Sentiment Analysis Project

Author: Jason Park

Project Title:

Sentiment Analysis of Twitter Data using Machine Learning

Contents

1 Problem Statement

With the exponential growth of social media usage, platforms like Twitter have become valuable sources of real-time public opinion. Text sentiment analysis on social networks involves the automatic identification and classification of the emotional tone conveyed in user-generated text, typically into categories such as positive, negative, or neutral. This capability is vital for various applications, including brand monitoring, political opinion tracking, market research, and emergency response.

Given the dynamic and unstructured nature of social media content, sentiment analysis poses several challenges: informal language, slang, abbreviations, spelling errors, and the use of emojis or sarcasm. Therefore, the development of an accurate sentiment classification pipeline requires robust preprocessing and effective machine learning models.

This project focuses on analyzing and classifying tweet sentiment using the Sentiment140 dataset, which contains 1.6 million pre-labeled tweets. Each tweet is annotated with a sentiment label (0 = negative, 4 = positive), along with metadata such as tweet ID, timestamp, query term, user ID, and the tweet text. Neutral tweets (label = 2) were removed to perform binary classification.

The primary goal is to construct a reproducible NLP pipeline that preprocesses raw text, extracts meaningful features using TF-IDF, and applies classification models to distinguish between positive and negative sentiments. The performance of two widely-used classifiers—Multinomial Naive Bayes and Logistic Regression—is compared using accuracy, precision, recall, F1-score, and confusion matrices. Additionally, cross-validation is employed to assess model generalizability.

By demonstrating a complete sentiment classification workflow, this project serves as a strong foundation for future enhancements using more advanced models such as transformers or recurrent neural networks.

2 Data Source

We used the Sentiment140 dataset, which contains 1.6 million labeled tweets. For computational efficiency and faster experimentation, we sampled a balanced subset:

- 50,000 positive and 50,000 negative tweets selected from the training set (total = 100,000)
- An additional test set of 498 tweets: 321 positive and 177 negative

Each tweet includes metadata such as tweet ID, timestamp, user info, and the tweet content.

Label in the dataset	**Explanation**	**Example**
target	Sentiment label of the tweet: 0 for negative, 1 for positive	0 (negative), 1 (positive)
id	Unique identifier of the tweet	1974671194
datetime	Date and time when the tweet was posted	Sat May 30 13:36:31 PDT 2009
query	Search query used to obtain the tweet (often 'NO_QUERY')	NO_QUERY
userid	Username of the Twitter account that posted the tweet	simba98
tweet	Actual text content of the tweet	`@xnausikaax oh no! where did u order from?...`

Table 1: Description of Sentiment140 Dataset Columns

The table below describes the structure of the Sentiment140 dataset used in both training and test sets. For computational reasons, we used a sampled training set of 100,000 tweets (50k positive, 50k negative). The test set includes 498 tweets, with 321 labeled positive and 177 negative.

3 Methodology

The analysis follows a structured Natural Language Processing (NLP) pipeline to classify tweet sentiment. The pipeline consists of the following stages:

1. **Dataset Loading:** The Sentiment140 dataset is loaded using `pandas`. Each tweet record contains a sentiment label (0 = negative, 4 = positive), tweet ID, timestamp, query string, user ID, and the tweet text.

2. **Data Cleaning and Preprocessing:** The raw tweet text undergoes extensive preprocessing to prepare it for feature extraction:

   - HTML tags, URLs, mentions (@username), and special characters are removed using regular expressions and BeautifulSoup.
   - Text is normalized by converting to lowercase, stripping extra whitespace, and removing accented characters using `unicodedata`.
   - Contractions are expanded using the `contractions` package.
   - Tokenization, stopword removal, and lemmatization are performed using the NLTK library to reduce vocabulary size and standardize word forms.

3. **Label Transformation:** Sentiment labels are transformed from {0, 2, 4} to binary values:

   - 0 → `negative`, 4 → `positive`
   - All tweets labeled with 2 (neutral) are removed to ensure binary classification

4. **Feature Extraction:** Cleaned tweets are transformed into numerical vectors using TF-IDF vectorization. The following settings are used:

   - Both unigrams and bigrams (`ngram_range=(1,2)`)
   - Sublinear term frequency scaling
   - Minimum document frequency of 5
   - Maximum number of features: 10,000

5. **Model Training:** Two baseline classifiers were trained:

   - **Multinomial Naive Bayes** using Laplace smoothing (`alpha=1.0`)
   - **Logistic Regression** with `L2` regularization, `C=1.0`, `solver='liblinear'`, and `max_iter=500`

   Models were trained on the sampled training set and used to generate predictions on the test set.

