# Social Media Sentiment Analysis


**THESIS FOR INFOTACT SOLUTIONS**

Aman Jawalekar

# INDEX

# Acknowledgement

We sincerely thank **Infotact Solutions** for granting us the opportunity to work on the **Social Media Sentiment Analysis** project. This experience has enriched our knowledge of real-world data analytics and its practical applications.

We are deeply grateful to our mentors and instructors for their continuous guidance and constructive feedback, which ensured that our work aligned with industry standards and business objectives.

Lastly, we appreciate the data sources and tools that facilitated this study, enabling us to transform raw data into meaningful insights and actionable strategies.

## Preface

This project, **"Social Media Sentiment Analysis,"** has been developed to understand how Natural Language Processing (NLP) can be used to analyze opinions expressed on social media platforms. With the rise of digital interactions, sentiment analysis has become a crucial tool for businesses to gauge customer satisfaction and brand reputation.

The purpose of this project is to apply NLP techniques on real-world social media data, classify sentiments into **Positive**, **Negative**, or **Neutral**, and present meaningful insights through visualizations. This report summarizes the methodology, analysis, and findings, aiming to provide practical recommendations for improving brand perception.

## Introduction

Social media platforms, such as Twitter, have become major channels for users to express their views on products, services, and brands. These opinions can influence the reputation and success of a business, making sentiment analysis an essential practice in today's competitive environment.

This project leverages **Natural Language Processing (NLP)** techniques to analyze sentiments in social media text. The process involves **data collection, text preprocessing (tokenization, lemmatization, stopword removal), and sentiment classification** using tools like **VADER** and **TextBlob**. The output includes **visualizations** such as word clouds and sentiment trends to identify user perceptions over time.

The insights derived from this analysis help businesses monitor public opinion, detect shifts in customer sentiment, and implement strategies to enhance brand perception.

## Methodology

1. **Data Collection**

2. **Data Cleaning**

3. **Text Pre-processing**

4. **VADER Sentiment Analysis**

5. **Sentiment Distribution**

6. **Results**

# Data Collection

The dataset used for this project was sourced from **Kaggle**: Twitter Entity Sentiment Analysis. It contains tweets mentioning various entities (brands, products, organizations) along with sentiment labels.

The dataset consists of **74,681 rows and 4 columns**:

- **ID** – Unique identifier for each tweet.
- **Entity** – The brand or product mentioned in the tweet.
- **Sentiment** – Labeled as **Positive**, **Negative**, or **Neutral**.
- **Tweet Content** – The actual text of the tweet.

This dataset provides a balanced mix of sentiments and entities, making it suitable for analyzing public opinions and brand perception on social media platforms.

| | id | entity | sentiment | text |
|---|---|---|---|---|
| **0** | 2401 | Borderlands | Positive | im getting on borderlands and i will murder yo... |
| **1** | 2401 | Borderlands | Positive | I am coming to the borders and I will kill you... |
| **2** | 2401 | Borderlands | Positive | im getting on borderlands and i will kill you ... |
| **3** | 2401 | Borderlands | Positive | im coming on borderlands and i will murder you... |
| **4** | 2401 | Borderlands | Positive | im getting on borderlands 2 and i will murder ... |
| **...** | ... | ... | ... | ... |
| **74677** | 9200 | Nvidia | Positive | Just realized that the Windows partition of my... |
| **74678** | 9200 | Nvidia | Positive | Just realized that my Mac window partition is ... |
| **74679** | 9200 | Nvidia | Positive | Just realized the windows partition of my Mac ... |
| **74680** | 9200 | Nvidia | Positive | Just realized between the windows partition of... |
| **74681** | 9200 | Nvidia | Positive | Just like the windows partition of my Mac is l... |

74682 rows × 4 columns

---

The dataset contains 74,682 rows and 4 columns, suggesting a large-scale analysis. The columns are:

- **id**: A unique identifier for each entry.
- **entity**: The subject of the text, such as "Borderlands" (a video game) or "Nvidia" (a technology company).
- **sentiment**: The sentiment associated with the text, which is labeled as "Positive" in all the visible rows. This suggests the project is classifying text into different sentiment categories.
- **text**: The actual text data being analyzed. The text samples shown are varied and include mentions of the entities.

