```
    Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
    To https://github.com/Sydneyhelsel/COMM158_final_Helsel_Barrientos.git
       453e467..27f95c0  main -> main
```

```
# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import seaborn as sns
import re
from collections import Counter
import nltk
from nltk.tokenize import word_tokenize

        #1.1
# Load CSV files
trump_tweets = pd.read_csv('data/trump_encoded.csv')
clinton_tweets = pd.read_csv('data/clinton_encoded.csv')
# Add candidate column
trump_tweets['candidate'] = 'Donald Trump'
clinton_tweets['candidate'] = 'Hillary Clinton'
# Combine DataFrames
combined_df = pd.concat([trump_tweets, clinton_tweets], ignore_index=True)
# Display the first few rows to verify
combined_df.head()
    #1.2 (first downloading the lexicon and cleaning data)
!wget https://raw.githubusercontent.com/aditeyabaral/lok-sabha-election-twitter-analysis/master/NRC-Emotion-Lexicon-Wordlevel-v0.92.

nrc_lexicon = pd.read_csv("NRC-Emotion-Lexicon-Wordlevel-v0.92.txt", sep="\t", header=None, names=["word", "emotion", "association"]
nrc_lexicon = nrc_lexicon[nrc_lexicon["association"] == 1].drop(columns=["association"])
print(nrc_lexicon.head())

# Function to clean text
def clean_text(text):
    # Remove mentions (@username) and hashtags (#hashtag)
    text = re.sub(r'@[\w]+', '', text)  # Remove mentions
    text = re.sub(r'#[\w]+', '', text)  # Remove hashtags
    # Remove non-alphabetical characters (punctuation, numbers, etc.)
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Only keep alphabets and spaces
    # Remove extra spaces
    text = re.sub(r'\s+', ' ', text).strip()
    return text

# Download the 'punkt' tokenizer and 'punkt_tab'
nltk.download('punkt')
nltk.download('punkt_tab')
# 1.2.1 Create a Sentiment Analysis Function
    # Fix the sentiment analysis function and apply it to cleaned text
def get_sentiment_counts(tweet_text, lexicon):
    # Tokenize the tweet text into words
    tokens = word_tokenize(str(tweet_text).lower())  # Lowercasing for matching

    # Initialize a dictionary to store sentiment counts
    sentiment_counts = {emotion: 0 for emotion in lexicon['emotion'].unique()}

    # Check each token in the tweet text
    for token in tokens:
        # Check if the token is in the NRC Emotion Lexicon
        emotion_matches = lexicon[lexicon['word'] == token]
        # For each matching emotion, increment the corresponding count
        for _, row in emotion_matches.iterrows():
            sentiment_counts[row['emotion']] += 1  # Increment the count for the emotion

    return sentiment_counts

# 1.3.1 Group by candidate and generate summary statistics for each emotion category
# Load data
trump_tweets = pd.read_csv('data/trump_encoded.csv')
clinton_tweets = pd.read_csv('data/clinton_encoded.csv')
# Add candidate column
trump_tweets['candidate'] = 'Donald Trump'
clinton_tweets['candidate'] = 'Hillary Clinton'
# Combine DataFrames
combined_df = pd.concat([trump_tweets, clinton_tweets], ignore_index=True)
# Clean text and perform sentiment analysis
def clean_text(text):
    # Remove mentions (@username) and hashtags (#hashtag)
    text = re.sub(r'@[\w]+', '', text)  # Remove mentions
```

```
    text = re.sub(r'#[\w]+', '', text)  # Remove hashtags
    # Remove non-alphabetical characters (punctuation, numbers, etc.)
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Only keep alphabets and spaces
    # Remove extra spaces
    text = re.sub(r'\s+', ' ', text).strip().lower()
    return text
# Load lexicon
nrc_lexicon = pd.read_csv("NRC-Emotion-Lexicon-Wordlevel-v0.92.txt",
                         sep="\t",
                         header=None,
                         names=["word", "emotion", "association"])
nrc_lexicon = nrc_lexicon[nrc_lexicon["association"] == 1].drop(columns=["association"])
def get_sentiment_counts(tweet_text, lexicon):
    # Clean the text
    cleaned_text = clean_text(str(tweet_text))

    # Use simple split as the TA confirmed is acceptable
    words = cleaned_text.split()

    # Initialize counts for each emotion
    sentiment_counts = {emotion: 0 for emotion in lexicon['emotion'].unique()}

    # Count emotions for each word, allowing duplicates
    for word in words:
        # Find matching emotions for this word
        matches = lexicon[lexicon['word'] == word]
        for _, row in matches.iterrows():
            sentiment_counts[row['emotion']] += 1

    return sentiment_counts
# First, create a cleaned text column
combined_df['cleaned_text'] = combined_df['text'].apply(clean_text)
# Apply the sentiment analysis function to cleaned text
sentiment_counts = combined_df['cleaned_text'].apply(lambda tweet: get_sentiment_counts(tweet, nrc_lexicon))
# Convert the dictionary of sentiment counts into separate columns
sentiment_df = pd.DataFrame(list(sentiment_counts))
# Merge the sentiment columns with the original DataFrame
merged_df_with_sentiments = pd.concat([combined_df, sentiment_df], axis=1)
# Define emotion categories
emotion_categories = ['anger', 'anticipation', 'disgust', 'fear', 'joy',
                     'sadness', 'surprise', 'trust', 'positive', 'negative']
# 1.3.1 Group by candidate and generate summary statistics for each emotion
print("1.3.1 Summary statistics for emotions by candidate:")
emotion_stats_by_candidate = merged_df_with_sentiments.groupby('candidate')[emotion_categories].describe()
print(emotion_stats_by_candidate)

# 1.3.2 Group by candidate and each emotion category, then compute summary statistics
# Load data
trump_tweets = pd.read_csv('data/trump_encoded.csv')
clinton_tweets = pd.read_csv('data/clinton_encoded.csv')
# Add candidate column
trump_tweets['candidate'] = 'Donald Trump'
clinton_tweets['candidate'] = 'Hillary Clinton'
# Combine DataFrames
combined_df = pd.concat([trump_tweets, clinton_tweets], ignore_index=True)
# Clean text function
def clean_text(text):
    # Remove mentions (@username) and hashtags (#hashtag)
    text = re.sub(r'@[\w]+', '', text)  # Remove mentions
    text = re.sub(r'#[\w]+', '', text)  # Remove hashtags
    # Remove non-alphabetical characters (punctuation, numbers, etc.)
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Only keep alphabets and spaces
    # Remove extra spaces
    text = re.sub(r'\s+', ' ', text).strip().lower()
    return text
# Load lexicon
nrc_lexicon = pd.read_csv("NRC-Emotion-Lexicon-Wordlevel-v0.92.txt",
                         sep="\t",
                         header=None,
                         names=["word", "emotion", "association"])
nrc_lexicon = nrc_lexicon[nrc_lexicon["association"] == 1].drop(columns=["association"])
def get_sentiment_counts(tweet_text, lexicon):
    # Clean the text
    cleaned_text = clean_text(str(tweet_text))

    # Use simple split as the TA confirmed is acceptable
    words = cleaned_text.split()

    # Initialize counts for each emotion
```

```
        sentiment_counts = {emotion: 0 for emotion in lexicon['emotion'].unique()}

        # Count emotions for each word, allowing duplicates
        for word in words:
            # Find matching emotions for this word
            matches = lexicon[lexicon['word'] == word]
            for _, row in matches.iterrows():
                sentiment_counts[row['emotion']] += 1

        return sentiment_counts
    # First, create a cleaned text column
    combined_df['cleaned_text'] = combined_df['text'].apply(clean_text)
    # Apply the sentiment analysis function to cleaned text
    sentiment_counts = combined_df['cleaned_text'].apply(lambda tweet: get_sentiment_counts(tweet, nrc_lexicon))
    # Convert the dictionary of sentiment counts into separate columns
    sentiment_df = pd.DataFrame(list(sentiment_counts))
    # Merge the sentiment columns with the original DataFrame
    merged_df_with_sentiments = pd.concat([combined_df, sentiment_df], axis=1)
    # List of all emotion categories
    emotion_categories = ['anger', 'anticipation', 'disgust', 'fear', 'joy',
                          'sadness', 'surprise', 'trust', 'positive', 'negative']
    # 1.3.2 Group by candidate and emotion category, compute summary stats for engagement
    print("1.3.2 Summary statistics for engagement metrics by candidate and emotion:")
    engagement_metrics = ['favorite_count', 'retweet_count']
    # Analyze all emotions, but highlight two selected ones for deeper analysis
    for emotion in emotion_categories:
        # Create bins for emotion counts
        merged_df_with_sentiments[f'{emotion}_level'] = pd.cut(
            merged_df_with_sentiments[emotion],
            bins=[-1, 0, 2, float('inf')],
            labels=['None', 'Low', 'High']
        )

        # Group by candidate and emotion level, calculate engagement stats
        stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
        print(f"\nEngagement metrics for {emotion}:")
        print(stats)
    # Based on the analysis, select two emotions for visualization in 1.4
    selected_emotions = ['fear', 'disgust']
    print(f"\nSelected emotions for visualization in 1.4: {selected_emotions}")
    print("1. Fear: Selected because it showed significant differences between candidates' tweets.")
    print("2. Disgust: Selected as a complementary negative emotion that reveals interesting patterns in engagement metrics.")

