization

```
#Visualize sentiment distribution
plt.figure(figsize=(6,4))
sns.countplot(data=df, x='Sentiment', palette='viridis')
plt.title('Sentiment Distribution')
plt.xlabel('Sentiment')
plt.ylabel('Number of Posts')
plt.grid(True)
plt.show()
```

```
<ipython-input-8-7d1ed7c7ad48>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.countplot(data=df, x='Sentiment', palette='viridis')
```

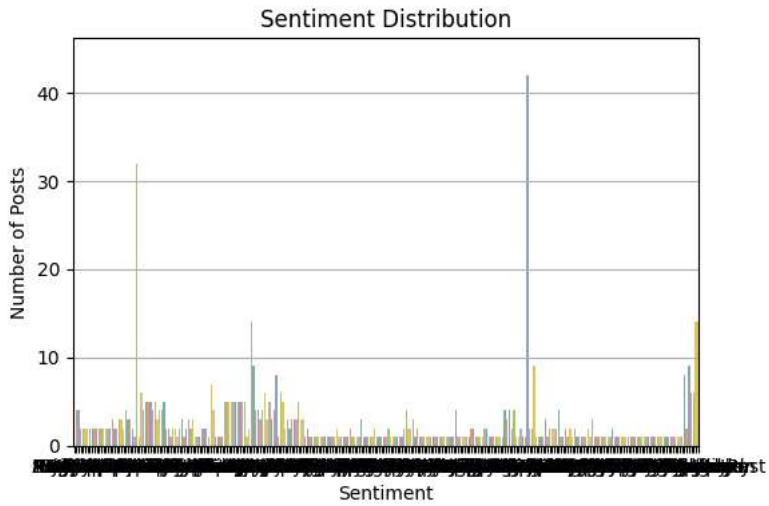



```
#Convert Timestamp to datetime
df['Timestamp'] = pd.to_datetime(df['Timestamp'], errors='coerce')
```

```
<ipython-input-9-c988f8c5ee63>:2: UserWarning: Parsing dates in %d-%m-%Y %H:%M format when dayfirst=False (the default) was specified. F
  df['Timestamp'] = pd.to_datetime(df['Timestamp'], errors='coerce')
```

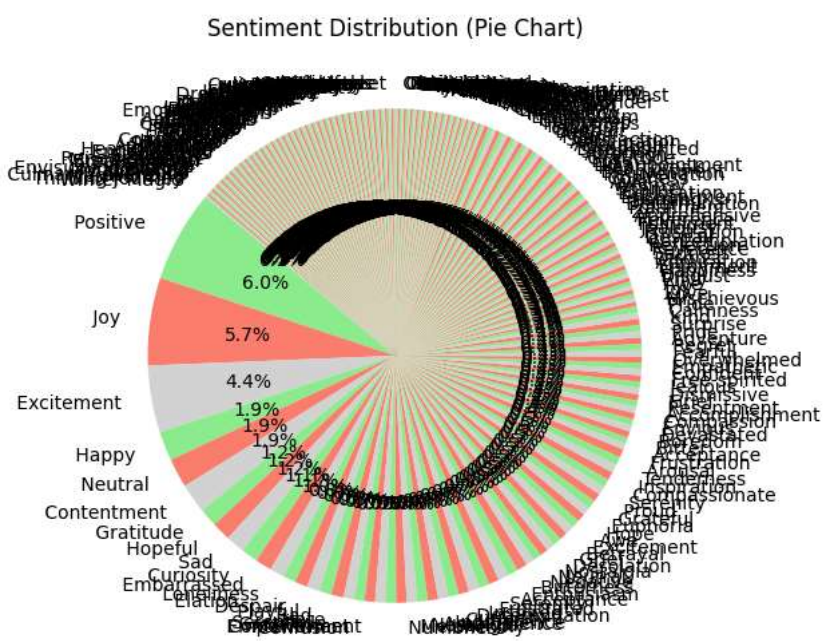
```
import matplotlib.pyplot as plt
import seaborn as sns

# Ensure Timestamp column is datetime
# Changed df_sentiment to df
df['Timestamp'] = pd.to_datetime(df['Timestamp'], errors='coerce')
# Changed df_sentiment to df
df['Date'] = df['Timestamp'].dt.date

# Plot 1: Sentiment Distribution
plt.figure(figsize=(6, 4))
# Changed df_sentiment to df and TextBlob_Sentiment to Sentiment (assuming the original 'Sentiment' column is what's intended)
sns.countplot(data=df, x='Sentiment', palette='Set2')
plt.title("Sentiment Distribution")
plt.xlabel("Sentiment")
plt.ylabel("Number of Posts")
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
<ipython-input-14-de4bd0839f09>:13: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.countplot(data=df, x='Sentiment', palette='Set2')
```