The goal of this project is to determine the sentiment (positive, negative, or neutral) of a given text in relation to a specific entity.

# Text Pre-Processing:

```
# Load English NLP model
nlp = spacy.load("en_core_web_sm", disable=["parser","ner"])
stop_words = set(stopwords.words('english'))

def clean_text(text):
    text = str(text).lower()  # lowercase
    text = re.sub(r"http\S+|www\S+|https\S+", '', text)  # remove URLs
    text = re.sub(r"@\w+", '', text)  # remove mentions
    text = re.sub(r"[^a-z\s]", '', text)  # remove punctuation/numbers
    doc = nlp(text)
    tokens = [t.lemma_ for t in doc if t.lemma_ not in stop_words and t.lemma_.strip()]
    return " ".join(tokens)

df['clean_text'] = df['text'].apply(clean_text)
print(df[['text', 'clean_text', 'sentiment']].head())
```

The code demonstrates the **data preprocessing** stage of a natural language processing (NLP) project for sentiment analysis. The main objective is to clean and prepare raw text data for analysis by a machine learning model. It uses the spaCy library for advanced text processing and regular expressions for initial cleaning.

## Key Points

1. **NLP Model Loading**: The code loads a pre-trained English NLP model (en_core_web_sm) from the spaCy library. It disables the parser and named entity recognizer (parser, ner) to focus on basic tokenization and lemmatization, which makes the process more efficient.

2. **Stop Word Removal**: A set of common English stop words (like "the", "a", "is") is created to be removed from the text. This is a standard practice to reduce noise and focus on meaningful words.
3. **Text Cleaning Function (clean_text)**: A function is defined to perform a series of cleaning operations on the text data:
    - **Lowercasing**: All text is converted to lowercase to ensure consistency.
    - **URL Removal**: Regular expressions are used to find and remove URLs starting with "http", "https", or "www".
    - **Mention Removal**: Mentions (words starting with "@") are removed, as they are often not relevant for sentiment analysis.
    - **Punctuation/Number Removal**: Punctuation and numbers are removed using a regular expression, leaving only alphabetical characters.
    - **Tokenization & Lemmatization**: The cleaned text is processed by the spaCy model to break it into individual words (tokens) and convert them to their base or root form (lemmas), e.g., "running" becomes "run".
    - **Stop Word Filtering**: The function filters out the previously loaded stop words.
    - **Output**: The cleaned and processed tokens are joined back into a single string.
4. **Applying the Cleaning Function**: The clean_text function is applied to the 'text' column of a pandas DataFrame (df). The results are stored in a new column named 'clean_text'.
5. **Displaying Results**: The final line of code prints the first few rows (.head()) of the DataFrame, showing the original 'text', the new 'clean_text', and the 'sentiment' column, allowing for a quick check of the cleaning process's effectiveness.

## VADER Sentiment Analysis

```
analyzer = SentimentIntensityAnalyzer()

def vader_label(text):
    score = analyzer.polarity_scores(text)['compound']
    if score >= 0.05:
        return 'Positive'
    elif score <= -0.05:
        return 'Negative'
    else:
        return 'Neutral'

df['vader_sentiment'] = df['clean_text'].apply(vader_label)
```

**VADER Initialization**: The code initializes an instance of SentimentIntensityAnalyzer, which is the VADER model. VADER does not require training and works by using a pre-built dictionary of sentiment-ranked words.

**Sentiment Scoring Function (vader_label)**: A function is defined to apply VADER's logic to each text entry.

**Polarity Score**: It calculates the sentiment score of the text using analyzer.polarity_scores(). This returns a dictionary with different scores, but the code specifically uses the **'compound' score**. The compound score is a normalized, single-value metric that represents the overall sentiment of the text.