    # Part 2: Correlation Analysis
    # 2.1 Research Question: Is there a relationship between emotion categories
    # and the number of retweets or favorites?
    # This code assumes that parts 1.1-1.4 have been run first
    # and that the merged_df_with_sentiments variable exists
    # Define emotion categories and engagement metrics
    emotion_categories = ['anger', 'anticipation', 'disgust', 'fear', 'joy',
                          'sadness', 'surprise', 'trust', 'positive', 'negative']
    engagement_metrics = ['favorite_count', 'retweet_count']
    # Calculate correlation between emotion counts and engagement metrics
    print("\nResearch Question: Is there a relationship between emotion categories and engagement?")
    correlation_df = merged_df_with_sentiments[emotion_categories + engagement_metrics].corr()
    # Extract the correlations between emotions and engagement metrics
    emotion_engagement_corr = correlation_df.loc[emotion_categories, engagement_metrics]
    print("\nCorrelation between emotions and engagement metrics:")
    print(emotion_engagement_corr)
    # Create a heatmap to visualize the correlations
    plt.figure(figsize=(10, 8))
    sns.heatmap(emotion_engagement_corr, annot=True, cmap='coolwarm', vmin=-0.2, vmax=0.2)
    plt.title('Correlation between Emotions and Engagement Metrics', fontsize=16)
    plt.tight_layout()
    plt.savefig('output/emotion_engagement_correlation.png')
    # Analyze correlations separately for each candidate
    fig, axes = plt.subplots(1, 2, figsize=(20, 8))
    candidates = merged_df_with_sentiments['candidate'].unique()
    for i, candidate in enumerate(candidates):
        # Filter data for this candidate
        candidate_df = merged_df_with_sentiments[merged_df_with_sentiments['candidate'] == candidate]

        # Calculate correlations
        candidate_corr = candidate_df[emotion_categories + engagement_metrics].corr()
        candidate_emotion_engagement = candidate_corr.loc[emotion_categories, engagement_metrics]

        # Plot heatmap
        sns.heatmap(candidate_emotion_engagement,
                    annot=True,
```

```
                cmap='coolwarm',
                vmin=-0.2,
                vmax=0.2,
                ax=axes[i])
    axes[i].set_title(f'Correlations for {candidate}', fontsize=14)
plt.tight_layout()
plt.savefig('output/emotion_engagement_correlation_by_candidate.png')
"""
The heatmap analysis reveals slight relationships between emotional expressions in tweets and engagement, specifically favorite and
counts. Tweets expressing emotions such as disgust and negativity are slightly slightly correlated with higher engagement, suggestin
people may respond more actively to emotionally charged content, specifically negatively charged. Tweets characterized by anticipati
surprise, or positive emotions exhibit minor negative correlations, implying these emotions might be marginally less effective at dr
engagement. Emotions such as anger, fear, sadness, and trust don't seem to have an impact on engagement. Overall, while these correl
are small, they indicate that tweets with a negative or intense sentiment, particularly with emotions of disgust, may be somewhat mo
effective in eliciting likes and retweets.
"""

# 1.4 Visualizing Results
# Load data
trump_tweets = pd.read_csv('data/trump_encoded.csv')
clinton_tweets = pd.read_csv('data/clinton_encoded.csv')

# Add candidate column
trump_tweets['candidate'] = 'Donald Trump'
clinton_tweets['candidate'] = 'Hillary Clinton'

# Combine DataFrames
combined_df = pd.concat([trump_tweets, clinton_tweets], ignore_index=True)

# Clean text function
def clean_text(text):
    # Remove mentions (@username) and hashtags (#hashtag)
    text = re.sub(r'@[\w]+', '', text)  # Remove mentions
    text = re.sub(r'#[\w]+', '', text)  # Remove hashtags
    # Remove non-alphabetical characters (punctuation, numbers, etc.)
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Only keep alphabets and spaces
    # Remove extra spaces
    text = re.sub(r'\s+', ' ', text).strip().lower()
    return text

# Load lexicon
nrc_lexicon = pd.read_csv("NRC-Emotion-Lexicon-Wordlevel-v0.92.txt",
                         sep="\t",
                         header=None,
                         names=["word", "emotion", "association"])
nrc_lexicon = nrc_lexicon[nrc_lexicon["association"] == 1].drop(columns=["association"])

def get_sentiment_counts(tweet_text, lexicon):
    # Clean the text
    cleaned_text = clean_text(str(tweet_text))

    # Use simple split as the TA confirmed is acceptable
    words = cleaned_text.split()

    # Initialize counts for each emotion
    sentiment_counts = {emotion: 0 for emotion in lexicon['emotion'].unique()}

    # Count emotions for each word, allowing duplicates
    for word in words:
        # Find matching emotions for this word
        matches = lexicon[lexicon['word'] == word]
        for _, row in matches.iterrows():
            sentiment_counts[row['emotion']] += 1

    return sentiment_counts

# First, create a cleaned text column
combined_df['cleaned_text'] = combined_df['text'].apply(clean_text)

# Apply the sentiment analysis function to cleaned text
sentiment_counts = combined_df['cleaned_text'].apply(lambda tweet: get_sentiment_counts(tweet, nrc_lexicon))

# Convert the dictionary of sentiment counts into separate columns
sentiment_df = pd.DataFrame(list(sentiment_counts))

# Merge the sentiment columns with the original DataFrame
merged_df_with_sentiments = pd.concat([combined_df, sentiment_df], axis=1)

# Selected emotions for visualization based on 1.3.1 and 1.3.2 analysis
```

```
selected_emotions = ['fear', 'disgust']
print(f"Visualizing engagement metrics for selected emotions: {selected_emotions}")

# Create a figure with 2 subplots
fig, axes = plt.subplots(1, 2, figsize=(18, 6))

# Define explicit colors for candidates
clinton_color = 'blue'
trump_color = 'red'

# Group the data by candidate and calculate total engagement metrics for tweets with these emotions
engagement_by_candidate = {}

for emotion in selected_emotions:
    # Create empty DataFrames to store results
    likes_data = pd.DataFrame(columns=['candidate', 'count'])
    retweets_data = pd.DataFrame(columns=['candidate', 'count'])

    # Calculate engagement metrics for each candidate
    for candidate in merged_df_with_sentiments['candidate'].unique():
        # Get tweets for this candidate that have this emotion
        candidate_tweets = merged_df_with_sentiments[merged_df_with_sentiments['candidate'] == candidate]
        emotion_tweets = candidate_tweets[candidate_tweets[emotion] > 0]

        # Calculate total likes and retweets
        total_likes = emotion_tweets['favorite_count'].sum()
        total_retweets = emotion_tweets['retweet_count'].sum()

        # Store in DataFrames
        likes_data = pd.concat([likes_data, pd.DataFrame({'candidate': [candidate], 'count': [total_likes]})], ignore_index=True)
        retweets_data = pd.concat([retweets_data, pd.DataFrame({'candidate': [candidate], 'count': [total_retweets]})], ignore_index

    # Store data for this emotion
    engagement_by_candidate[emotion] = {
        'likes': likes_data,
        'retweets': retweets_data
    }

# Plot likes for selected emotions
x_pos = np.arange(len(selected_emotions))
width = 0.35

# Plot likes for each candidate
clinton_likes = [engagement_by_candidate[emotion]['likes'][engagement_by_candidate[emotion]['likes']['candidate'] == 'Hillary Clinto
trump_likes = [engagement_by_candidate[emotion]['likes'][engagement_by_candidate[emotion]['likes']['candidate'] == 'Donald Trump']['

axes[0].bar(x_pos - width/2, clinton_likes, width, label='Hillary Clinton', color=clinton_color)
axes[0].bar(x_pos + width/2, trump_likes, width, label='Donald Trump', color=trump_color)
axes[0].set_title('Total Likes by Emotion Category', fontsize=14)
axes[0].set_ylabel('Total Likes', fontsize=12)
axes[0].set_xticks(x_pos)
axes[0].set_xticklabels(selected_emotions)
axes[0].legend()

# Plot retweets for each candidate
clinton_retweets = [engagement_by_candidate[emotion]['retweets'][engagement_by_candidate[emotion]['retweets']['candidate'] == 'Hilla
trump_retweets = [engagement_by_candidate[emotion]['retweets'][engagement_by_candidate[emotion]['retweets']['candidate'] == 'Donald

axes[1].bar(x_pos - width/2, clinton_retweets, width, label='Hillary Clinton', color=clinton_color)
axes[1].bar(x_pos + width/2, trump_retweets, width, label='Donald Trump', color=trump_color)
axes[1].set_title('Total Retweets by Emotion Category', fontsize=14)
axes[1].set_ylabel('Total Retweets', fontsize=12)
axes[1].set_xticks(x_pos)
axes[1].set_xticklabels(selected_emotions)
axes[1].legend()

# Adjust layout and save
plt.tight_layout()
plt.savefig('output/emotion_engagement_comparison.png')
print("Visualization completed and saved to output/emotion_engagement_comparison.png")

# Part 3: Open Ended Exploration - Hashtag Analysis
# This exploration examines the hashtags used by Trump and Clinton during the election,
# testing the hypothesis that Trump used more actionable and negative hashtags than Clinton,
# and analyzing the emotional content and engagement of tweets containing these hashtags.