6. **Model Evaluation:** Models were evaluated using the following metrics:

   - `classification_report()`: Reports accuracy, precision, recall, and F1-score
   - Confusion matrix visualizations were generated for both classifiers

7. **5-Fold Cross-Validation (CV):** To assess model robustness and generalization, we performed 5-fold cross-validation on the training set. Accuracy scores were averaged across folds to compare models.

Models Highlights

We trained and evaluated two classifiers:

- **Multinomial Naive Bayes**: using Laplace smoothing (`alpha=1.0`)
- **Logistic Regression**: with `L2` penalty, `C=1`, `max_iter=500`

4 Evaluation and Final Results

4.1 Exploratory Data Analysis

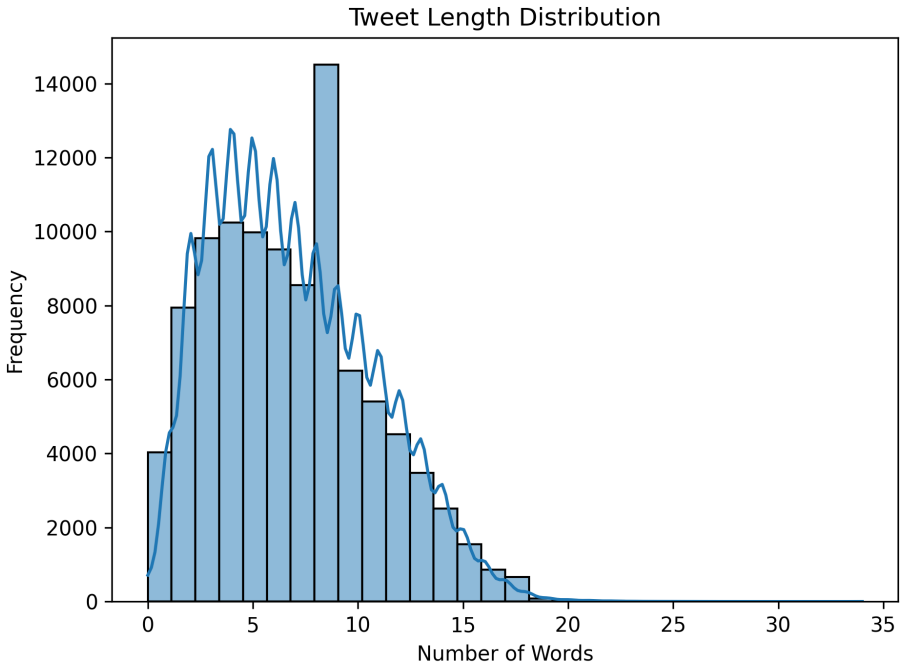

Figure 1: Tweet Length Distribution. Most tweets contain between 3 to 15 words, consistent with the short-form nature of Twitter posts. This insight helps inform preprocessing and feature extraction decisions.

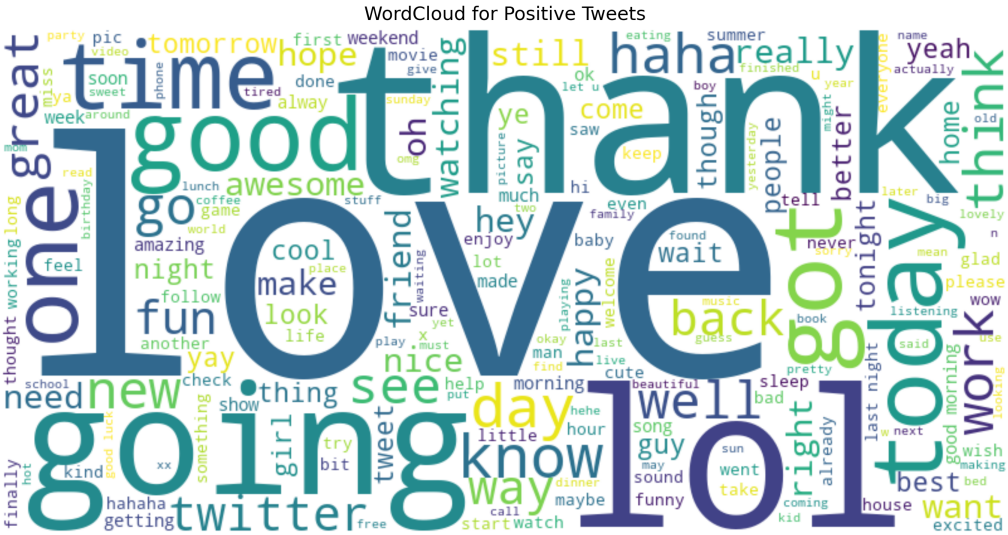

Figure 2: WordCloud for Positive Tweets. Frequently used words in positive tweets include *love*, *thank*, *fun*, and *great*, reflecting overall positive sentiment and expressions of enjoyment or gratitude.