**Sentiment Labeling**: The function then uses a simple rule-based system to classify the sentiment based on the compound score:

1. If the score is **greater than or equal to 0.05**, the text is labeled **'Positive'**.
2. If the score is **less than or equal to -0.05**, the text is labeled **'Negative'**.
3. Otherwise (if the score is between -0.05 and 0.05), the text is labeled **'Neutral'**.

**Applying the Model**: The vader_label function is applied to the 'clean_text' column of the DataFrame. The resulting sentiment labels are stored in a new column called 'vader_sentiment', which can then be used for further analysis or reporting.

## Sentiment Distribution ¶

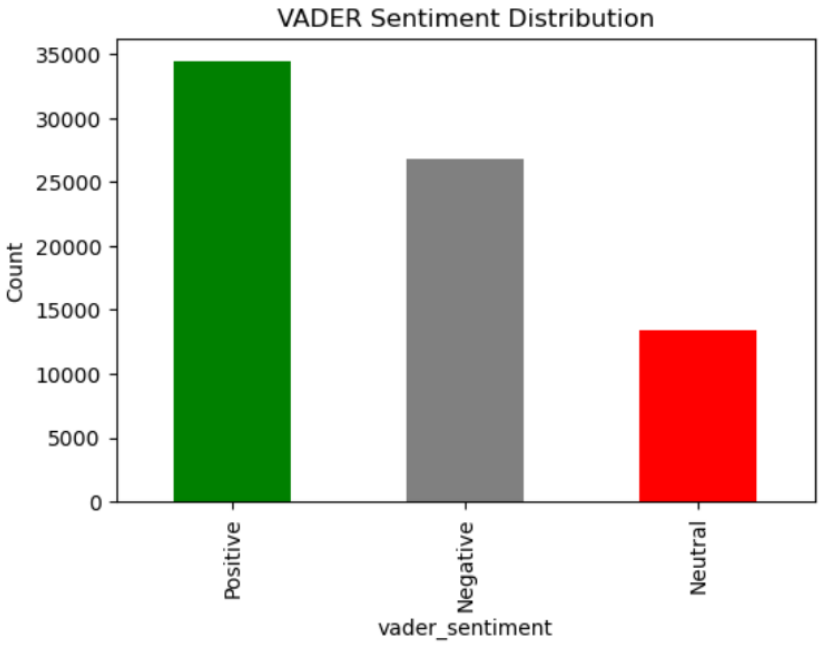
```
plt.figure(figsize=(6,4))
df['sentiment'].value_counts().plot(kind='bar', color=['green','grey','red'])
plt.title("Dataset Sentiment Distribution")
plt.ylabel("Count")
plt.show()

plt.figure(figsize=(6,4))
df['vader_sentiment'].value_counts().plot(kind='bar', color=['green','grey','red'])
plt.title("VADER Sentiment Distribution")
plt.ylabel("Count")
plt.show()
```


"Dataset Sentiment Distribution," shows the count of each sentiment category, which includes 'Negative', 'Positive', 'Neutral', and 'Irrelevant'.

---

1. **Dominant Sentiments**: The 'Negative' sentiment category has the highest count, with over 20,000 entries. 'Positive' sentiment is the second most frequent, with a count slightly above 20,000. This indicates a high volume of both negative and positive opinions in the dataset.
2. **Neutral and Irrelevant Sentiments**: The 'Neutral' category has a count of approximately 18,000, while the 'Irrelevant' category has the lowest count, at just over 12,500 entries.
3. **Data Imbalance**: The chart reveals a slight imbalance in the dataset. The negative and positive classes are the most numerous, while the irrelevant class is the least represented. This information is crucial for model training, as it may influence the choice of evaluation metrics or the need for techniques like oversampling or undersampling to balance the classes.