# Research question: Did Trump use more actionable and negative hashtags than Clinton?
# Analysis method: Frequency analysis of hashtags, categorization of hashtag types,
# emotion profile analysis, and engagement metrics comparison.
```

```
# Note:
# - Fixed 'mcincle' to 'rncincle' in hashtag extraction
# - Combined 'maga' and 'makeamericagreatagain' as one hashtag theme
# - Combined 'votetrump' and 'trump2016' as one hashtag theme
# - Categorized hashtags as 'Event-related', 'Action-oriented', 'Negative/Attack', or 'Other'

# Extract hashtags from original tweets
def extract_hashtags(text):
    # Use regex to find all hashtags
    hashtags = re.findall(r'#(\w+)', str(text))

    # Normalize specific hashtags
    normalized = []
    for tag in hashtags:
        tag = tag.lower()
        # Fix mcincle to rncincle
        if tag == 'mcincle':
            normalized.append('rncincle')
        else:
            normalized.append(tag)

    return normalized

# Normalize and combine similar hashtags
def normalize_hashtags(hashtags):
    normalized = []
    for tag in hashtags:
        # Combine MAGA-related hashtags
        if tag in ['maga', 'makeamericagreatagain']:
            normalized.append('maga/makeamericagreatagain')
        # Combine Trump campaign hashtags
        elif tag in ['votetrump', 'trump2016']:
            normalized.append('votetrump/trump2016')
        else:
            normalized.append(tag)
    return normalized

# Function to get hashtag emotion data - for combined hashtags
def get_hashtag_emotions(df, hashtag):
    # For combined hashtags, split and check for either one
    if '/' in hashtag:
        tag1, tag2 = hashtag.split('/')
        mask = (df['text'].str.contains(f'#{tag1}', case=False, na=False) |
                df['text'].str.contains(f'#{tag2}', case=False, na=False))
    else:
        mask = df['text'].str.contains(f'#{hashtag}', case=False, na=False)

    hashtag_tweets = df[mask]

    if len(hashtag_tweets) == 0:
        return None

    # Calculate average emotion scores
    emotion_scores = hashtag_tweets[emotion_categories].mean()

    # Calculate average engagement
    engagement = {
        'favorite_count': hashtag_tweets['favorite_count'].mean(),
        'retweet_count': hashtag_tweets['retweet_count'].mean(),
        'tweet_count': len(hashtag_tweets)
    }

    return pd.Series({**emotion_scores, **engagement})

# Categorize hashtags
def categorize_hashtag(tag):
    if tag in event_hashtags:
        return 'Event-related'
    elif tag in action_hashtags:
        return 'Action-oriented'
    elif tag in negative_hashtags:
        return 'Negative/Attack'
    else:
        return 'Other'

# Analysis:
"""
Though It would've been intriguing to see whether or not there was any interaction between engagement levels and the expected outcom
```

```
"""

##############################################
# === Part 3 Exploration #1 (Paolo) ===
##############################################

# This exploration examines the hashtags used by Trump and Clinton during the election,
# testing the hypothesis that Trump used more actionable and negative hashtags than Clinton,
# and analyzing the emotional content and engagement of tweets containing these hashtags.

# Research question: Did Trump use more actionable and negative hashtags than Clinton?
# Analysis method: Frequency analysis of hashtags, categorization of hashtag types,
# emotion profile analysis, and engagement metrics comparison.

# Note:
# - Fixed 'mcincle' to 'rncincle' in hashtag extraction
# - Combined 'maga' and 'makeamericagreatagain' as one hashtag theme
# - Combined 'votetrump' and 'trump2016' as one hashtag theme
# - Categorized hashtags as 'Event-related', 'Action-oriented', 'Negative/Attack', or 'Other'

# Extract hashtags from original tweets
def extract_hashtags(text):
    # Use regex to find all hashtags
    hashtags = re.findall(r'#(\w+)', str(text))

    # Normalize specific hashtags
    normalized = []
    for tag in hashtags:
        tag = tag.lower()
        # Fix mcincle to rncincle
        if tag == 'mcincle':
            normalized.append('rncincle')
        else:
            normalized.append(tag)

    return normalized

# Normalize and combine similar hashtags
def normalize_hashtags(hashtags):
    normalized = []
    for tag in hashtags:
        # Combine MAGA-related hashtags
        if tag in ['maga', 'makeamericagreatagain']:
            normalized.append('maga/makeamericagreatagain')
        # Combine Trump campaign hashtags
        elif tag in ['votetrump', 'trump2016']:
            normalized.append('votetrump/trump2016')
        else:
            normalized.append(tag)
    return normalized

# Function to get hashtag emotion data - for combined hashtags
def get_hashtag_emotions(df, hashtag):
    # For combined hashtags, split and check for either one
    if '/' in hashtag:
        tag1, tag2 = hashtag.split('/')
        mask = (df['text'].str.contains(f'#{tag1}', case=False, na=False) |
                df['text'].str.contains(f'#{tag2}', case=False, na=False))
    else:
        mask = df['text'].str.contains(f'#{hashtag}', case=False, na=False)

    hashtag_tweets = df[mask]

    if len(hashtag_tweets) == 0:
        return None

    # Calculate average emotion scores
    emotion_scores = hashtag_tweets[emotion_categories].mean()

    # Calculate average engagement
    engagement = {
        'favorite_count': hashtag_tweets['favorite_count'].mean(),
        'retweet_count': hashtag_tweets['retweet_count'].mean(),
        'tweet_count': len(hashtag_tweets)
    }

    return pd.Series({**emotion_scores, **engagement})

# Categorize hashtags
```

```
def categorize_hashtag(tag):
    if tag in event_hashtags:
        return 'Event-related'
    elif tag in action_hashtags:
        return 'Action-oriented'
    elif tag in negative_hashtags:
        return 'Negative/Attack'
    else:
        return 'Other'

# Analysis:
"""
Though It would've been intriguing to see whether or not there was any interaction between engagement levels and the expected outcom
"""

##############################################
# === Part 3 Exploration #2 (Sydney) ===
##############################################
"""
# The previous exploration looked at engagement with certain hashtags associated with Trump and Clinton during the 2016 election.
# To further understand the amount of engagement with these hashtags, I decided to make a graph that visualizes the engagement with
# top five hashtags over time. This way, we can see when certain hashtags were created, and when they were most popular.

# My research question is: Do certain hashtags have more engagement during certain weeks of the election?
# This could give insights into more specific questions, such as:  is there more engagement with hashtags relating to debates during

# My hypothesis is that engagement with hashtags will fluctuate over time, with greater variance in engagement for hashtags associat
# with Hillary Clinton compared to those associated with Donald Trump. This is because Clinton's top hashtags are more event-driven,
# leading to spikes and drops in engagement during certain times, while Trump's hashtags function more as slogans and maintain more

# Analysis method: I created a time-series graph that visualizes the engagement of the five most popular hashtags over time,
# highlighting when each hashtag emerged and peaked.

# The visualization validates my hypothesis, confirming that hashtags vary in engagement
# over the course of the election, and specifically that Trump's hashtags seem to be more popular for longer periods of time.
# For example the orange line on the graph for the Trump tweets, which was for the hashtag "#Makeamericagreatagain" spans for nearly
# entire period of time that the data examined. Notably, it is his campaign slogan. Similarly, the "#Imwithher" hashtag from the Hil
# represented by the purple line, which was the most similar to a slogan out of her top five hashtags, has the longest period of eng
# out of the five. Another insight we can make from this visualization is that engagement as a whole with certain hashtags spiked in
# and early November for both candidates. This makes sense, due to the proximity to the election. I also had it confirm the sentimen
# or negative) of the hashtag to see if there was a difference, but it turns out the most popular hashtags were all positive.

