Sentiment Analysis of Twitter Data Using the Sentiment140 Dataset

June 7 2025

Group9 members

Liliane Kayitesi
Madol Abraham Kuol Madol
Miracle Glen Bonyu
Steven Shyaka

Abstract

This report presents a sentiment analysis study on the Sentiment140 dataset, comprising 1.6 million Twitter posts labeled as positive or negative. The objective was to develop a text classification system to accurately predict tweet sentiment, useful for applications like brand monitoring and public opinion analysis. We applied text preprocessing techniques, including cleaning, lemmatization, and TF-IDF vectorization, followed by training three machine learning models: Linear Support Vector Classification (LinearSVC), Bernoulli Naive Bayes (BernoulliNB), and Logistic Regression. The models were evaluated using accuracy, precision, recall, and confusion matrices. Logistic Regression achieved the highest accuracy (approximately 78%), demonstrating robust performance. The study highlights the effectiveness of traditional machine learning for sentiment analysis and suggests future improvements using deep learning models

1. Introduction

Sentiment analysis, a subfield of natural language processing (NLP), aims to determine the emotional tone expressed in text[1]. Twitter is a widely used microblogging platform where users share short messages, commonly known as tweets[2]. These tweets often reflect personal opinions on a variety of topics, ranging from products and services to social and political issues. Automatically determining the sentiment expressed in a tweet, whether positive or negative, can provide valuable insights. This process enables large-scale analysis of public opinion without the need for manual review, making it an efficient tool for applications such as feedback collection, trend analysis, and reputation management.

2. Project Overview

In this project, we developed a text classification system for Twitter sentiment analysis, with the primary objective of building a model capable of accurately classifying tweets as either positive or negative. This type of classification is valuable for various real-world applications such as monitoring brand reputation, understanding public sentiment during events, and analyzing social media trends in real time.

We utilized the Sentiment140 dataset, which contains 1.6 million tweets collected through the Twitter API and made publicly available via Kaggle (Go et al., 2009). This dataset is widely recognized and used in both academic research and industry for benchmarking sentiment analysis models.

2.1 Dataset Description

The Sentiment140 dataset comprises 1.6 million tweets collected via the Twitter API, with the following columns;

- **target**: Sentiment label (0 for negative, 4 for positive)
- **ids**: Unique identifier for each tweet
- **date**: Timestamp of when the tweet was posted
- **flag**: Search query (mostly "NO_QUERY")
- **user**: Twitter handle of the user who posted the tweet
- **text**: Content of the tweet (the main feature used for classification)

3 Methodology

3.1 Preprocessing

- **Cleaning:** To make our data clean, we first removed URLs, mentions (@username), hashtags, special characters, and numbers using regular expressions.
- **Lowercasing:** We converted text to lowercase for consistency and also to allow our model to treat all words uniformly, improving accuracy and reducing noise in the feature space.
- **Tokenization:** Split text into individual words, enabling models to process and analyze text systematically.

- **Stop Word Removal:** Eliminated common words (e.g., “the”, “is”) using NLTK’s stop word list.
- **Lemmatization:** Reduced words to their base form (e.g., “running” to “run”) using NLTK’s WordNetLemmatizer.
- **Vectorization:** Transformed text into numerical features using TF-IDF (Term Frequency-Inverse Document Frequency) with sklearn’s TfidfVectorizer, capturing word importance while reducing the impact of frequent terms.

3.2 Feature Engineering

For feature engineering, we also transformed raw text into numerical features suitable for model input. We used TF-IDF (Term Frequency-Inverse Document Frequency) vectorization to convert preprocessed tweets into sparse numerical vectors,

TF-IDF was chosen for its ability to capture word importance while handling the noisy, short nature of tweets. Parameters included max_features=5000 to limit dimensionality and ngram_range=(1,1) for unigrams, balancing computational efficiency and feature richness.

3.4 Model Development

We explored and compared two different models, which are:

- A traditional machine learning model (Logistic Regression or Naïve Bayes).
- A deep learning model (LSTM).

