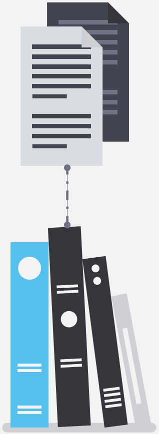
Sentiment Classification of Tweets About Products and Brands Using NLP

Table of contents



Business Understanding

In today's competitive tech landscape, brand perception plays a crucial role in product adoption, customer loyalty, and overall market performance. Social media platforms like Twitter provide real-time, unfiltered insights into how users feel about products and services. For global tech giants like **Apple** and **Google**, monitoring public sentiment around their products (e.g., iPhones, Android devices, iOS, Pixel phones, etc.) can help:

- Identify product issues early
- Understand customer preferences
- Measure the impact of product launches or controversies
- Inform marketing, PR, and customer support strategies

Given the vast volume and velocity of tweets, manual sentiment tracking is impractical — which makes automated sentiment classification an essential tool for brand managers and product teams.

PROBLEM STATEMENT

The goal of this project is to develop a machine learning model that can automatically classify the sentiment of tweets related to Apple and Google products as Positive, Negative, or Neutral based on the content of the tweet.

Stakeholders:

Marketing teams
Brand managers
Customer experience teams

Objectives

- **Develop a machine learning model** to classify whether a tweet contains an emotion directed at a product or brand, based on its text content.
- **Automated monitoring of consumer emotions** toward tech products (e.g., Apple, Google) on social media.
- **Perform Robust Data Preprocessing:** Implement thorough text cleaning, tokenization, stopword removal, and stemming to prepare high-quality textual data for modeling.
- **Engineer Predictive Features:**
  Extract relevant textual features (e.g., TF-IDF unigrams) and use mutual information for selecting the most informative terms.
- **Evaluate and Compare Models:**
  Train and evaluate multiple classification algorithms (e.g., Logistic Regression, Random Forest) and select the best-performing one based on accuracy, precision, recall, and F1-score.

Data Understanding

Link : https://data.world/crowdflower/brands-and-product-emotions

The dataset contains over 9,000 tweets mentioning Apple or Google products, labeled with whether an emotion is present and directed at a brand or product. Each tweet includes text, a target entity (like iPhone or Google), and a sentiment label such as "Positive emotion," "Negative emotion," or "No emotion."

Features Before Cleaning

1. tweet_text: the raw tweet
2. emotion_in_tweet_is_directed_at: specific product/brand (e.g., iPhone, Google)
3. Is_there_an_emotion_directed_at_a_brand_or_ product: sentiment label