Sentiment Distribution Pie Chart

```
# Pie chart of sentiment distribution
# Changed df_sentiment['TextBlob_Sentiment'] to df['Sentiment']
df['Sentiment'].value_counts().plot.pie(
    autopct='%1.1f%%', figsize=(6, 6), startangle=140, colors=['lightgreen', 'salmon', 'lightgrey'],
    title='Sentiment Distribution (Pie Chart)')
plt.ylabel('')
plt.show()
```

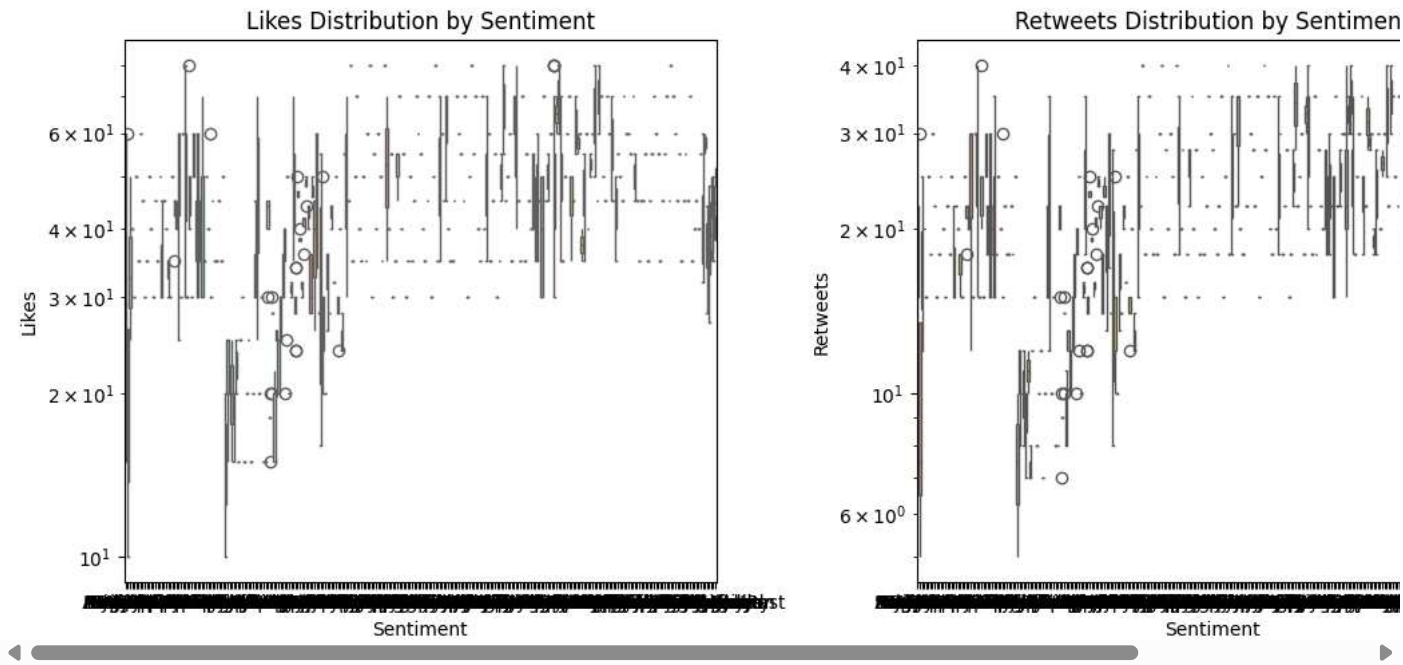


** Likes and Retweets by Sentiment**

```
# Likes and Retweets stats per sentiment
plt.figure(figsize=(12, 5))

# Likes
plt.subplot(1, 2, 1)
# Corrected df_sentiment to df and TextBlob_Sentiment to Sentiment
sns.boxplot(data=df, x='Sentiment', y='Likes', palette='Set3')
plt.title("Likes Distribution by Sentiment")
```

```
plt.yscale('log')  # Handle skewed data
plt.xlabel("Sentiment")
plt.ylabel("Likes")

# Retweets
plt.subplot(1, 2, 2)
# Corrected df_sentiment to df and TextBlob_Sentiment to Sentiment
sns.boxplot(data=df, x='Sentiment', y='Retweets', palette='Set2')
plt.title("Retweets Distribution by Sentiment")
plt.yscale('log')
plt.xlabel("Sentiment")
plt.ylabel("Retweets")

plt.tight_layout()
plt.show()
```