#The code for the time series graph is below:
"""

##############################################
# === Load Data ===
##############################################
trump_df = pd.read_csv('data/trump_encoded.csv')
clinton_df = pd.read_csv('data/clinton_encoded.csv')
trump_df['candidate'] = 'Donald Trump'
clinton_df['candidate'] = 'Hillary Clinton'
df = pd.concat([trump_df, clinton_df], ignore_index=True)

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

# Calculate engagement for each tweet
df['engagement'] = df['favorite_count'] + df['retweet_count']

##############################################
# Use hashtagged tweets with sentiment analysis
##############################################

# --- Clean text function (already in your code) ---
def clean_text(text):
    text = re.sub(r'@[\w]+', '', text)
    text = re.sub(r'#[\w]+', '', text)
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    text = re.sub(r'\s+', ' ', text).strip().lower()
    return text

# --- Load NRC Emotion Lexicon ---
nrc = pd.read_csv('NRC-Emotion-Lexicon-Wordlevel-v0.92.txt', sep="\t", header=None,
                  names=["word", "emotion", "association"])
nrc = nrc[nrc['association'] == 1].drop(columns=['association'])

# --- Sentiment Function (raw counts) ---
def get_sentiment_counts(text, lexicon):
```

```
    tokens = word_tokenize(text)
    counts = {emotion: 0 for emotion in lexicon['emotion'].unique()}
    for token in tokens:
        matches = lexicon[lexicon['word'] == token]
        for _, row in matches.iterrows():
            counts[row['emotion']] += 1
    return counts

# --- Filter hashtagged tweets ---
df['has_hashtag'] = df['text'].str.contains(r'#\w+')
hashtagged = df[df['has_hashtag']].copy()

# --- Clean and apply sentiment analysis ---
hashtagged['cleaned_text'] = hashtagged['text'].apply(clean_text)
sentiment_counts = hashtagged['cleaned_text'].apply(lambda t: get_sentiment_counts(t, nrc))
sentiment_df = pd.DataFrame(list(sentiment_counts))
# Merge sentiment results back into hashtagged tweets
hashtagged_sentiment = pd.concat([hashtagged.reset_index(drop=True), sentiment_df.reset_index(drop=True)], axis=1)

# Compute net sentiment as positive - negative
hashtagged_sentiment['net_sentiment'] = hashtagged_sentiment['positive'] - hashtagged_sentiment['negative']

# Create a week column (Period) for aggregation
hashtagged_sentiment['week'] = hashtagged_sentiment['created_at'].dt.to_period('W')
hashtagged_sentiment['week_start'] = hashtagged_sentiment['week'].dt.start_time

##############################################
# Extract hashtags and determine top 5 per candidate
##############################################

def extract_hashtags(text):
    tags = re.findall(r'#\w+', text)
    return [tag.lower() for tag in tags]

hashtagged_sentiment['hashtags'] = hashtagged_sentiment['text'].apply(extract_hashtags)

# Explode the hashtags so each hashtag gets its own row
hs_exploded = hashtagged_sentiment.explode('hashtags')
# Remove rows with no hashtag
hs_exploded = hs_exploded[hs_exploded['hashtags'].notna()]

# Determine top 5 hashtags for each candidate by frequency
top_hashtags = {}
for candidate in hs_exploded['candidate'].unique():
    candidate_df = hs_exploded[hs_exploded['candidate'] == candidate]
    top5 = candidate_df['hashtags'].value_counts().head(5).index.tolist()
    top_hashtags[candidate] = top5

print("Top 5 Hashtags per Candidate:", top_hashtags)

# Filter hs_exploded to include only the top hashtags for each candidate
filtered_hs = hs_exploded[hs_exploded.apply(lambda row: row['hashtags'] in top_hashtags[row['candidate']], axis=1)]

##############################################
# Aggregate Engagement & Sentiment by Candidate, Hashtag, and Week
##############################################

agg = filtered_hs.groupby(['candidate', 'hashtags', 'week', 'week_start']).agg({
    'engagement': 'mean',
    'net_sentiment': 'mean'
}).reset_index()

# Compute overall net sentiment for each candidate and hashtag over the entire period
overall_sentiment = filtered_hs.groupby(['candidate', 'hashtags']).agg({
    'net_sentiment': 'mean'
}).reset_index()

# Classify overall sentiment as Positive if net sentiment > 0, else Negative
sentiment_class = {}
for _, row in overall_sentiment.iterrows():
    sentiment_class[(row['candidate'], row['hashtags'])] = "Positive" if row['net_sentiment'] > 0 else "Negative"

##############################################
# Plot: Engagement Over Time for Top Hashtags with Sentiment Annotation
##############################################

# Create one subplot per candidate
candidates = list(top_hashtags.keys())
n_candidates = len(candidates)
```

```
fig, axs = plt.subplots(n_candidates, 1, figsize=(14, 6 * n_candidates), sharex=True)
if n_candidates == 1:
    axs = [axs]

for ax, candidate in zip(axs, candidates):
    candidate_data = agg[agg['candidate'] == candidate]
    for tag in top_hashtags[candidate]:
        tag_data = candidate_data[candidate_data['hashtags'] == tag].sort_values('week')
        overall_label = sentiment_class.get((candidate, tag), "Unknown")
        # Label includes the hashtag and its overall sentiment classification
        ax.plot(tag_data['week_start'], tag_data['engagement'],
                label=f"{tag} ({overall_label})", marker='o', linestyle='-')
    ax.set_title(f"Top 5 Hashtags Engagement & Sentiment Over Time - {candidate}", fontsize=16)
    ax.set_xlabel("Week")
    ax.set_ylabel("Average Engagement")
    ax.legend()
    ax.grid(True)
    ax.xaxis.set_major_locator(mdates.WeekdayLocator(byweekday=mdates.MO, interval=2))
    ax.xaxis.set_major_formatter(mdates.DateFormatter('%Y-%m-%d'))
    plt.setp(ax.get_xticklabels(), rotation=45)

plt.tight_layout()
plt.savefig('output/hashtag_engagement_sentiment_over_time.png')
plt.show()
```

```
--2025-03-22 04:26:48--  https://raw.githubusercontent.com/aditeyabaral/lok-sabha-election-twitter-analysis/master/NRC-Emotion-L
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2579145 (2.5M) [text/plain]
Saving to: 'NRC-Emotion-Lexicon-Wordlevel-v0.92.txt'

NRC-Emotion-Lexicon 100%[===================>]   2.46M  --.-KB/s    in 0.08s

2025-03-22 04:26:48 (31.8 MB/s) - 'NRC-Emotion-Lexicon-Wordlevel-v0.92.txt' saved [2579145/2579145]

         word   emotion
19     abacus     trust
23    abandon      fear
25    abandon  negative
27    abandon   sadness
30  abandoned     anger
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
1.3.1 Summary statistics for emotions by candidate:
                  anger                                           \
                  count      mean       std  min  25%  50%  75%  max
candidate
Donald Trump     4794.0  0.338340  0.637714  0.0  0.0  0.0  1.0  4.0
Hillary Clinton  4711.0  0.372745  0.694146  0.0  0.0  0.0  1.0  4.0

                anticipation            ... positive      negative            \
                       count      mean  ...      75%  max    count      mean
candidate                               ...
Donald Trump          4794.0  0.417397  ...      1.0  7.0   4794.0  0.620567
Hillary Clinton       4711.0  0.506686  ...      2.0  7.0   4711.0  0.602844


                      std  min  25%  50%  75%  max
candidate
Donald Trump     0.927905  0.0  0.0  0.0  1.0  6.0
Hillary Clinton  0.885352  0.0  0.0  0.0  1.0  7.0

[2 rows x 80 columns]
1.3.2 Summary statistics for engagement metrics by candidate and emotion:

Engagement metrics for anger:
                            favorite_count                              \
                                     count          mean           std
candidate       anger_level
Donald Trump    None                3530.0  13237.448442  13407.883382
                Low                 1200.0  16003.732500  13290.952776
                High                  64.0  20620.156250  14151.815033
Hillary Clinton None                3409.0   5226.577295  14054.552945
                Low                 1210.0   5153.908264   6913.511476
                High                  92.0   4625.358696   5576.737724

                                                                      \
                                min       25%      50%       75%       max
candidate       anger_level
Donald Trump    None          846.0   4501.00   9880.0  17869.50  275228.0
                Low           888.0   5604.00  12571.5  23016.25   82161.0