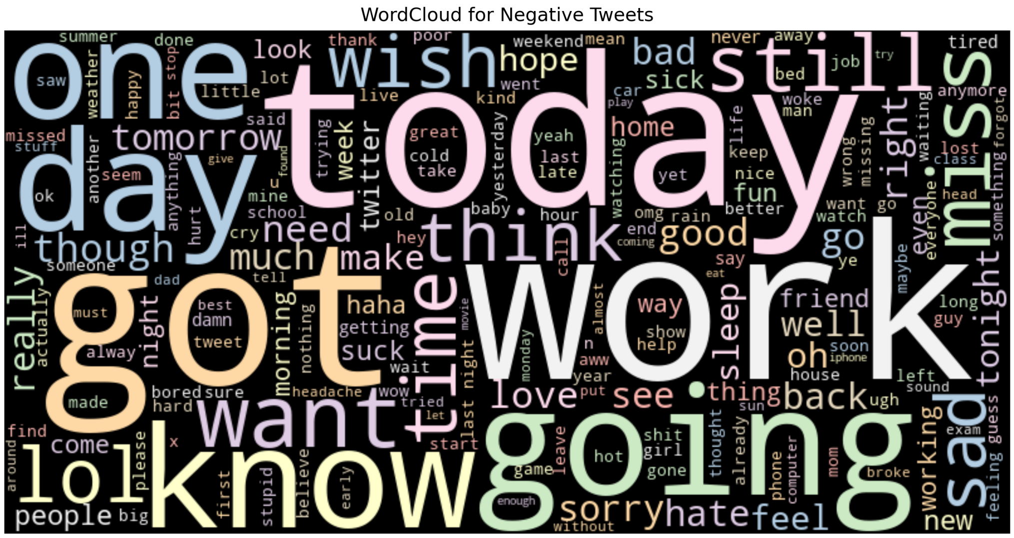

Figure 3: WordCloud for Negative Tweets. Common terms include *work*, *miss*, *hate*, and *tired*, indicating topics related to stress, fatigue, or dissatisfaction.

4.2 Confusion Matrices

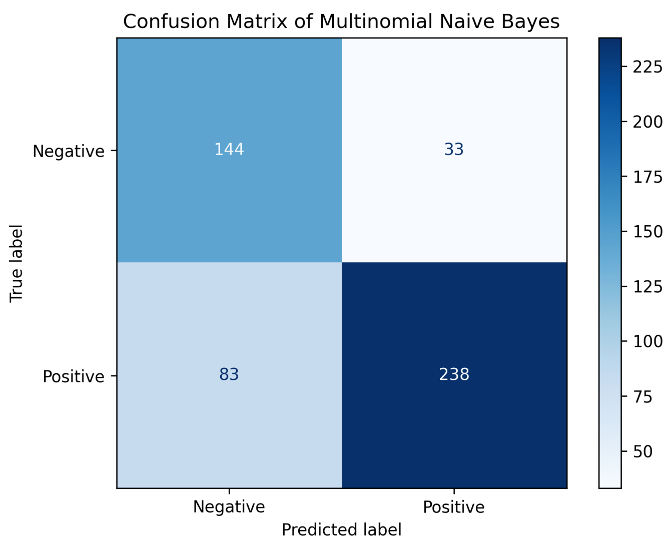

Figure 4: Confusion Matrix of Multinomial Naive Bayes. The model correctly classified 144 negative and 238 positive tweets, while misclassifying 116 tweets in total. Performance is decent, with slightly more errors on the positive class.

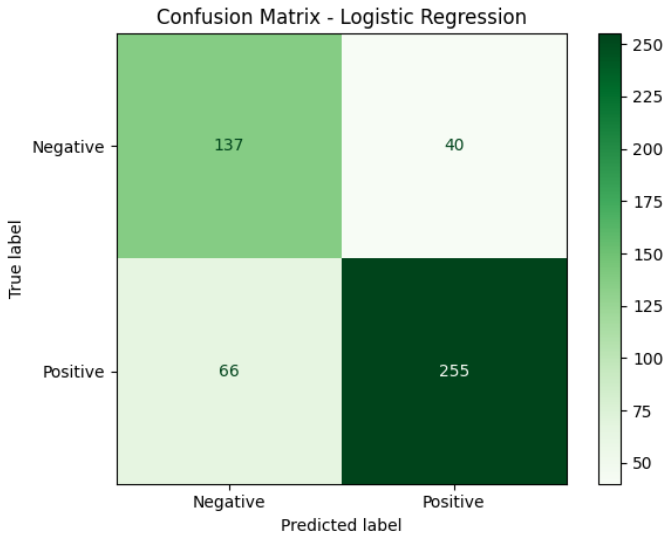

Figure 5: Confusion Matrix of Logistic Regression. This model showed improved classification with fewer misclassifications overall. It correctly predicted 137 negative and 255 positive tweets, suggesting better balance and precision than Naive Bayes.