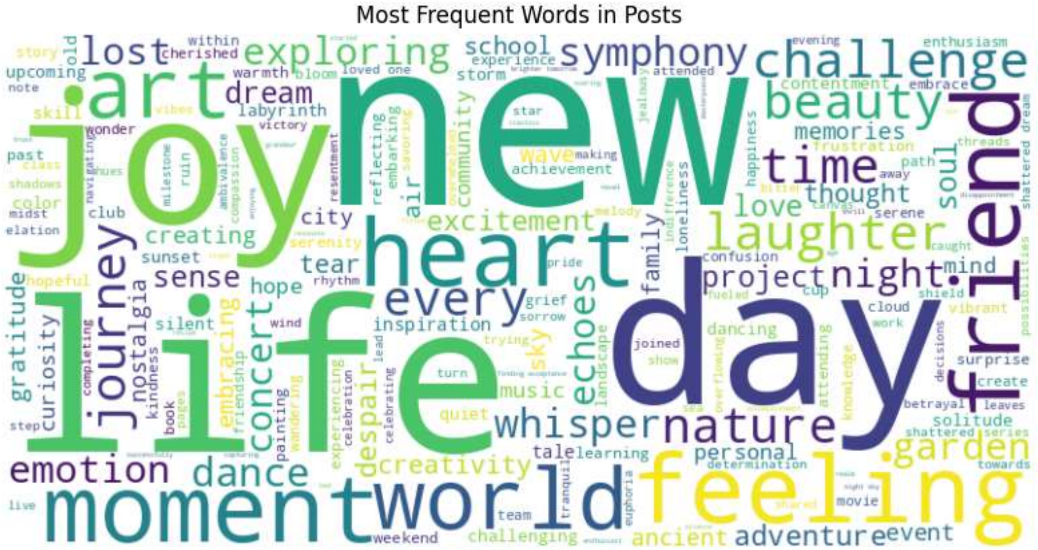
```
<ipython-input-24-2c6b404d2fef>:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.boxplot(data=df, x='Sentiment', y='Likes', palette='Set3')
<ipython-input-24-2c6b404d2fef>:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.boxplot(data=df, x='Sentiment', y='Retweets', palette='Set2')
```



**Word Cloud of Most Common Words **

```
!pip install wordcloud
from wordcloud import WordCloud

# Combine all cleaned text
# Changed df_sentiment to df as df_sentiment is not defined
all_words = ' '.join(df['Clean_Text'])

# Generate word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(all_words)

# Plot word cloud
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Most Frequent Words in Posts")
plt.show()
```

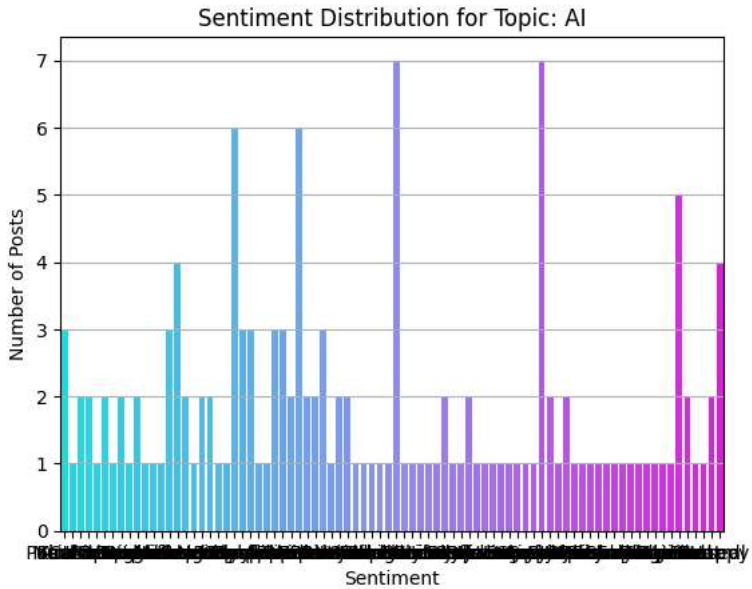
```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.11/dist-packages (1.9.4)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.11/dist-packages (from wordcloud) (2.0.2)
Requirement already satisfied: pillow in /usr/local/lib/python3.11/dist-packages (from wordcloud) (11.2.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (from wordcloud) (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->wordcloud) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib->wordcloud) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib->wordcloud) (4.58.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->wordcloud) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib->wordcloud) (24.2)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->wordcloud) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib->wordcloud) (2.9.0.post
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matplotlib->wordcloud) (1
```