                High         2658.0  11118.50  18099.5  27062.75   84715.0
Hillary Clinton None          123.0   1610.00   3068.0   5809.00  702603.0
                Low           118.0   1790.25   3167.5   5910.50  103240.0
                High          734.0   1828.00   2972.5   5531.25   43287.0

                            retweet_count                             \
                                    count         mean          std    min
candidate       anger_level
Donald Trump    None               3530.0  4944.041360  5955.200050  370.0
                Low                1200.0  6104.675833  5424.402755  393.0
                High                 64.0  7589.562500  5465.990405  970.0
Hillary Clinton None               3409.0  2319.579349  9931.173274   42.0
                Low                1210.0  2270.085124  3256.272535   86.0
                High                 92.0  2098.663043  2686.351107  397.0


                                 25%     50%       75%       max
candidate       anger_level
Donald Trump    None         1544.00  3421.5   6531.75  152412.0
                Low          2070.00  4517.5   8540.25   34855.0
                High         3838.50  6375.5  10040.25   35669.0
Hillary Clinton None          699.00  1233.0   2377.00  543692.0
                Low           821.00  1338.5   2528.25   56773.0
                High          859.25  1354.0   2330.25   22317.0

Engagement metrics for anticipation:
                                   favorite_count                              \
                                            count          mean           std
```

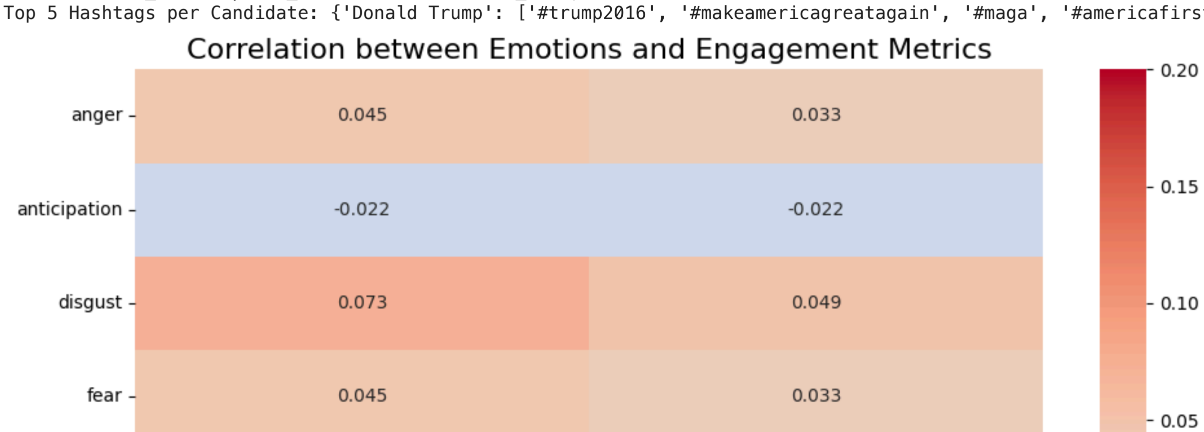
```
candidate       anticipation_level
Donald Trump    None                      3174.0  13919.163201  12484.506040
                Low                       1581.0  14338.701455  15297.522546
                High                        39.0  10344.897436   9583.299146
Hillary Clinton None                      2908.0   5274.299519  14972.073442
                Low                       1723.0   5063.653511   6783.587396
                High                        80.0   5210.325000   6063.107844

                                                                          \
                                   min      25%      50%       75%
candidate       anticipation_level
Donald Trump    None             846.0  4737.50  10804.0  19159.75
                Low             1051.0  5041.00  10502.0  19298.00
                High            1582.0  3058.50   6903.0  15232.00
Hillary Clinton None             150.0  1638.75   3107.0   5744.25
                Low              118.0  1656.00   3048.0   5997.50
                High             401.0  2129.75   3259.5   6818.25

                                         retweet_count               \
                                     max         count         mean
candidate       anticipation_level
Donald Trump    None            120575.0        3174.0  5256.929427
                Low             275228.0        1581.0  5335.575585
                High             46634.0          39.0  3660.769231
Hillary Clinton None            702603.0        2908.0  2386.759629
                Low             116320.0        1723.0  2167.809054
                High             43287.0          80.0  2143.675000

                                                                    \
                                         std    min      25%     50%
candidate       anticipation_level
Donald Trump    None             5114.276372  383.0  1718.75  3834.0
                Low              7120.361996  370.0  1671.00  3555.0
                High             4183.499765  632.0  1080.00  2209.0
Hillary Clinton None            10644.200269   59.0   731.00  1293.0
                Low              3362.680497   42.0   721.50  1238.0
                High             3021.590003  135.0   794.50  1204.0


                                    75%       max
candidate       anticipation_level
Donald Trump    None            7105.75   54440.0
                Low             7054.00  152412.0
                High            4921.50   24698.0
Hillary Clinton None            2440.00  543692.0
                Low             2378.00   66926.0
                High            2358.00   22317.0

Engagement metrics for disgust:
                             favorite_count                              \
                                      count          mean           std
candidate       disgust_level
Donald Trump    None                 3916.0  13586.138662  13416.999981
                Low                   861.0  15972.531940  13357.058388
                High                   17.0  17452.529412  19508.545839
Hillary Clinton None                 4088.0   5098.207192  12966.655551
                Low                   615.0   5824.252033   8670.180878
                High                    8.0   6972.250000   5455.371030

                                                                            \
                              min      25%      50%       75%       max
candidate       disgust_level
Donald Trump    None        846.0  4629.75  10281.0  18357.00  275228.0
                Low        1087.0  5686.00  12225.0  22938.00   80743.0
                High       3357.0  6346.00  11997.0  22407.00   84715.0
Hillary Clinton None        118.0  1605.00   3026.5   5745.25  702603.0
                Low         202.0  2079.00   3380.0   6316.00  103240.0
                High       2040.0  4291.25   5951.0   7286.75   19440.0

                             retweet_count                                   \
                                     count         mean          std     min
candidate       disgust_level
Donald Trump    None                3916.0  5110.769152  5934.212008   370.0
                Low                  861.0  5963.577236  5328.615385   393.0
                High                  17.0  6788.058824  8036.197480  1375.0
Hillary Clinton None                4088.0  2254.584638  9115.795485    42.0
                Low                  615.0  2612.990244  4014.092823   144.0
                High                   8.0  2949.375000  1745.453518  1032.0


                               25%     50%      75%       max
candidate       disgust_level
Donald Trump    None       1622.00  3569.0  6785.25  152412.0
                Low        2048.00  4494.0  8172.00   34855.0
                High       2397.00  4658.0  7321.00   35669.0
Hillary Clinton None        703.00  1233.0  2363.25  543692.0
                Low         896.50  1541.0  2830.50   56773.0
                High       1946.25  2855.5  3305.75    6682.0
```

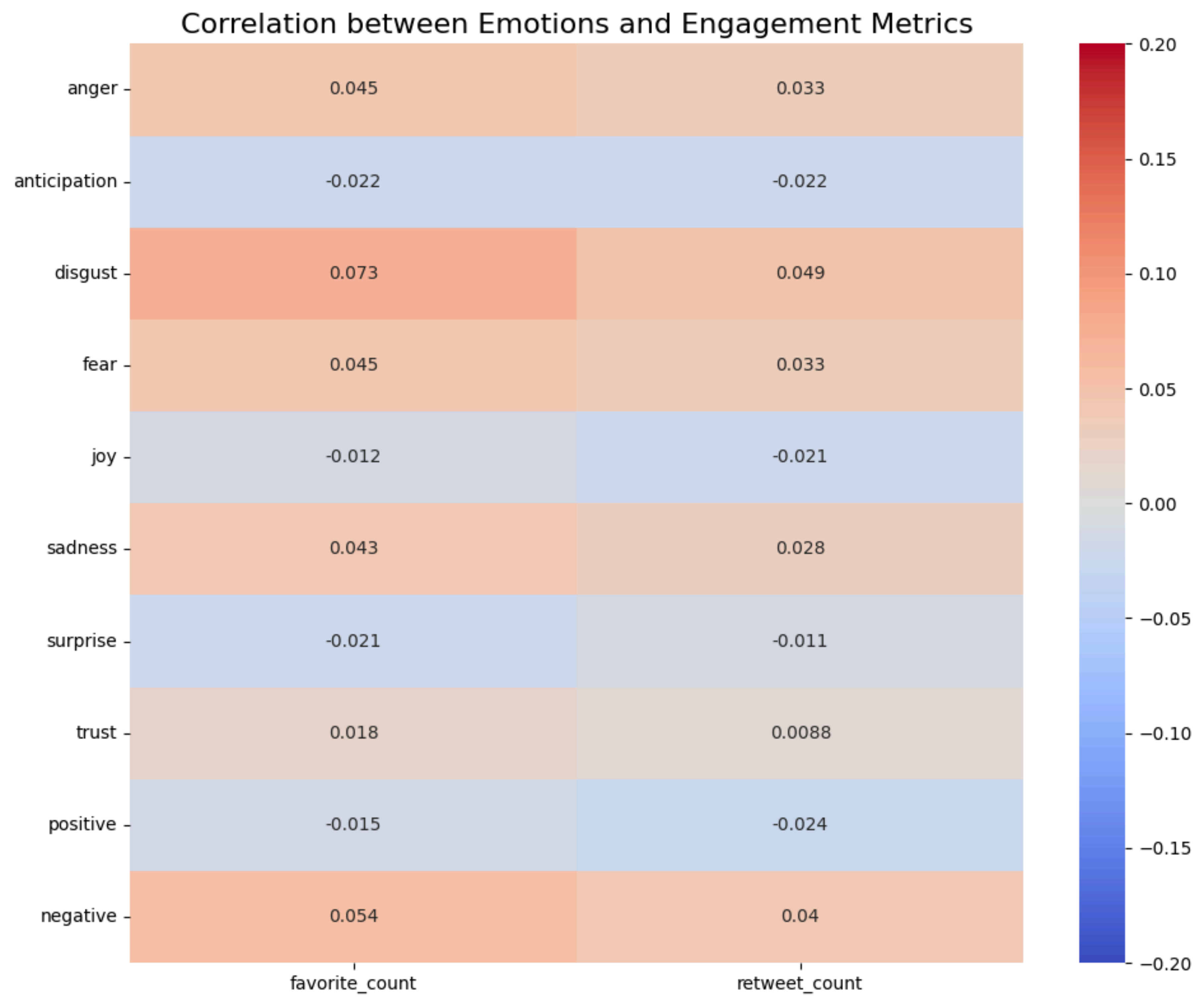
```
                High           1946.25  2855.5  3303.75    6682.0

Engagement metrics for fear:
                           favorite_count                                        \
                                    count          mean           std     min
candidate       fear_level
Donald Trump    None               3629.0  13005.244420  13203.301463   846.0
                Low                1116.0  16975.146953  13431.296025   912.0
                High                 49.0  22695.224490  19218.985530  2893.0
Hillary Clinton None               3380.0   5277.071598  14199.555838   118.0
                Low                1242.0   5057.603865   6432.254968   202.0
                High                 89.0   4057.505618   3421.216753   364.0

                                                                    \
                                25%      50%       75%       max
candidate       fear_level
Donald Trump    None        4494.00   9725.0  17491.00  275228.0
                Low         6242.25  13638.0  24537.75   84715.0
                High        8518.00  17100.0  30932.00   82161.0
Hillary Clinton None        1619.75   3028.5   5821.00  702603.0
                Low         1774.25   3223.5   5934.25  103240.0
                High        1671.00   3096.0   5493.00   19440.0

                           retweet_count                                       \
                                   count         mean           std    min
candidate       fear_level
Donald Trump    None              3629.0  4863.674290   5893.093981  370.0
                Low               1116.0  6452.078853   5431.665298  391.0