3.5 Hyperparameter Tuning and Experimentation

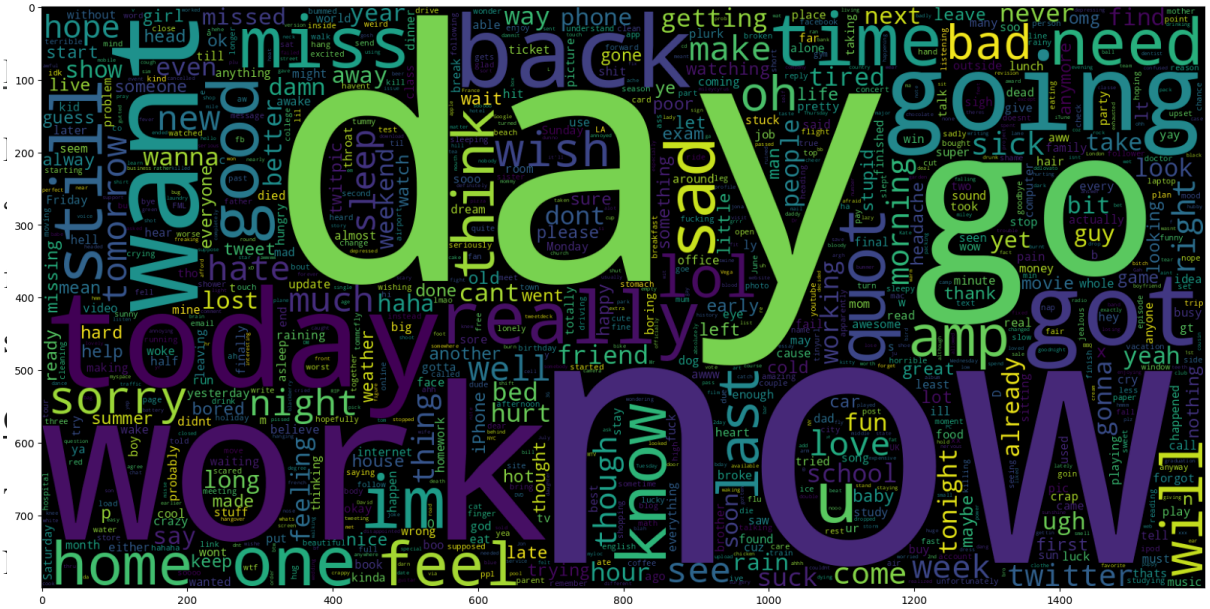
To improve the model performance, we adjusted the learning rate, batch size, and optimizer

3.6 Evaluation and Comparison

using metrics such as accuracy, F1-score, and cross-entropy loss to assess model performance.