Topic-Specific Filtering

```
# Example: Filter posts related to "AI"
topic_keyword = "AI"
# Corrected df_sentiment to df
df_topic = df[df['Clean_Text'].str.contains(topic_keyword.lower(), na=False)]

print(f"Total posts related to '{topic_keyword}':", df_topic.shape[0])

# Plot sentiment for filtered topic
# Corrected df_sentiment to df and TextBlob_Sentiment to Sentiment (assuming 'Sentiment' is the intended column)
sns.countplot(data=df_topic, x='Sentiment', palette='cool')
plt.title(f"Sentiment Distribution for Topic: {topic_keyword}")
plt.xlabel("Sentiment")
plt.ylabel("Number of Posts")
plt.grid(axis='y')
plt.show()
```

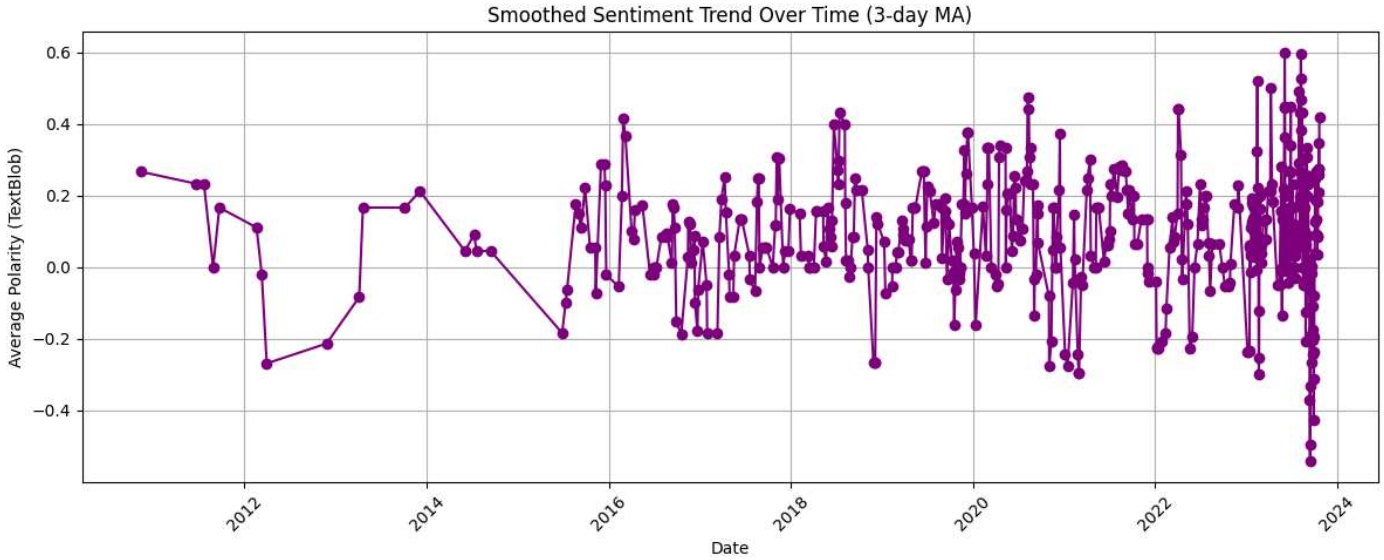
```
Total posts related to 'AI': 147
<ipython-input-32-4ef81bb08349>:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.countplot(data=df_topic, x='Sentiment', palette='cool')
```