4.3 Performance Metrics

The classification performance of both models is summarized below using four standard evaluation metrics: accuracy, precision, recall, and F1-score. These metrics are calculated based on predictions made on the test set (498 tweets).

- **Multinomial Naive Bayes:**
  - Accuracy: 0.7651
  - Precision: 0.7680
  - Recall: 0.7651
  - F1-Score: 0.7651

  Multinomial Naive Bayes performs well given its simplicity and speed. Its F1-score suggests a good balance between precision and recall, but its performance is slightly lower than Logistic Regression.

- **Logistic Regression:**
  - Accuracy: 0.7872
  - Precision: 0.7888
  - Recall: 0.7872
  - F1-Score: 0.7872

  Logistic Regression outperforms Naive Bayes on all metrics. The higher precision and recall indicate that it is better at correctly identifying both positive and negative sentiments, making it a stronger candidate for practical deployment.

Overall, both models provide competitive performance on the sentiment classification task, with Logistic Regression showing a clear edge in prediction quality.

4.4 Cross-Validation Results

To evaluate the generalizability of the models beyond a single train-test split, 5-fold cross-validation (CV) was conducted on the training dataset (100,000 tweets). This method splits the training set into five equal parts, training the model on four folds and validating on the remaining one, rotating through each fold. The average accuracy across folds provides a more robust estimate of model performance on unseen data.

- **Multinomial Naive Bayes:**
  - Fold Accuracies: [0.74485, 0.75015, 0.75030, 0.74590, 0.74500]
  - Average CV Accuracy: **0.7472**

  These scores indicate consistent performance across all folds, with Naive Bayes maintaining a stable accuracy near 74.7%.

- **Logistic Regression:**
  - Fold Accuracies: [0.75960, 0.76170, 0.76405, 0.76235, 0.75920]
  - Average CV Accuracy: **0.7614**

  Logistic Regression again outperformed Naive Bayes with slightly higher and more consistent accuracy across folds, confirming its reliability on this sentiment classification task.

The cross-validation results reinforce earlier findings from test set evaluation, showing that Logistic Regression has better generalization capability than Naive Bayes in this context.

5 Conclusion

This project demonstrated a complete NLP pipeline for binary sentiment classification of tweets using the Sentiment140 dataset. After preprocessing, feature extraction using TF-IDF, and training both Multinomial Naive Bayes and Logistic Regression models, we found that both models performed reasonably well. Logistic Regression slightly outperformed Naive Bayes across all evaluation metrics, including accuracy, precision, recall, and F1-score.

The 5-fold cross-validation results confirmed that Logistic Regression generalizes better and maintains stable performance across multiple data splits. These findings suggest that simple linear models can provide strong baselines for sentiment analysis on social media data.

Potential applications of this sentiment analysis model include:

- **Business Insight:** Companies can analyze customer feedback in real time to detect dissatisfaction or praise, helping improve product quality and customer service.
- **Political Sentiment Analysis:** Governments or researchers can use the model to track voter opinions during election campaigns or public reactions to policy announcements.

Future work could explore more sophisticated models, such as transformer-based architectures (e.g., BERT), or expand the analysis to include neutral sentiment and real-time tweet streams.

6 Novelty and Contribution

This project highlights how a simple yet effective NLP pipeline—consisting of text preprocessing, TF-IDF vectorization, and classical machine learning classifiers—can produce strong baseline performance on a large-scale and noisy dataset such as Sentiment140. Despite its simplicity, the approach offers practical value in real-world sentiment analysis tasks.

Key contributions of this work include:

- A reproducible machine learning pipeline for binary sentiment classification of tweets
- Comparative evaluation of Multinomial Naive Bayes and Logistic Regression using classification metrics and 5-fold cross-validation
- A demonstration that classical models can still perform competitively on social media data, forming a foundation for future deep learning-based approaches
- Comprehensive visual and statistical EDA to understand dataset structure and sentiment distribution

7 Team Member Responsibilities

This is an individual project conducted by Jason Park. All tasks—including data preprocessing, model implementation, hyperparameter tuning, evaluation, result visualization, and report writing—were completed independently.

8 References

1. Go, Alec, Richa Bhayani, and Lei Huang. "Twitter Sentiment Classification using Distant Supervision." Stanford University, 2009.
   `http://cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf`
2. Pedregosa, Fabian, et al. "Scikit-learn: Machine Learning in Python." Journal of Machine Learning Research 12 (2011): 2825–2830.
3. Bird, Steven, Edward Loper, and Ewan Klein. "Natural Language Processing with Python." O'Reilly Media Inc., 2009. (Used for NLTK-based preprocessing)
4. Kaggle Dataset: Sentiment140 Twitter Dataset
   `https://www.kaggle.com/datasets/kazanova/sentiment140`