

Reproducing Airline Twitter Sentiment Classification with TF-IDF + Linear SVM

Replication of “A Comparative Study of Sentiment Analysis Using NLP and Different Machine Learning Techniques on US Airline Twitter Data”
(with a custom Kaggle-based pipeline).

Safiya Khanum, Adhnan ali Rashid | Project—FCSS

Problem Statement

Research question. Can we reproduce the sentiment classification performance reported in a prior study on US airline tweets, and how does a small change in preprocessing affect results?

Motivation. Airline-related tweets reflect real passenger experiences such as delays, cancellations, and customer support interactions. Automatically classifying these messages into *negative*, *neutral*, and *positive* sentiment enables large-scale analysis of public opinion. Converting this stream into *negative* / *neutral* / *positive* sentiment helps summarize public opinion and identify recurring pain points.

Replication goal. We follow the original paper’s overall workflow to verify that the reported performance generalizes to the same dataset family, to see whether a seemingly minor decision measurably shifts accuracy and Macro-F1.

Data

Dataset. Twitter US Airline Sentiment (Kaggle / CrowdFlower): ~14,640 tweets across 6 airlines, labeled *negative/neutral/positive* (imbalanced). We use the same core columns emphasized in the original study: *text*, *airline_sentiment*, *airline*, *negativereason*.

Before modeling, we summarize the label distribution and how sentiment varies across airlines. This motivates using stratified cross-validation and reporting Macro-F1 (more informative than accuracy under class imbalance).

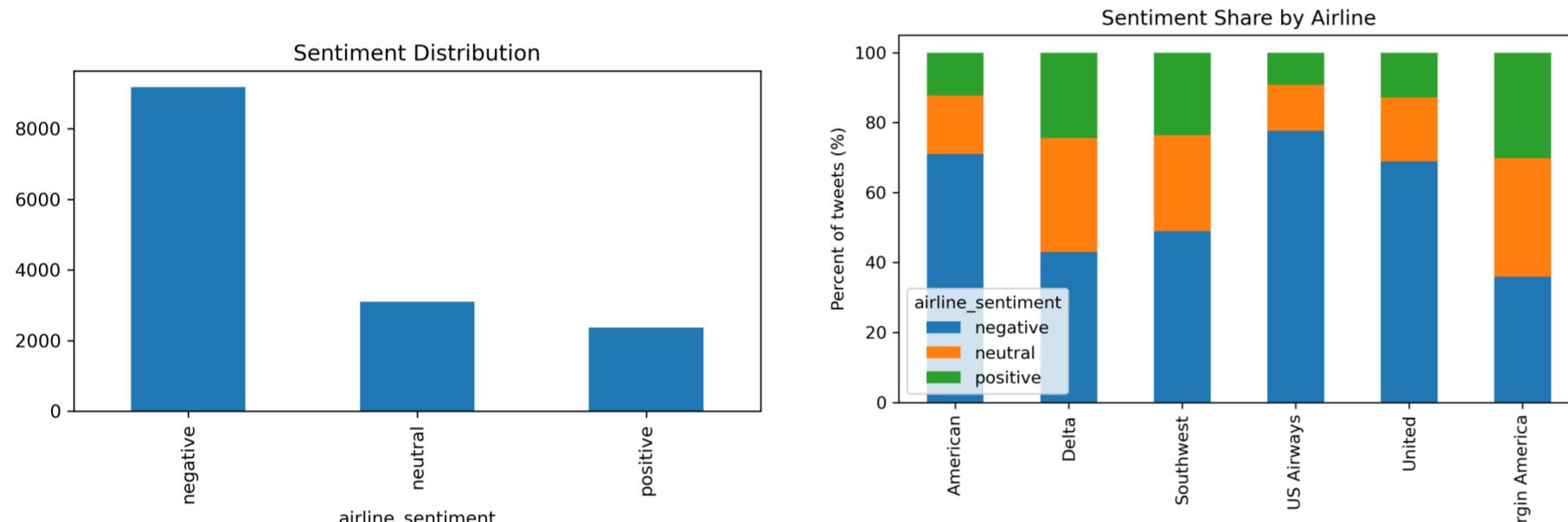


Figure: Data overview. Left: class imbalance (negative dominates). Right: sentiment share varies by airline, motivating stratified evaluation and class-balanced metrics.

Exploratory Insights

Rather than treating sentiment labels as abstract classes, we first inspect what *people complain about*. The *negativereason* field provides interpretable categories (e.g., delays, cancellations, service issues) that help explain why the negative class dominates and where models are likely to confuse neutral vs negative.

These patterns suggest the task is driven by a mix of operational issues (delays/cancellations) and customer-service language, which can be subtle and context-dependent in short tweets.