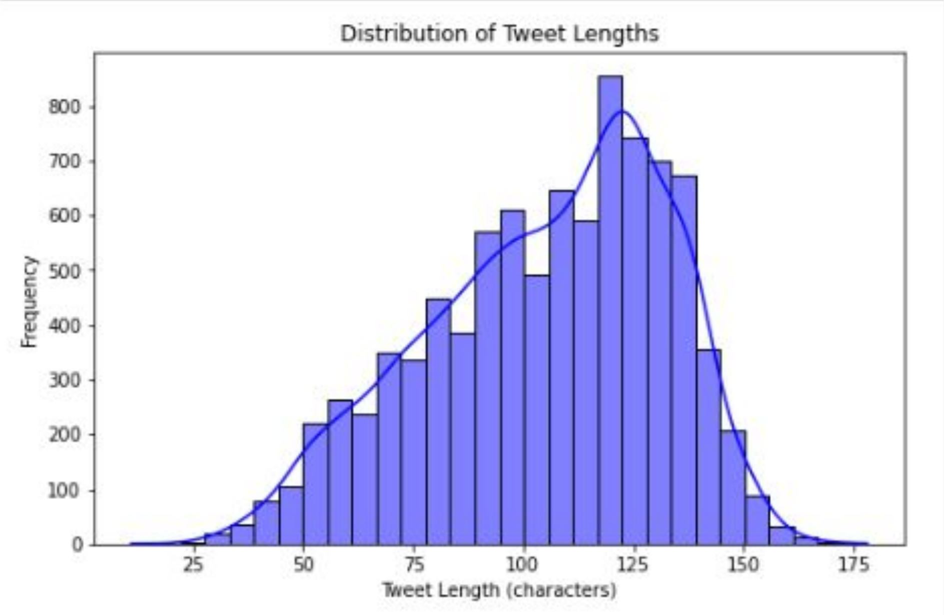
Final Features After Cleaning

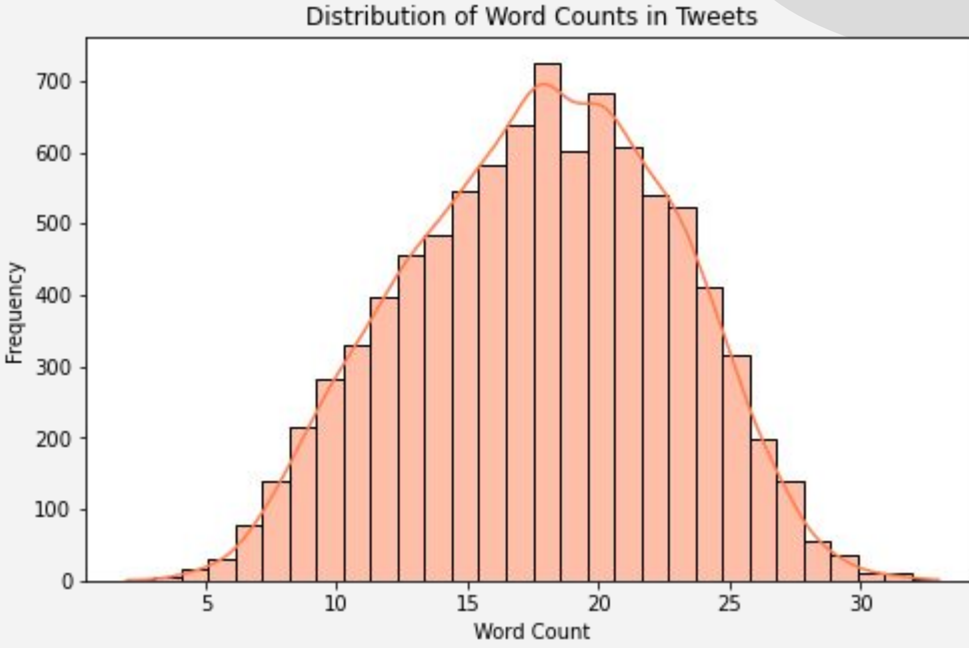
1. **text** – original tweet text
2. **target** – specific product/brand
3. **sentiment** – cleaned label (3-class)
4. **brand** – mapped brand (Apple, Google, None)
5. **tweet_length** – number of characters in the tweet
6. **word_count** – number of words in the tweet**clean_text** – lowerca punctuation/URL/mention-free text
7. **tokens** – tokenized version of clean text
8. **preprocessed_text** – tokenized + stopwords removed + stemme

Data Preprocessing Steps

- Lowercasing- All text was converted to lowercase to ensure consistency in word representation
- Removing Punctuation, Mentions, URLs
- Whitespace Normalization- Extra spaces were cleaned up using regex to make text uniform.
- Tokenization- Cleaned text was split into individual words (tokens), forming a list of tokens for each tweet.
- Stopword Removal- Common English stopwords (e.g., "the", "is", "and") were removed using NLTK's stopword list to reduce noise.
- Stemming- Words were reduced to their root forms using PorterStemmer (e.g., "running" → "run", "tweets" → "tweet").
- Creating Final Preprocessed Text- The cleaned, stemmed tokens were joined back into a single string (preprocessed_text) to be used in vectorization.
- TF-IDF Vectorization- Used TfidfVectorizer with unigrams (single words) to convert text into numerical features for model input.
- Train-Test Split- Final dataset was split into training and testing sets (80/20) for modeling.