"VADER Sentiment Distribution" and displays the count for 'Positive', 'Negative', and 'Neutral' sentiments as categorized by the model's logic.

---

1. **Positive Sentiment Dominance**: The 'Positive' sentiment category has the highest count by a significant margin, with approximately 34,000 entries. This suggests that the VADER model classified a large majority of the text as positive.
2. **Negative and Neutral Sentiments**: The 'Negative' sentiment is the second most frequent, with a count of around 27,000. 'Neutral' sentiment has the lowest count, at just over 13,000 entries.
3. **Model's Classification Tendency**: Comparing this chart with the previous one (Dataset Sentiment Distribution), there is a notable shift. The VADER model seems to have a strong tendency to classify text as "Positive" and "Negative," while assigning a much smaller proportion to the "Neutral" category. The 'Irrelevant' category from the original dataset is not present here, as VADER only classifies text into positive, negative, and neutral categories based on its scoring system.

## Results:

### 5. Word Clouds

```
# Positive tweets
pos_text = " ".join(df[df['sentiment']=='Positive']['clean_text'])
wc_pos = WordCloud(width=800, height=400, background_color='white').generate(pos_text)
plt.figure(figsize=(12,6))
plt.imshow(wc_pos, interpolation='bilinear')
plt.axis('off')
plt.title("Word Cloud - Positive Tweets")
plt.show()
```

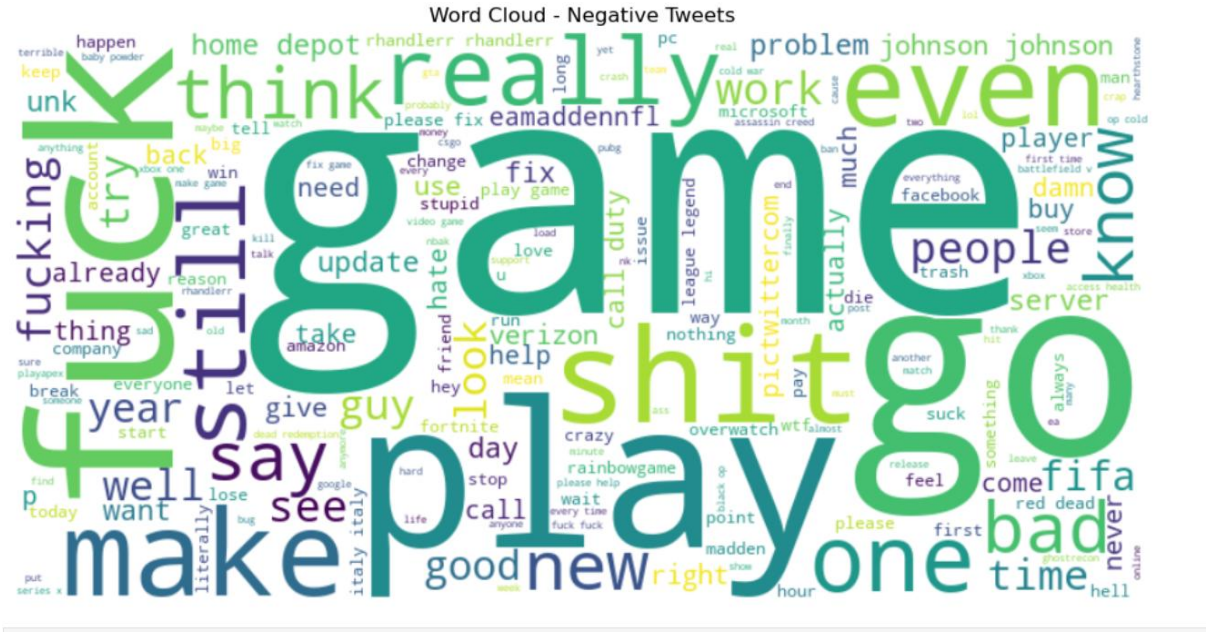

This part of the project visualizes the most frequent words found in the "Positive" sentiment texts

---

**Key Observations:**

1. **Prominent Words**: The word cloud highlights the most common and impactful words. The largest words, indicating the highest frequency, include:
    - **"game"**: This word is centrally located and significantly larger than all others, confirming that a large portion of the positive text is related to gaming. This aligns with the "Borderlands" entity seen in the initial dataset.
    - **"new"**, **"one"**, **"well"**, and **"play"**: These words are also very large, suggesting they are frequently used in a positive context, such as "new game," "play one," or "play well."