Time Series Smoothing (Moving Average Sentiment)

```
# Install textblob if you haven't already
!pip install textblob

# Import TextBlob
from textblob import TextBlob

# Ensure Timestamp column is datetime
df['Timestamp'] = pd.to_datetime(df['Timestamp'], errors='coerce')

# Calculate polarity using TextBlob on the cleaned text
# The polarity attribute of a TextBlob object returns a float between -1.0 and 1.0
df['Polarity'] = df['Clean_Text'].apply(lambda text: TextBlob(text).sentiment.polarity)

# Sort by timestamp and compute daily average polarity
# Changed df_sentiment to df
daily_avg = df.groupby(df['Timestamp'].dt.date)['Polarity'].mean().rolling(window=3).mean()

# Plot smoothed sentiment over time
plt.figure(figsize=(12, 5))
plt.plot(daily_avg.index, daily_avg.values, marker='o', linestyle='-', color='purple')
plt.title("Smoothed Sentiment Trend Over Time (3-day MA)")
plt.xlabel("Date")
plt.ylabel("Average Polarity (TextBlob)")
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()
```

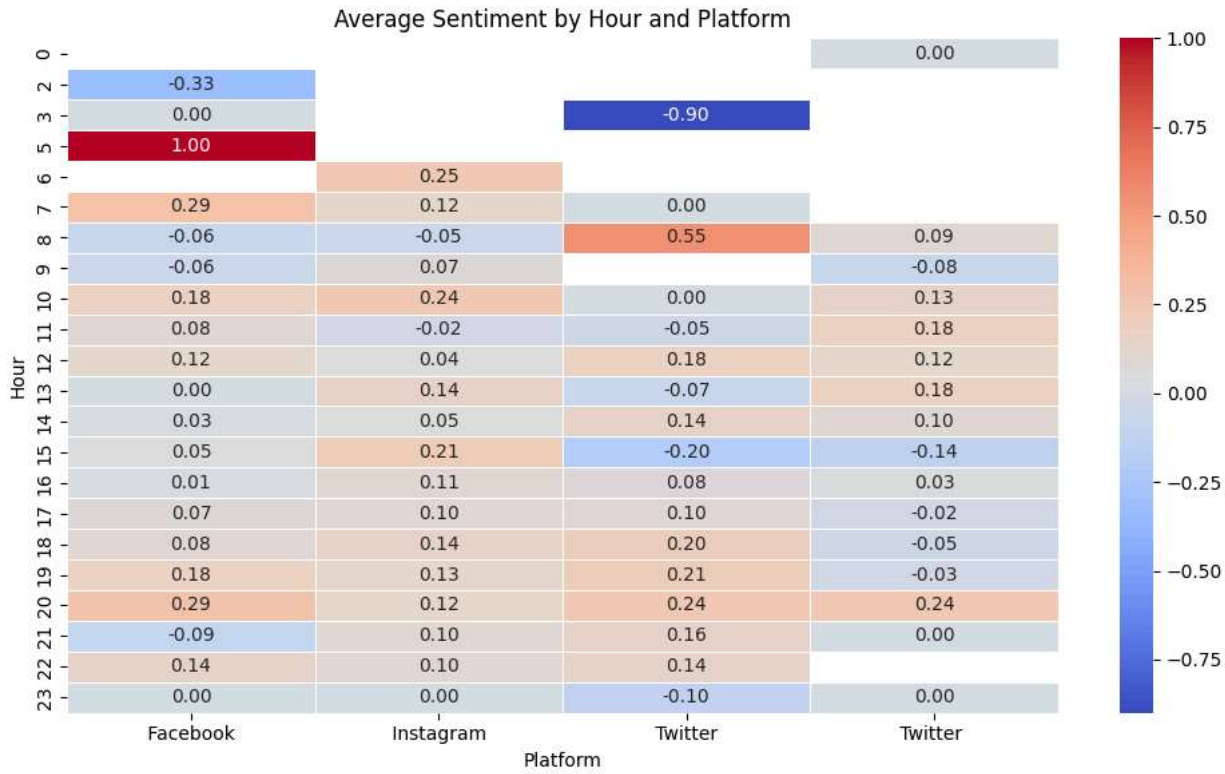
```
Requirement already satisfied: textblob in /usr/local/lib/python3.11/dist-packages (0.19.0)
Requirement already satisfied: nltk>=3.9 in /usr/local/lib/python3.11/dist-packages (from textblob) (3.9.1)
Requirement already satisfied: click in /usr/local/lib/python3.11/dist-packages (from nltk>=3.9->textblob) (8.2.0)
Requirement already satisfied: joblib in /usr/local/lib/python3.11/dist-packages (from nltk>=3.9->textblob) (1.5.0)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.11/dist-packages (from nltk>=3.9->textblob) (2024.11.6)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from nltk>=3.9->textblob) (4.67.1)
```



