P15: Sentiment Classification of Texts

CS485 – Project Report

Zervos Spiridon Chrisovalantis (csd4878)
Drakakis Rafail (csd5310)

May 25, 2025

Abstract

We implement and compare three classes of sentiment-classification pipelines on the IMDb Movie Reviews dataset:

1. *Classical ML* using TF–IDF (Term Frequency–Inverse Document Frequency) and Logistic Regression, Naive Bayes, Linear SVM
2. *Deep Learning* using PyTorch LSTM and 1D-CNN with learned embeddings

We evaluate each on accuracy, precision/recall/F1 (for classical), and inference or epoch time.

Contents

1 Introduction

Sentiment analysis classifies text as positive or negative. We use the standard IMDb review benchmark to compare:

- **Classical ML**: TF–IDF $\rightarrow$ Logistic Regression / Naive Bayes / SVM
- **Deep Learning**: LSTM & 1D-CNN over a learned embedding layer

2 Dataset

- **Name:** IMDb Movie Reviews
- **Size:** 50 000 total (25 000 train, 25 000 test)
- **Labels:** Positive (1) / Negative (0)
- **Source:** `https://ai.stanford.edu/~amaas/data/sentiment/`

3 Implementation Details

3.1 Preprocessing & Vocabulary

- NLTK downloads guarded for `punkt`, `punkt_tab`, `stopwords`.
- Lowercasing, tokenization (`word_tokenize`), alpha-filter, stopword removal.
- For deep models: build vocab from training tokens (min_freq=2, max_size=20 000), pad/truncate to length 200.

3.2 Feature Extraction

- **Classical:** TF–IDF (unigrams, max_features=5 000).
- **Deep:** learned `nn.Embedding` (dim=100).

3.3 Models & Hyperparams

LogisticRegression $C = 1.0$, ℓ_2, max_iter=1000.

MultinomialNB $\alpha = 1.0$.

LinearSVC $C = 1.0$.

LSTM embed_dim=100, hidden_dim=128, single layer, trained 5 epochs, Adam lr=1e-3.

CNN embed_dim=100, 3 conv-filters (sizes 3,4,5) each 100 channels, train 5 epochs.

4 Results

4.1 Classical ML

Table 1: Accuracy & Inference Time (ms/sample) on 25 000 review test set

Model	Accuracy	Time (ms/sample)
Logistic Regression	0.880	0.47
Multinomial Naïve Bayes	0.840	0.13
Linear SVM	0.863	0.65

Table 2: Classification report for Logistic Regression

Class	Precision	Recall	F1-score	Support
Negative	0.88	0.88	0.88	12 500
Positive	0.88	0.88	0.88	12 500
Overall accuracy: 0.88				

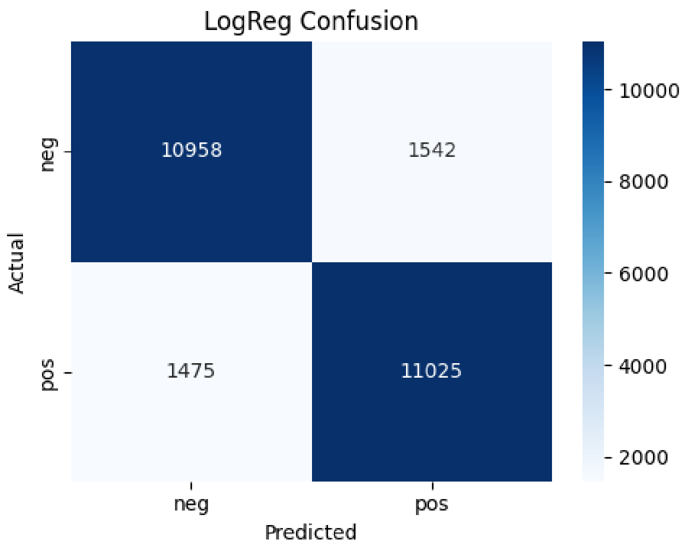

Figure 1: Logistic Regression confusion matrix: TN=10958, FP=1542, FN=1475, TP=11025.