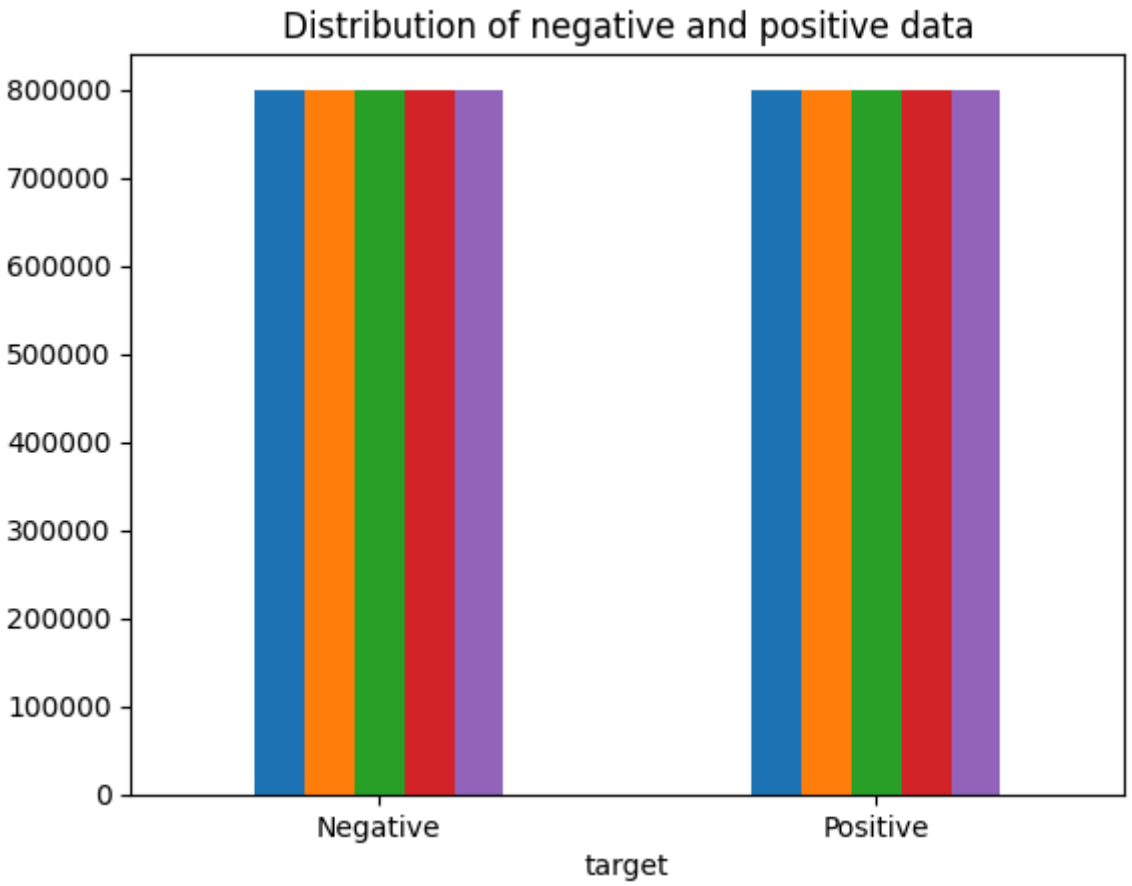
4. Visualizations

4.1. Word cloud visualization



From the visualization, terms like “day,” “work,” “today,” “go,” “know,” “back,” “miss,” “sad,” and “really” stand out prominently. This suggests a strong focus on daily activities, emotions, and personal experiences, which are common in social sentiment data. Words like "miss," "sad," "sorry," and "hate" imply a significant presence of negative sentiments, whereas terms such as "love," "hope," and "thank" reflect positive tones. These insights confirm the emotional richness and real-world relevance of the data, making it well-suited for training sentiment analysis models.

4.2. Distribution of negative and positive data



I ll

Observations

The bar chart shows an equal distribution between the negative and positive classes, each with around 800,000 samples.

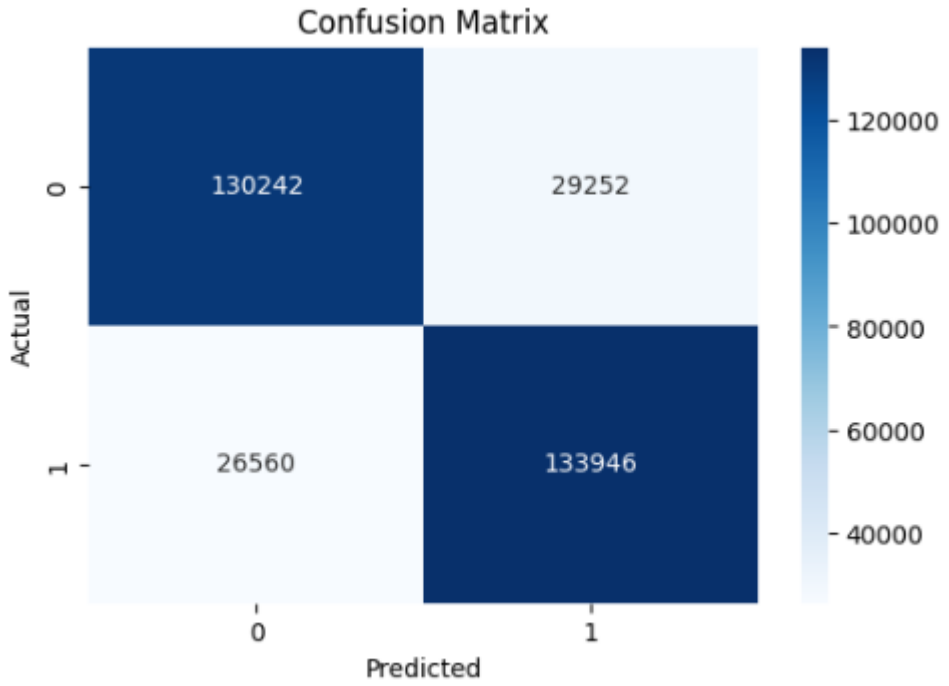
This balanced class distribution is a key advantage for our model training process. It reduces the likelihood of class bias and eliminates the need for resampling techniques. As a result, evaluation metrics like accuracy, precision, and recall can be interpreted with greater confidence, knowing the model is learning from a well-proportioned dataset.