2. **Entity Mentions**: Words related to specific entities are visible, though smaller than the most frequent ones. This includes "borderland," "nvidia," "microsoft," "xbox," and "fifa," all of which were likely subjects of the sentiment analysis.
3. **Positive Context Words**: Many words with a positive connotation are present, such as "awesome," "amazing," "good," "nice," "fun," and "great." These words provide insight into why the text was classified as positive, confirming that the VADER model correctly identified the sentiment based on these words.
4. **Overall Insight**: The word cloud provides a clear, at-a-glance summary of the most important terms driving the positive sentiment in the dataset. It visually validates the sentiment analysis findings and confirms the topics of discussion (primarily gaming and technology).




This part of the project visualizes the most frequent words found in the "Negative" sentiment texts.

---

**Key Observations:**

1. **Prominent Words**: The word cloud highlights the most common and impactful words in the negative tweets. Similar to the positive word cloud, the word **"game"** is centrally located and significantly large, indicating that a large portion of the negative

text is also related to gaming. This shows that the **gaming topic** is a major subject of discussion, regardless of the sentiment.

2. **Negative Connotation Words**: The word cloud contains several words with a negative or strong, angry connotation. The word **"shit"** is prominently displayed, as are words like **"bad," "terrible," "suck,"** and **"problem."** These terms clearly indicate the negative sentiment and are likely a key reason for the VADER model's classification.
3. **Action-Oriented Verbs**: Words like **"fix," "need,"** and **"work"** are also very frequent. These often appear in a negative context, such as "fix the game," "need to work," or "it doesn't work," suggesting frustration with a product or service.
4. **Overall Insight**: The word cloud for negative sentiment provides a visual summary of the most frequent terms associated with user dissatisfaction. It confirms that the negative sentiment is often directed at the functionality or quality of the games and services, reinforcing the findings of the sentiment analysis.

- **Engage via Mobile:** SMS or app notifications for deals.

## Recommend strategies to improve brand perception

1. **Identify Key Pain Points**

   Use sentiment spikes (negative tweets) to detect recurring complaints or frustrations.

   Example: If “Borderlands” has negative spikes around game updates, prioritize fixing bugs or clarifying updates.

   Strategy: Create a dedicated feedback response team to quickly address frequent complaints.

2. **Amplify Positive Experiences**

Analyze positive sentiment tweets to understand what customers love.

Example: Users praising Nvidia’s performance or drivers.

Strategy: Highlight these experiences through retweets, testimonials, or marketing campaigns to strengthen brand credibility.

3. **Engage with Customers Proactively**

   Track neutral or mixed sentiment tweets—they often indicate uncertainty.

   Example: Users unsure about product features or compatibility.

   Strategy: Launch targeted social media campaigns, tutorials, FAQs, or live Q&A sessions to clarify doubts.

4. **Monitor Trends and Timing**

   Analyze sentiment over time to identify when perception worsens or improves.

   Example: Negative spikes after software updates or product launches.

   Strategy: Plan proactive communication around launches—release notes, guides, and community engagement to preempt negative reactions.

5. **Personalize Communication**

Use entity-level sentiment to segment audiences by interest or product experience.

Example: Tailor campaigns separately for Nvidia GPU users versus general PC gamers.

Strategy: Deliver relevant content, promotions, or support based on sentiment and user segment to increase satisfaction.