                High                49.0  8428.877551   6971.058871  970.0
Hillary Clinton None              3380.0  2344.806509  10004.001811   42.0
                Low               1242.0  2223.516908   3006.261045  131.0
                High                89.0  1800.808989   1448.910852  220.0


                                25%     50%       75%       max
candidate       fear_level
Donald Trump    None        1543.00  3361.0   6362.00  152412.0
                Low         2367.75  5044.5   8971.25   35669.0
                High        3837.00  6140.0  11009.00   32951.0
Hillary Clinton None         702.75  1233.0   2364.00  543692.0
                Low          795.50  1337.5   2550.75   56773.0
                High         863.00  1453.0   2304.00    8595.0
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()

Engagement metrics for joy:
                          favorite_count                                        \
                                   count          mean           std     min
candidate       joy_level
Donald Trump    None              3486.0  14059.525818  13696.531918   846.0
                Low               1265.0  14026.562055  12884.873972   952.0
                High                43.0  11564.046512  10490.642651  1324.0
Hillary Clinton None              3278.0   5098.515558  13992.270639   202.0
                Low               1372.0   5415.524781   8117.343034   118.0
                High                61.0   5510.327869   5495.663584   401.0

                                                                 retweet_count  \
                               25%      50%       75%       max         count
candidate       joy_level
Donald Trump    None       4856.75  10826.5  19048.75  275228.0        3486.0
                Low        4742.00  10326.0  19488.00  140110.0        1265.0
                High       3655.50   9044.0  15893.50   52240.0          43.0
Hillary Clinton None       1591.50   3030.5   5604.00  702603.0        3278.0
                Low        1792.00   3170.0   6303.50  128783.0        1372.0
                High       2190.00   3627.0   7686.00   36599.0          61.0

                                                                           \
                                  mean           std    min     25%     50%
candidate       joy_level
Donald Trump    None       5351.027826   5942.394389  370.0  1753.0  3838.0
                Low        5110.367589   5639.340974  429.0  1586.0  3413.0
                High       3383.976744   3096.112228  483.0  1158.0  2723.0
Hillary Clinton None       2325.640024  10030.684727  100.0   713.0  1264.0
                Low        2257.817055   3770.924718   42.0   767.0  1300.5
                High       2068.081967   2548.366211  189.0   915.0  1236.0


                               75%       max
candidate       joy_level
Donald Trump    None       7129.75  152412.0
```

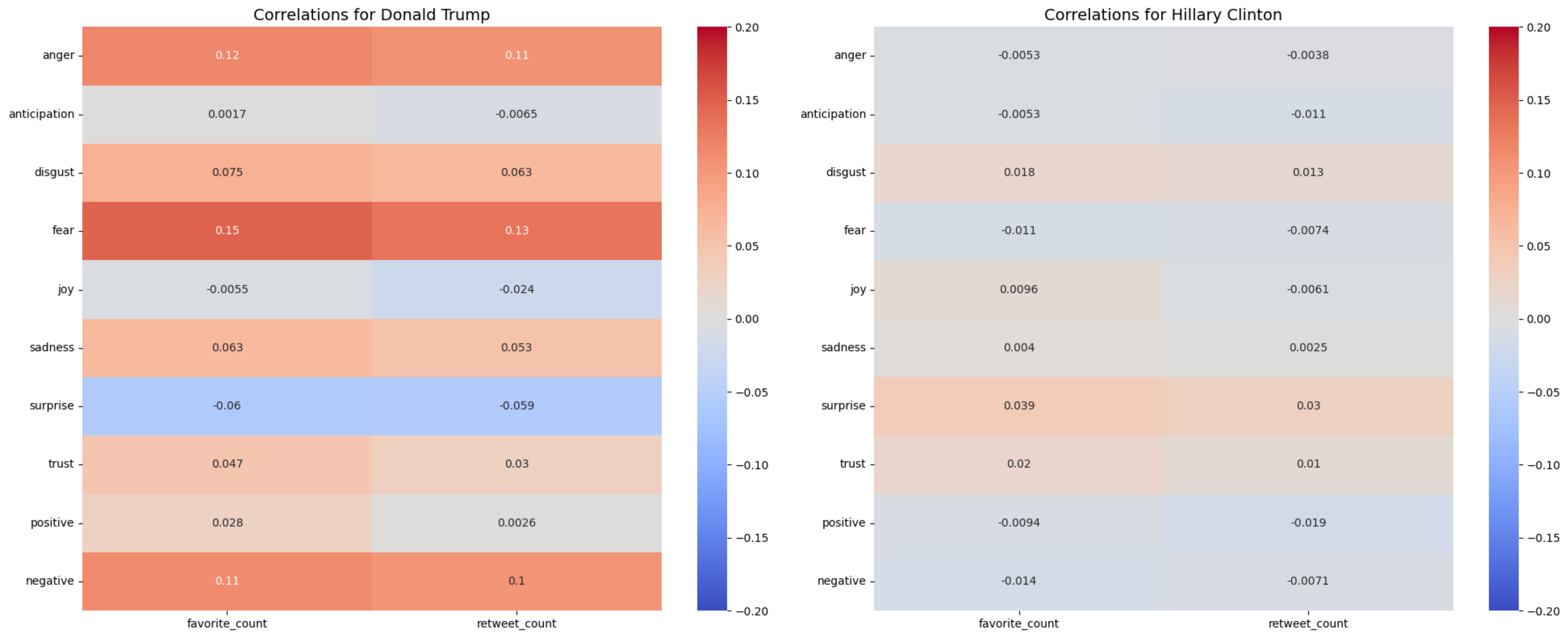
```
                Low        6990.00   99780.0
                High       4490.00   15836.0
Hillary Clinton None       2405.00  543692.0
                Low        2445.00   66926.0
                High       2591.00   18420.0

Engagement metrics for sadness:
                              favorite_count                              \
                                       count          mean           std
candidate       sadness_level
Donald Trump    None                  3515.0  13482.409957  13463.446344
                Low                   1220.0  15513.574590  13245.134330
                High                    59.0  15849.661017  15398.951694
Hillary Clinton None                  3524.0   5176.268161  13859.527154
                Low                   1157.0   5249.651685   6965.157907
                High                    30.0   5471.600000   5948.387829

                                                                             \
                                  min      25%      50%       75%       max
candidate       sadness_level
Donald Trump    None            846.0  4657.50  10146.0  18080.00  275228.0
                Low             888.0  5291.75  12183.0  22705.75   97229.0
                High           1775.0  5884.00  10059.0  20968.50   84715.0
Hillary Clinton None            118.0  1583.75   3004.0   5753.00  702603.0
                Low             202.0  1922.00   3281.0   6132.00  103240.0
                High            742.0  1541.00   3239.0   6329.50   24091.0

                              retweet_count                                  \
                                      count         mean          std    min
candidate       sadness_level
Donald Trump    None                 3515.0  5061.730868  6031.909045  370.0
                Low                  1220.0  5838.830328  5210.604444  391.0
                High                   59.0  5905.966102  6366.108184  601.0
Hillary Clinton None                 3524.0  2292.524404  9779.869177   42.0
                Low                  1157.0  2329.673293  3274.302317  144.0
                High                   30.0  2434.600000  2945.120694  434.0


                                  25%     50%      75%       max
candidate       sadness_level
Donald Trump    None           1614.0  3522.0  6642.00  152412.0
                Low            1923.5  4343.5  8324.25   37623.0
                High           2040.5  3837.0  7187.00   35669.0
Hillary Clinton None            690.5  1231.0  2353.25  543692.0
                Low             861.0  1401.0  2636.00   56773.0
                High            727.5  1379.5  2575.00   14234.0

Engagement metrics for surprise:
                               favorite_count                              \
                                        count          mean           std
candidate       surprise_level
Donald Trump    None                   3529.0  14486.815812  13920.295834
                Low                    1241.0  12771.995165  12047.127348
                High                     24.0  11597.625000   9463.839316
Hillary Clinton None                   3300.0   4886.364242  13974.647752
                Low                    1392.0   5931.359195   7944.662734
                High                     19.0   5142.631579   6259.247809

                                                                              \
                                   min      25%      50%       75%       max
candidate       surprise_level
Donald Trump    None             846.0  5237.00  11255.0  19307.00  275228.0
                Low             1084.0  3761.00   8846.0  18454.00  140110.0
                High            1658.0  3972.75   9424.5  15412.75   37317.0
Hillary Clinton None             118.0  1526.75   2832.0   5327.00  702603.0
                Low              123.0  2052.00   3741.5   7217.75  116320.0
                High            1094.0  1830.00   3090.0   4364.00   24091.0

                               retweet_count                                  \
                                       count         mean          std    min
candidate       surprise_level
Donald Trump    None                  3529.0  5466.976764  6047.491805  370.0
                Low                   1241.0  4733.458501  5239.611722  453.0
                High                    24.0  4026.083333  3431.679420  582.0
Hillary Clinton None                  3300.0  2136.136061  9973.424464   59.0
                Low                   1392.0  2696.512213  3886.826129   42.0
                High                    19.0  2343.789474  2752.206964  559.0


                                    25%     50%      75%       max
candidate       surprise_level
Donald Trump    None            1851.00  3919.0  7173.00  152412.0
                Low             1344.00  3134.0  6496.00   99780.0
                High            1626.25  3406.0  5087.75   15145.0
Hillary Clinton None             668.00  1165.0  2187.00  543692.0
                Low              900.00  1579.0  3192.50   58271.0
                High             643.00  1615.0  1936.50    9778.0
```

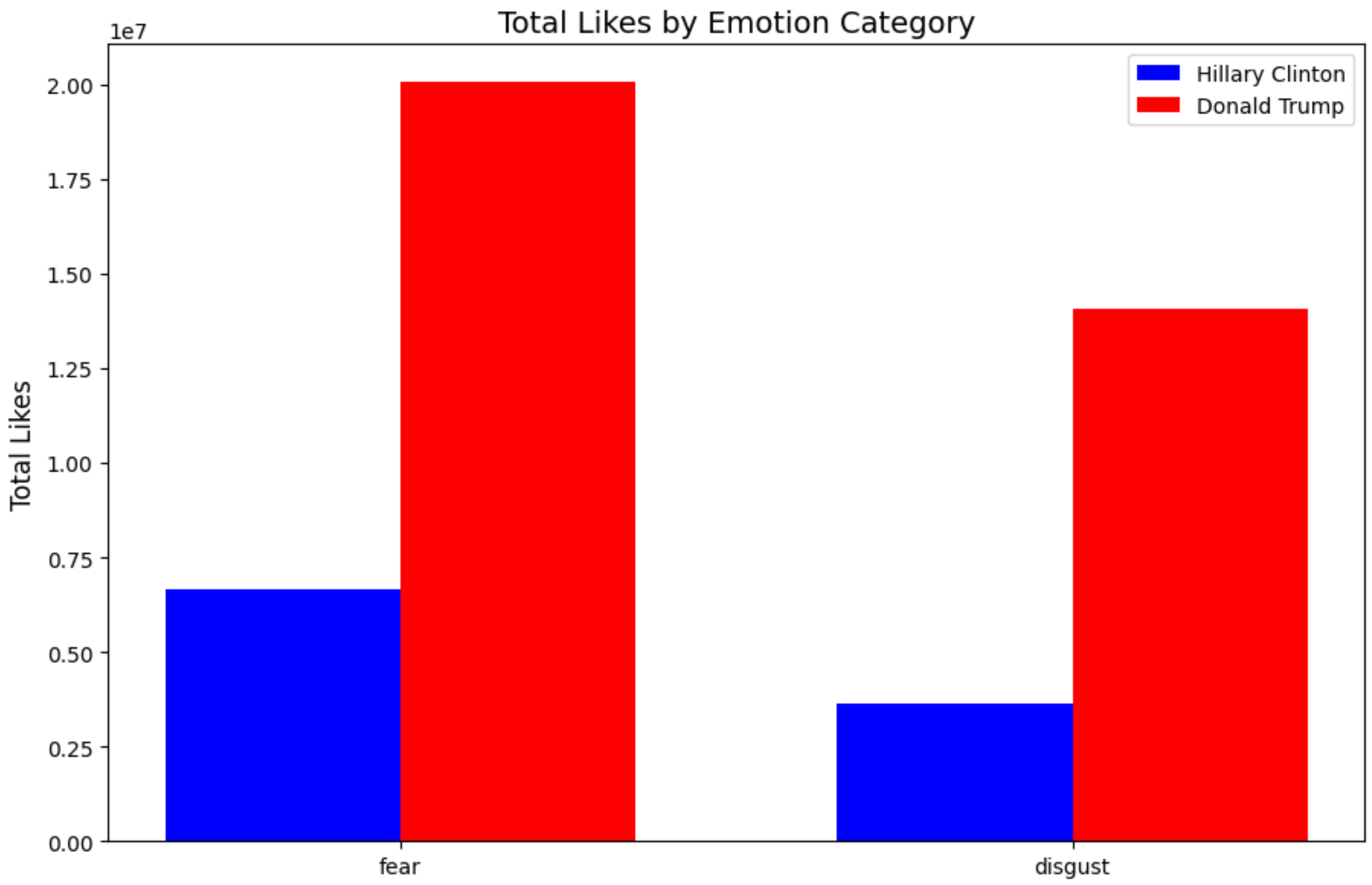
```
Engagement metrics for trust:
                             favorite_count                              \
                                      count          mean           std   
