

Comparative Study of Classical Machine Learning and Transformer-Based Models for Sentiment Analysis

1. Introduction

Sentiment analysis is a fundamental Natural Language Processing (NLP) task widely used for understanding user opinions in applications such as product reviews, customer feedback, and social media analysis.

This project presents a **comprehensive comparison of classical machine learning models and transformer-based deep learning models** for sentiment analysis. The comparison goes beyond accuracy metrics and evaluates **prediction correctness, semantic understanding, training efficiency, and computational cost**.

A key motivation of this study is to demonstrate that **models with similar accuracy scores can produce significantly different real-world predictions**.

2. Dataset and Text Preprocessing

The dataset consists of labeled text reviews categorized as **Positive** and **Negative** sentiments.

Preprocessing Steps

- Contraction expansion (e.g., *can't* → *cannot*)
- Tokenization
- Stopword removal
- Stemming
- TF-IDF vectorization (for classical models)

Transformer-based models used tokenizer-driven preprocessing without manual linguistic transformations.

3. Classical Machine Learning Models

The following classical algorithms were evaluated using TF-IDF feature representations:

- Logistic Regression with Cross-Validation
- Linear Support Vector Machine

- Multinomial Naive Bayes
- Gaussian Naive Bayes (experimental)
- Random Forest
- XGBoost (baseline configuration)

Model Performance Summary

Model	Accuracy (%)	Training Time (sec)	Prediction Time (sec)
Logistic Regression (CV)	89.65	68.69	0.005
Linear SVM	89.18	1.20	0.005
Multinomial Naive Bayes	86.27	0.39	0.013
XGBoost (Baseline)	85.71	372.25	3.55
Random Forest	85.11	167.04	1.37
Gaussian Naive Bayes	69.69	79.41	10.38

4. Important Observation: Accuracy vs Predictive Quality

Although **Random Forest** and **XGBoost** achieved approximately 85% accuracy, qualitative evaluation revealed that both models **misclassified complex negative reviews as positive**.

This behaviour occurred due to:

- Over-dependence on word frequency
- Inability to capture negations and contextual meaning
- Surface-level feature understanding

Key Insight

High accuracy does not necessarily imply correct or reliable sentiment predictions in real-world NLP tasks.

5. Transformer-Based Sentiment Analysis (**DistilBERT**)

To address the semantic limitations of classical models, transformer-based models were evaluated using **DistilBERT**.

5.1 Full Fine-Tuning Results

Metric	Value
Validation Accuracy	89.11%
F1 Score	0.8906
Trainable Parameters	66.9 Million

Training Time

14.92 minutes

Full fine-tuning provided strong contextual understanding but at high computational cost.

6. Parameter-Efficient Fine-Tuning with LoRA

LoRA (Low-Rank Adaptation) fine-tuning was applied to DistilBERT, freezing base model weights and training only a small number of additional parameters.

6.1 LoRA Fine-Tuning Results

Metric	Value
Validation Accuracy	87.11%
F1 Score	0.8732
Trainable Parameters	739 Thousand
Training Time	4.84 minutes

7. Efficiency Comparison: Full Fine-Tuning vs LoRA

Efficiency Metric	Improvement
Training Speed	3.08× Faster
Trainable Parameter Reduction	98.90% Fewer Parameters

Key Insight

LoRA achieves near-transformer-level performance while drastically reducing training time and resource requirements.

8. Overall Model Comparison

Model Category	Strengths	Limitations
Classical ML	Fast inference, simple deployment	Weak semantic understanding
RF / XGBoost	Structured feature handling	Poor contextual sentiment detection
Full DistilBERT	Best semantic accuracy	High compute and memory cost
LoRA DistilBERT	Efficient and scalable	Slight accuracy trade-off

9. Conclusion

This project demonstrates that:

- Accuracy alone is insufficient for evaluating NLP sentiment models
- Tree-based models can misclassify context-heavy reviews
- Transformer-based architectures significantly improve sentiment understanding
- LoRA offers an excellent balance between performance and efficiency