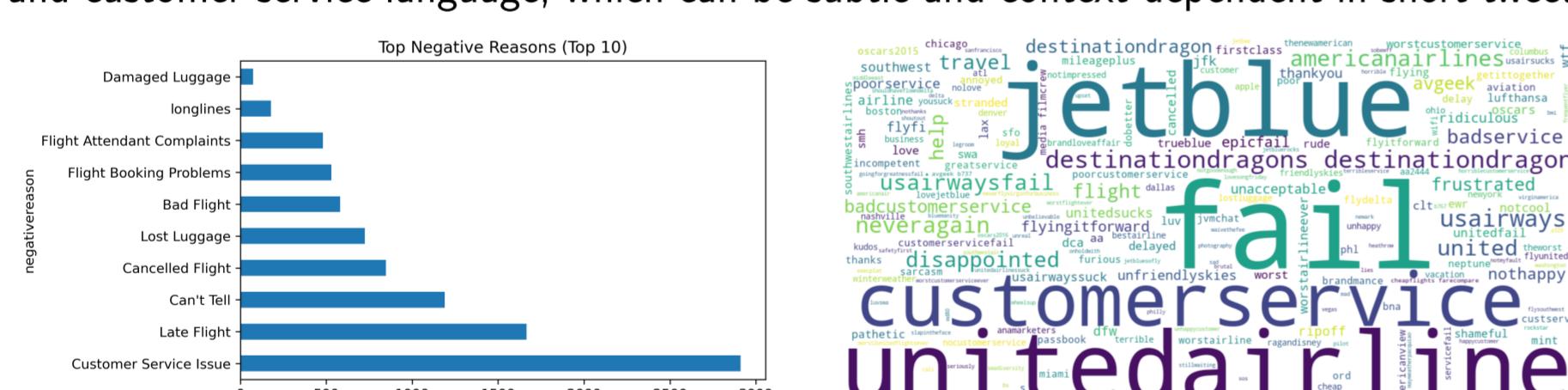


Figure: Left : Top negative reasons (Top 10). Right: Complaints cluster around service issues (e.g., delays, cancellations, customer support), which dominate the negative class.

Preprocessing Change (Key Replication Twist)

We replicate the standard pipeline (clean → vectorize → classify), but introduce a small, targeted change: **remove stopwords while preserving negations** (*not/no/nor*). Intuitively, negations flip sentiment (“not good”), so removing them can erase signal.

Cleaning (clean_text2b). Tweets are lowercased and stripped of URLs/@mentions. We keep alphabetic tokens, remove stopwords except negations (*not/no/nor*), and drop very short tokens (≥ 3 letters) to reduce noise while preserving sentiment flips.

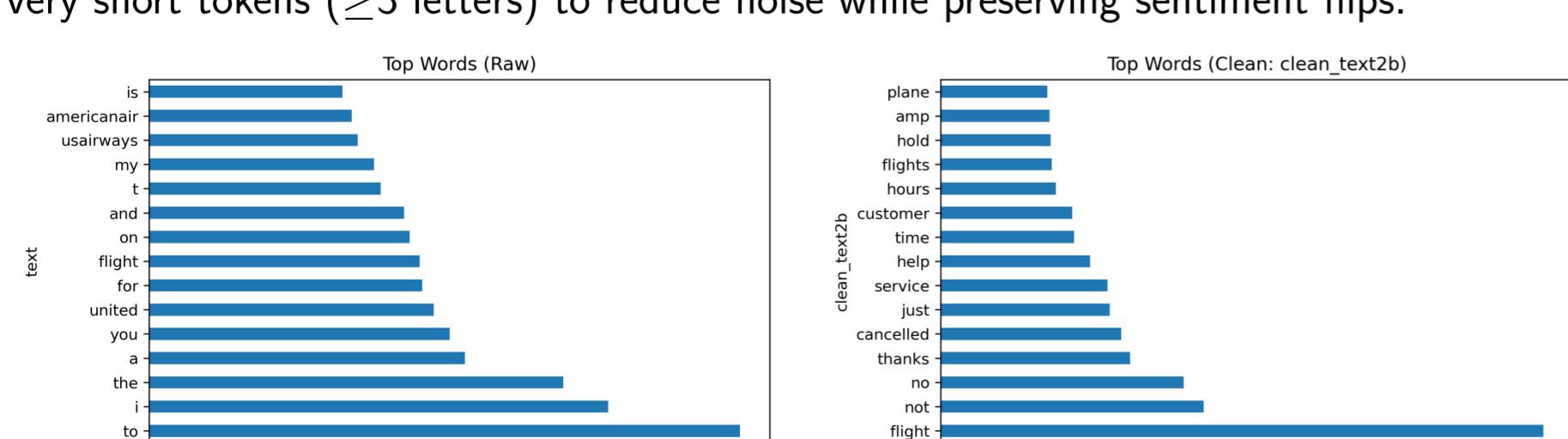


Figure: Effect of cleaning on vocabulary. Left: raw tweets include noisy/high-frequency tokens. Right: after lightweight cleaning, frequent terms are more content-bearing, improving TF-IDF representations.

Methods

We reproduce the original study’s pipeline (preprocess → vectorize → train → evaluate), and additionally report **Macro-F1** to account for class imbalance.

- ▶ **Preprocessing:** Tweets are cleaned (*clean_text2b*) by removing URLs, mentions, symbols, stopwords, while preserving negations (e.g., *not*, *no*).
- ▶ **Vectorization:** Text is converted into numerical features using TF-IDF (scikit-learn).
- ▶ **Models:** Linear SVM ($C = 0.5$) is used as the main classifier; Multinomial Naive Bayes ($\alpha = 0.1$) serves as a baseline.
- ▶ **Evaluation:** Performance is assessed using 10-fold stratified cross-validation with Accuracy and Macro-F1 as metrics.