candidate        trust_level                                              
Donald Trump     None                2669.0  13455.023979  13276.005052   
                 Low                 1999.0  14674.923462  13420.593116   
                 High                 126.0  15918.500000  17026.053976   
Hillary Clinton None                2544.0   4803.571541   6994.957587   
                 Low                 1959.0   5702.608474  17528.196961   
                 High                 208.0   5228.211538   5790.888274   

                                                                            \
                                 min      25%      50%       75%       max   
candidate        trust_level                                                 
Donald Trump     None          952.0  4821.00  10603.0  18003.00  275228.0   
                 Low           846.0  4804.50  10879.0  20551.00  140110.0   
                 High         1324.0  4886.25  11505.5  20683.75  120575.0   
Hillary Clinton None          150.0  1533.75   2897.5   5402.25  105777.0   
                 Low           118.0  1771.50   3302.0   6388.50  702603.0   
                 High          671.0  2050.75   3282.0   6451.75   43287.0   

                             retweet_count                                  \
                                     count         mean           std    min   
candidate        trust_level                                                   
Donald Trump     None               2669.0  5080.659423   5880.033033  383.0   
                 Low                1999.0  5512.190095   5804.810002  370.0   
                 High                126.0  5433.817460   5716.657644  483.0   
Hillary Clinton None               2544.0  2109.628931   3316.783005   59.0   
                 Low                1959.0  2568.106687  12782.869802   42.0   
                 High                208.0  2161.100962   2650.125159  297.0   


                                  25%     50%      75%       max  
candidate        trust_level                                      
Donald Trump     None         1717.00  3703.0  6677.00  152412.0  
                 Low          1664.50  3732.0  7486.00   99780.0  
                 High         1772.00  3996.5  7278.00   37976.0  
Hillary Clinton None          679.75  1214.0  2284.50   56675.0  
                 Low           776.50  1356.0  2588.50  543692.0  
                 High          842.75  1301.5  2362.75   22317.0  

Engagement metrics for positive:
                                favorite_count                              \
                                         count          mean           std   
candidate        positive_level                                              
Donald Trump     None                   2040.0  13548.005392  12936.262184   
                 Low                    2438.0  14306.055373  13452.161813   
                 High                    316.0  14988.189873  16459.021209   
Hillary Clinton None                   1797.0   5379.711742  18065.051421   
                 Low                    2439.0   5080.350964   7121.810121   
                 High                    475.0   5096.517895   7110.341636   

                                                                           \
                                   min     25%      50%       75%       max   
candidate        positive_level                                               
Donald Trump     None            846.0  4847.5  10603.0  18565.50  275228.0   
                 Low             888.0  4813.0  10873.0  19499.25  231783.0   
                 High            952.0  4670.0  10185.5  19300.00  140110.0   
Hillary Clinton None            202.0  1517.0   2977.0   5546.00  702603.0   
                 Low             123.0  1740.5   3147.0   5984.00  128783.0   
                 High            118.0  1784.0   3188.0   6326.00  116320.0   

                                retweet_count                             \
                                        count         mean           std   
candidate        positive_level                                            
Donald Trump     None                  2040.0  5185.962745   5827.058139   
                 Low                   2438.0  5335.769483   5622.984535   
                 High                   316.0  5303.287975   7466.203180   
Hillary Clinton None                  1797.0  2514.150807  13251.050806   
                 Low                   2439.0  2191.552686   3441.238624   
                 High                   475.0  2072.000000   3324.879717   


                                   min      25%     50%      75%       max  
candidate        positive_level                                             
Donald Trump     None            386.0  1752.75  3790.5  6956.75  152412.0  
                 Low             370.0  1671.50  3709.5  7189.75  109908.0  
                 High            429.0  1622.25  3520.0  6324.00   99780.0  
Hillary Clinton None            100.0   682.00  1265.0  2463.00  543692.0  
                 Low              42.0   761.00  1282.0  2378.00   66926.0  
                 High             69.0   746.50  1276.0  2535.00   58271.0  
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
```

```
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()
<ipython-input-27-5e903c925637>:189: FutureWarning: The default of observed=False is deprecated and will be changed to True in a
  stats = merged_df_with_sentiments.groupby(['candidate', f'{emotion}_level'])[engagement_metrics].describe()

Engagement metrics for negative:
                              favorite_count                              \
                                       count          mean           std
candidate       negative_level
Donald Trump    None                  2891.0  12821.592183  12689.461129
                Low                   1645.0  15588.015805  14479.279683
                High                   258.0  17607.957364  13518.740458
Hillary Clinton None                  2797.0   5383.825170  15368.734082
                Low                   1713.0   4882.538821   6232.773155
                High                   201.0   5257.791045   5766.779107

                                                                           \
                                  min     25%      50%      75%       max
candidate       negative_level
Donald Trump    None            846.0  4522.5   9784.0  17142.0  275228.0
                Low             888.0  5105.0  11847.0  22216.0  231783.0
                High           1775.0  6470.0  13779.0  25097.0   84715.0
Hillary Clinton None            123.0  1545.0   2980.0   5937.0  702603.0
                Low             118.0  1823.0   3165.0   5571.0  103240.0
                High            364.0  1915.0   3290.0   6345.0   43287.0

                              retweet_count                             \
                                      count         mean           std
candidate       negative_level
Donald Trump    None                 2891.0  4768.699412   5705.439699
                Low                  1645.0  5947.810942   6070.035099
                High                  258.0  6563.372093   5332.575714
Hillary Clinton None                 2797.0  2378.782267  10918.235586
                Low                  1713.0  2171.775248   2932.398568
                High                  201.0  2356.323383   2769.794637


                                 min      25%     50%      75%       max
candidate       negative_level
Donald Trump    None            370.0  1554.50  3364.0  6208.50  152412.0
                Low             391.0  1817.00  4207.0  8294.00  109908.0
                High            601.0  2689.75  5065.5  8901.75   35669.0
Hillary Clinton None             42.0   669.00  1199.0  2365.00  543692.0
                Low              86.0   826.00  1322.0  2456.00   56773.0
                High            220.0   900.00  1451.0  2837.00   22317.0

Selected emotions for visualization in 1.4: ['fear', 'disgust']
1. Fear: Selected because it showed significant differences between candidates' tweets.
2. Disgust: Selected as a complementary negative emotion that reveals interesting patterns in engagement metrics.

Research Question: Is there a relationship between emotion categories and engagement?