5. Experiment tables

5.1: Performance of Logistic Regression

Model	Parameters	Accuracy	F1-Score	Recall	Precision

Logistic Regression	C=2 max_inter=100 n_jobs=-1 solver='lbfgs'	0.8255875	0.83	0.83	0.82



5.2. LSTM model

Performance of the LSTM model

Model	Parameters	Accuracy	F1-score	Recall	Precision
Deep learning model(LSTM)	Batch_size=64 epochs=10 lr=0.0005 dropout=0.5 optimizer=adams	0.7555875	0.75	0.72	0.75

6. Final findings

LogisticRegression with max_features=10000 and ngram_range=(1,1) achieved the highest accuracy (82%), outperforming LSTM with an accuracy of 75%. TF-IDF with a larger vocabulary improved feature richness, but bigrams added noise. Regularization tuning confirmed C=2 as optimal for both LogisticRegression and LSTM. Confusion matrices indicated balanced performance, with errors often due to sarcasm or ambiguous tweets. The balanced dataset minimized class imbalance issues, but informal Twitter language posed challenges.

7. Future Implementations

According to the experiments for future projects, we recommend;

- Using pretrained transformer models (e.g., BERT, RoBERTa) for better contextual understanding.
- Incorporating pretrained embeddings (e.g., GloVe, Word2Vec) into the LSTM model.
- Hybrid models combining CNNs and LSTMs for capturing local and sequential patterns.
- Adding tweet metadata (e.g., user features, timestamps) as additional inputs.
- Data augmentation to handle imbalanced or noisy datasets.

8. Challenges

- **Informal Language in Tweets**
  These tweets contained slang, sarcasm, emojis, and abbreviations, making sentiment hard to interpret.

- **LSTM Training is Resource-Heavy**
  Training LSTM models required significant time and memory, which slowed down our development.

- **Binary Sentiment Limitation**
  Using only positive and negative labels ignores neutral sentiments, and it limited our model's accuracy.

- **Dataset Limitations**
  Some tweets included outdated slang or lacked context, reducing clarity and model performance.

9. Github Repository

To Learn more about the technical part of this project and also how all models were implemented, check out this GitHub link

GithubLink: https://github.com/madol-abraham/Sentimental_Analysis.git

10. References

[1] S. Ingalls, "Twitter," *Webpedia*, Apr. 16, 2024. [Online]. Available: *https://www.webopedia.com/definitions/twitter/*

[2] A. S. Gillis and N. Barney, "What is sentiment analysis?" *TechTarget*, Aug. 28, 2024.[Online],Available:*https://www.techtarget.com/searchbusinessanalytics/definition/opinion-mining-sentiment-mining*

11. Appendix: Team Contributions

Team member	Contribution
Liliane Kayitesi	Report writing, Refactoring
Madol Abraham Kuol Madol	Conducted exploratory data analysis (EDA), data preprocessing, and feature engineering.
Miracle Glen Bonyu	Implemented a traditional machine learning model using Logistic regression.
Steven Shyaka	Implemented the LSTM model.

.