The original paper reports approximately 77% accuracy using a 75/25 train–test split and weighted metrics, while our evaluation uses cross-validation and class-balanced Macro-F1.

Results

Best pipeline (TF-IDF + Linear SVM, C=0.5): Accuracy 78.64%, Macro-F1 0.713. A key replication insight is that **stopword handling matters**: removing stopwords *without* preserving negations reduced performance, while keeping negations partially recovered it.

Model	Accuracy (%)	Macro-F1
Linear SVM (C=0.5)	78.64	0.713
Multinomial NB ($\alpha=0.1$)	72.22	0.559

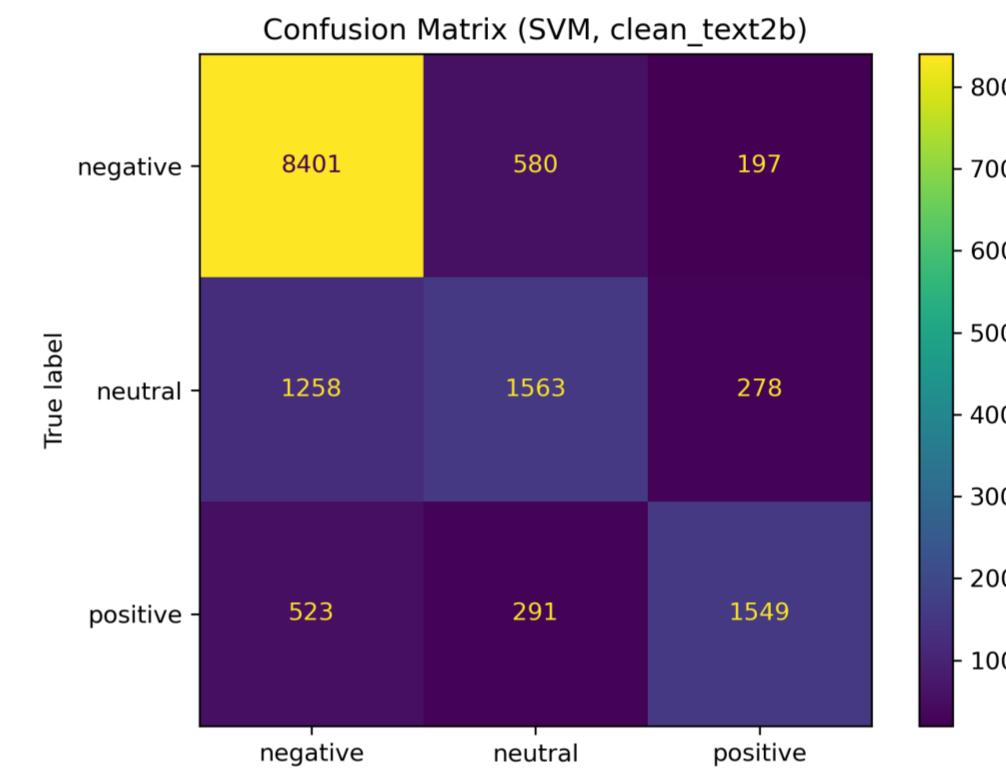


Figure: Confusion matrix for the best model (Linear SVM). Most errors occur between *neutral* and *positive/negative*, reflecting subtle wording in short tweets.

Replication Comparison

Study / Setup	Accuracy	F1
Original paper (SVM/LR, BoW/TF-IDF, 75/25 split)	≈ 77%	weighted
Our replication (TF-IDF + Linear SVM, 10-fold stratified CV)	78.64%	Macro-F1 = 0.713

Note: Results are comparable in scale, though not numerically identical due to different evaluation protocols (CV vs split) and F1 averaging..

Conclusions

- ▶ We replicate the core finding that **linear models (SVM)** perform strongly on airline tweet sentiment with bag-of-words style features.
- ▶ A **small preprocessing choice** (stopwords + negation preservation) changes results noticeably.
- ▶ The final pipeline is **transparent, lightweight, and reproducible** while achieving **78.64%** accuracy under stratified CV.

Limitations & Future Work

- ▶ Accuracy can look optimistic under class imbalance; Macro-F1 is more informative.
- ▶ CV vs train/test split makes direct numerical comparison to the original paper imperfect.
- ▶ Tweets are short/noisy; sarcasm and missing context remain hard for linear models.
- ▶ **Next:** try Logistic Regression + calibration, then compare with transformer baselines (reported separately).

References

1. M. T. H. K. Tusu and M. T. Islam, *A Comparative Study of Sentiment Analysis Using NLP and Different Machine Learning Techniques on US Airline Twitter Data*, 2021.
2. Kaggle / CrowdFlower: Twitter US Airline Sentiment dataset.
3. scikit-learn documentation (TF-IDF, LinearSVC, StratifiedKFold).
4. Anaconda Assistant 4.48.0