Correlation between emotions and engagement metrics:
              favorite_count  retweet_count
anger               0.045282       0.032942
anticipation       -0.022239      -0.021534
disgust             0.072825       0.049088
fear                0.044910       0.033123
joy                -0.012288      -0.021456
sadness             0.042586       0.028284
surprise           -0.020783      -0.011259
trust               0.017709       0.008753
positive           -0.014847      -0.024428
negative            0.054449       0.039811
Visualizing engagement metrics for selected emotions: ['fear', 'disgust']
Visualization completed and saved to output/emotion_engagement_comparison.png
<ipython-input-27-5e903c925637>:605: UserWarning: Could not infer format, so each element will be parsed individually, falling b
  df['created_at'] = pd.to_datetime(df['created_at'], errors='coerce')
Top 5 Hashtags per Candidate: {'Donald Trump': ['#trump2016', '#makeamericagreatagain', '#maga', '#americafirst', '#draintheswam
```

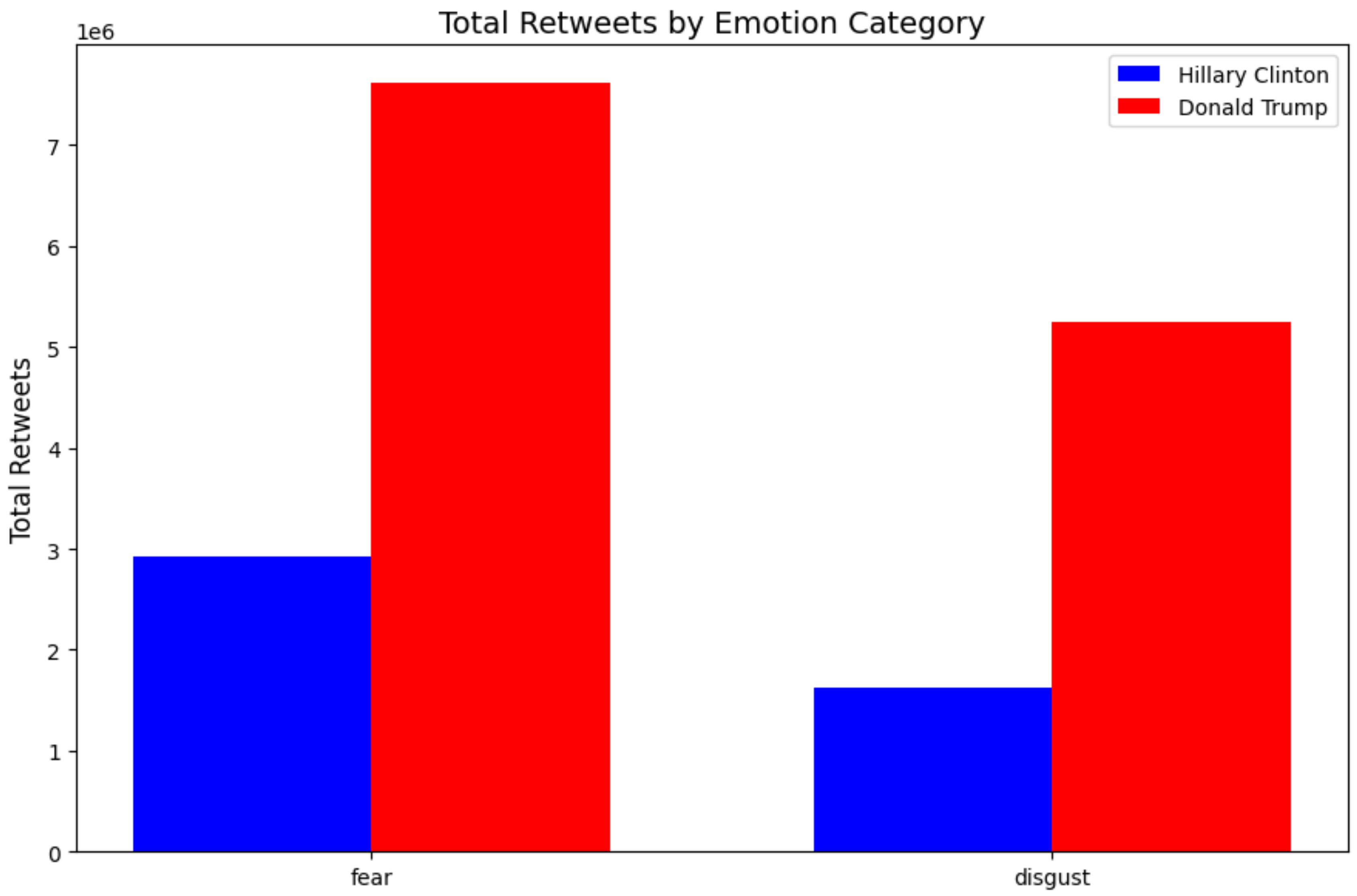



Correlation between Emotions and Engagement Metrics
anger
anticipation
disgust
fear
joy
sadness
surprise
trust
positive
negative
favorite_count
retweet_count
0.045
0.033
-0.022
-0.022
0.073
0.049
0.045
0.033
-0.012
-0.021
0.043
0.028
-0.021
-0.011
0.018
0.0088
-0.015
-0.024
0.054
0.04
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20

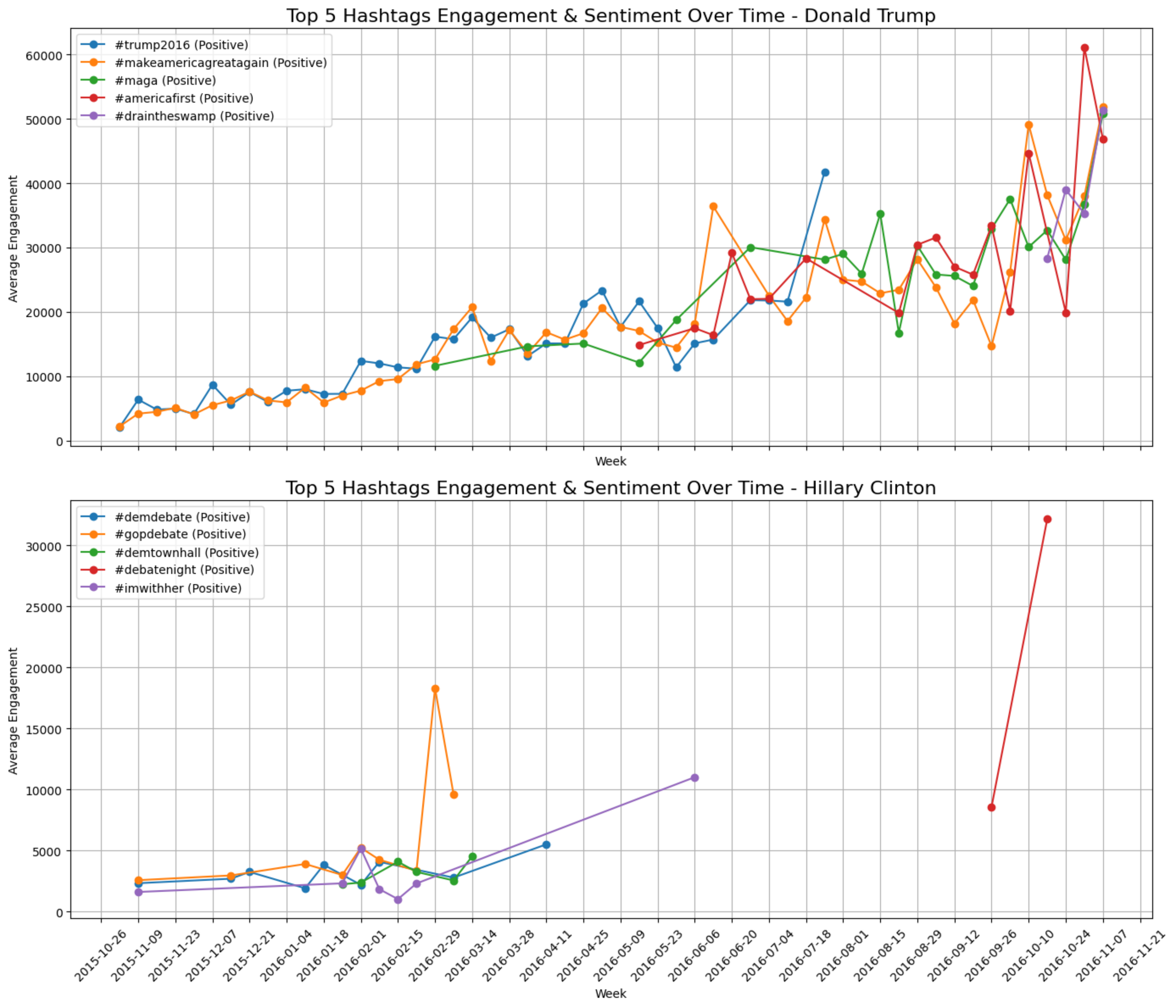

Correlations for Donald Trump
anger
anticipation
disgust
fear
joy
sadness
surprise
trust
positive
negative
favorite_count
retweet_count
0.12
0.11
0.0017
-0.0065
0.075
0.063
0.15
0.13
-0.0055
-0.024
0.063
0.053
-0.06
-0.059
0.047
0.03
0.028
0.0026
0.11
0.1
Correlations for Hillary Clinton
-0.0053
-0.0038
-0.0053
-0.011
0.018
0.013
-0.011
-0.0074
0.0096
-0.0061
0.004
0.0025
0.039
0.03
0.02
0.01
-0.0094
-0.019
-0.014
-0.0071
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20



Total Likes by Emotion Category
1e7
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Total Likes
fear
disgust
Hillary Clinton
Donald Trump



Total Retweets by Emotion Category
1e6
7
6
5
4
3
2
1
0
Total Retweets
fear
disgust
Hillary Clinton
Donald Trump



Top 5 Hashtags Engagement & Sentiment Over Time - Donald Trump
#trump2016 (Positive)
#makeamericagreatagain (Positive)
#maga (Positive)
#americafirst (Positive)
#draintheswamp (Positive)
Average Engagement
Week
Top 5 Hashtags Engagement & Sentiment Over Time - Hillary Clinton
#demdebate (Positive)
#gopdebate (Positive)
#demtownhall (Positive)
#debatenight (Positive)
#imwithher (Positive)
Average Engagement
Week