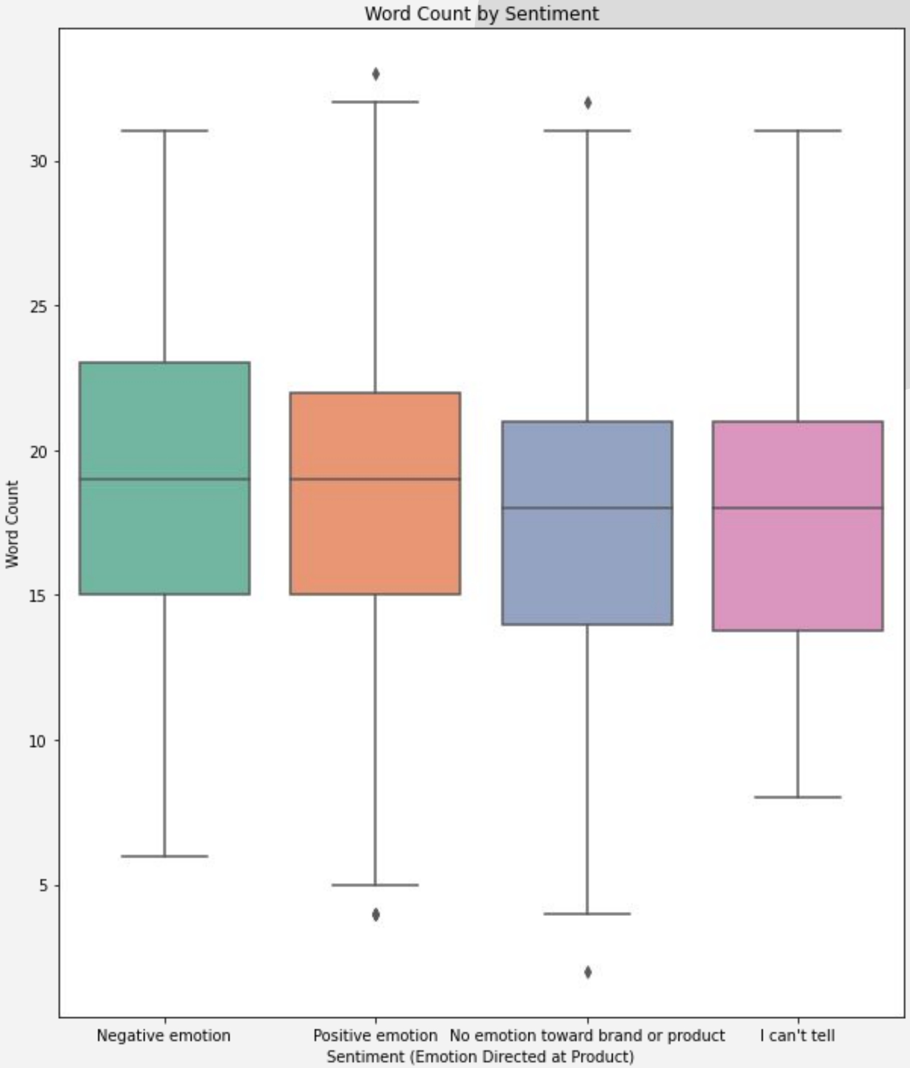
EXPLORATORY DATA ANALYSIS





Word Count by Sentiment

Emotional tweets tend to have more words, suggesting wordiness correlates with sentiment.



Data Preprocessing Steps

Modeling Approach

1. Problem Framing

- Binary Classification: Predict positive (1) vs. negative (0) sentiment.
- Multiclass Classification: Extend to 3 classes (positive, negative, neutral).
- Class Imbalance Addressed: Used class_weight='balanced' and oversampling for fairness.

2. Models Evaluated

Binary Classification:

- Baselines:
    - Naive Bayes (poor negative recall: 2%)
    - Logistic Regression (untuned: 11% negative recall; tuned: 61%)
    - Random Forest (25% negative recall)
    - SVM (best baseline: 58% negative recall, 88% accuracy)

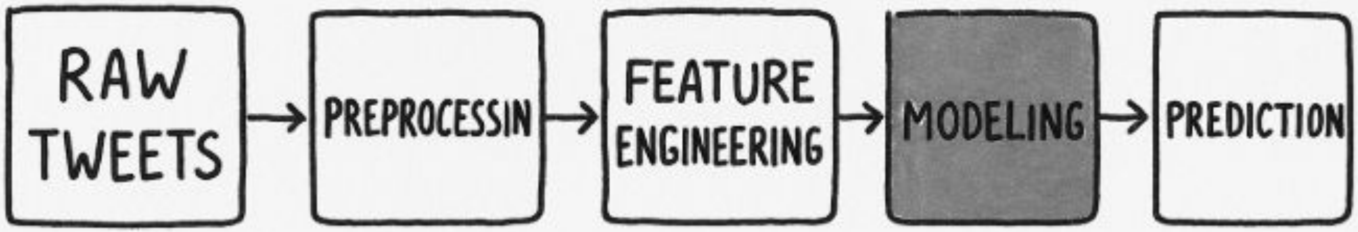
Data Preprocessing Steps

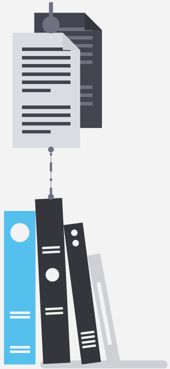
- Neural Network:
  - Architecture: Dense(128) → Dropout(0.3) → Dense(64) → Dropout(0.3) → Sigmoid
  - Result: Severe overfitting (100% train/test accuracy).

Multiclass Classification (positive, negative, neutral):

- Neural Network:
  - Architecture: Dense(64, L2) → Dropout(0.5) → Dense(32, L2) → Dropout(0.5) → Softmax
  - Regularization: L2 penalty + dropout to combat overfitting.
  - Result: 100% test accuracy (potential overfitting despite regularization).

FLOWCHART


RAW TWEETS
PREPROCESSIN
FEATURE ENGINEERING
MODELING
PREDICTION



THANK YOU
ANY
QUESTIONS?