Sentiment by Platform

Heatmap: Sentiment by Hour & Platform

```
# Create the 'Hour' column if it doesn't exist
if 'Hour' not in df.columns:
    df['Hour'] = df['Timestamp'].dt.hour

# Create pivot table using the correct DataFrame 'df'
pivot = pd.pivot_table(df, index='Hour', columns='Platform',
                       values='Polarity', aggfunc='mean')

# Plot heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(pivot, cmap='coolwarm', annot=True, fmt=".2f", linewidths=0.5)
plt.title("Average Sentiment by Hour and Platform")
plt.xlabel("Platform")
plt.ylabel("Hour")
plt.tight_layout()
plt.show()
```

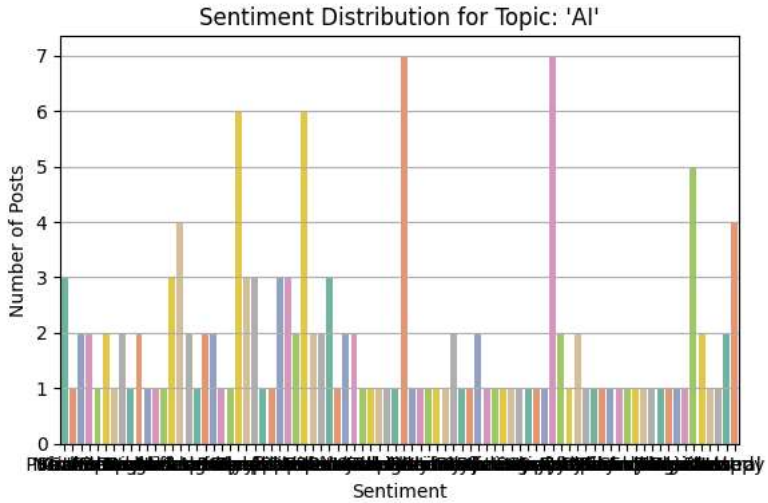


Topic-Specific Filtering and Sentiment Visualization

```
# 📝 Choose your topic keyword
topic_keyword = "AI"  # Change this to any topic like "covid", "election", "budget", etc.

# 🔍 Filter posts containing the keyword in cleaned text
df_topic = df_sentiment[df_sentiment['Clean_Text'].str.contains(topic_keyword.lower(), na=False)]

# 📊 Print number of related posts
print(f"Total posts related to '{topic_keyword}':", df_topic.shape[0])

# 📉 Plot sentiment distribution for filtered topic
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(6, 4))
sns.countplot(data=df_topic, x='TextBlob_Sentiment', palette='Set2')
plt.title(f"Sentiment Distribution for Topic: '{topic_keyword}'")
plt.xlabel("Sentiment")
plt.ylabel("Number of Posts")
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
Total posts related to 'AI': 147
<ipython-input-39-c2038eec90d0>:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc
```

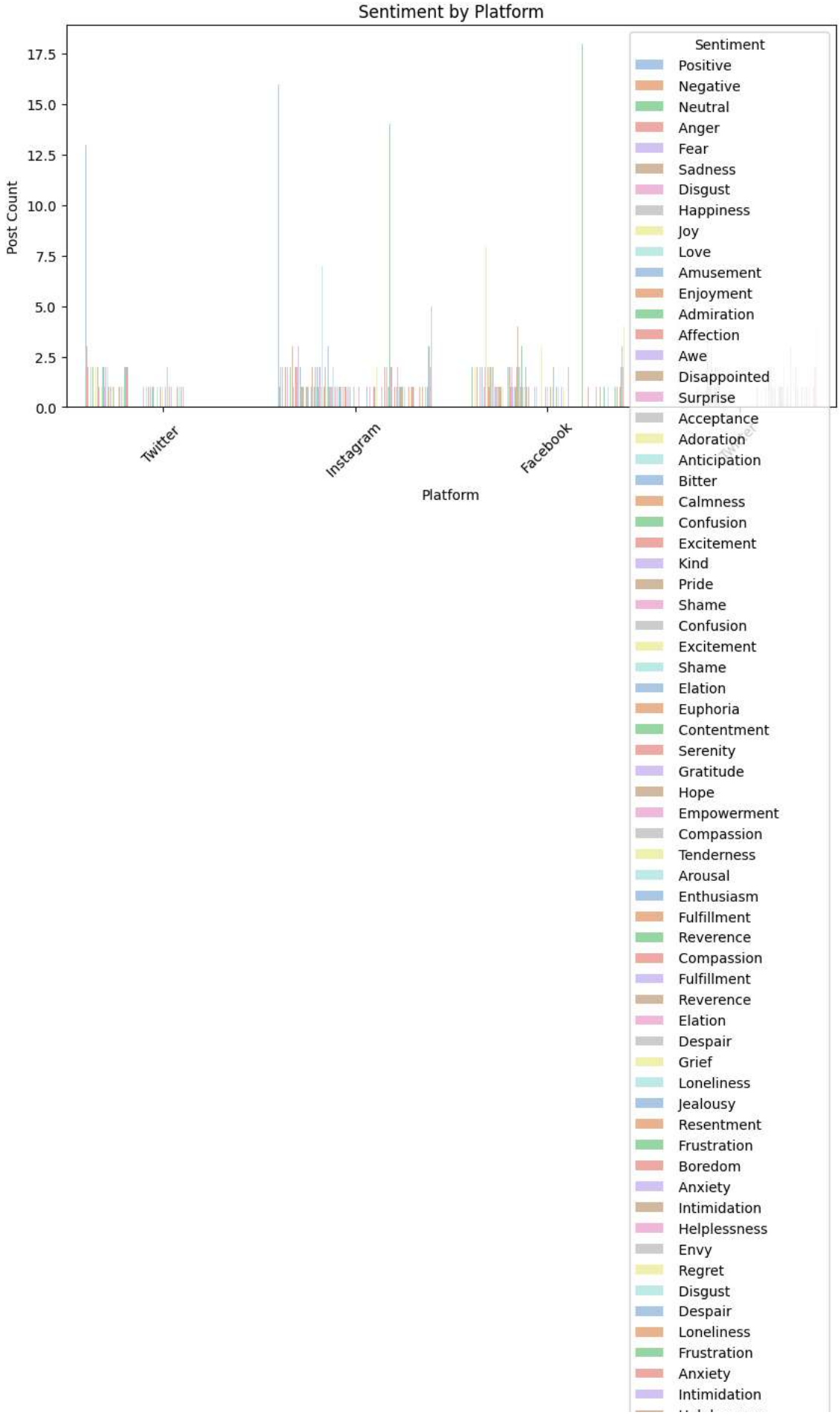
```
sns.countplot(data=df_topic, x='Sentiment', palette='Set2')
```



sentiment by platform

```
# Plot sentiment distribution across social platforms
plt.figure(figsize=(10, 5))
# Changed df_sentiment to df and TextBlob_Sentiment to Sentiment
sns.countplot(data=df, x='Platform', hue='Sentiment', palette='pastel')
plt.title("Sentiment by Platform")
plt.xlabel("Platform")
plt.ylabel("Post Count")
plt.legend(title="Sentiment")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
<ipython-input-22-dacb49008041>:10: UserWarning: Tight layout not applied. The bottom and top margins cannot be made large enough to
  plt.tight_layout()
```



Helplessness
Jealousy
Curiosity
Indifference
Confusion
Numbness
Melancholy
Nostalgia
Ambivalence
Acceptance
Determination
Serenity
Numbness
Zest
Contentment
Hopeful
Proud
Grateful
Empathetic
Compassionate
Playful
Free-spirited
Inspired
Confident
Serenity
Curiosity
Ambivalence
Despair
Bitterness
Yearning
Fearful
Apprehensive
Overwhelmed
Jealous
Devastated
Frustrated
Envious
Dismissive
Awe
Determination
Nostalgia
Thrill
Calmness
Overwhelmed
Gratitude
Bittersweet
Curiosity
Admiration
Overjoyed
Inspiration
Motivation
Amusement
Contemplation
JoyfulReunion
Excitement
Satisfaction
Blessed
Anticipation
Reflection
Nostalgia
Appreciation
Confidence
Surprise
Accomplishment
Wonderment
Optimism
Pride
Happiness
Curiosity
Enchantment
Intrigue