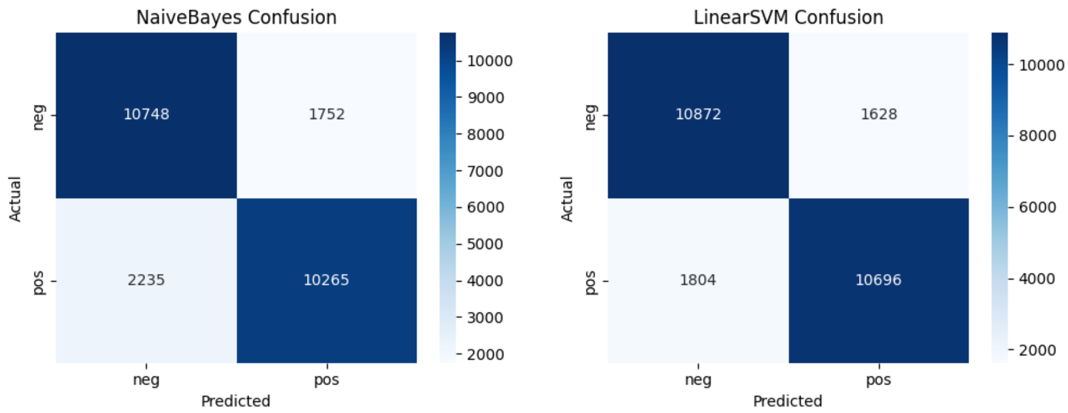

Figure 2: Confusion matrices for Multinomial Naïve Bayes (left) and Linear SVM (right).

4.2 Deep Learning

Table 3: Test Accuracy after 5 epochs (embed_dim=100)

Model	Test Accuracy	Notes
LSTM	0.511	Final hidden-state classifier; underperforms
CNN	0.856	Global max-pooled conv features

Both models were trained on batches of 64, padded to length 200, using Adam (lr=1e-3). No pretrained embeddings were used. LSTM underperformed significantly, due to insufficient learning.

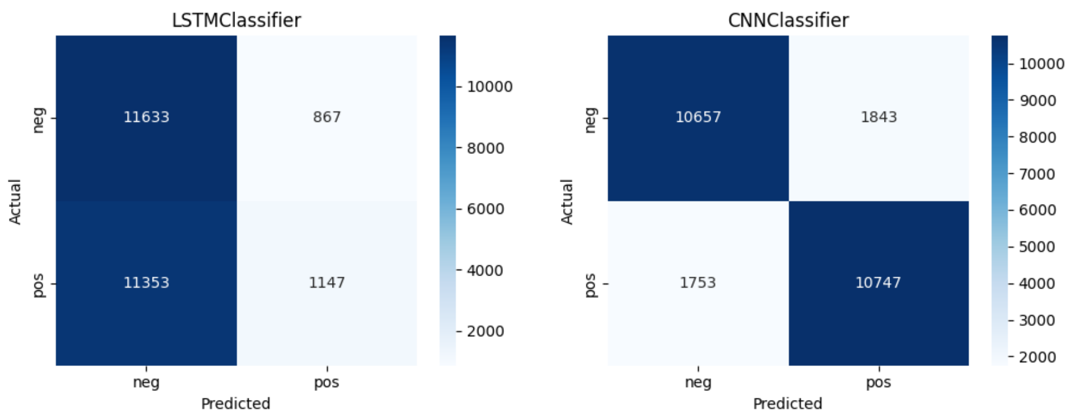

Figure 3: Confusion matrices for LSTM (left) and CNN (right).

5 Discussion

- **Classical vs. Deep:**
  - TF–IDF and Logistic Regression is extremely fast at inference (0.47 ms/sample) with 0.88 accuracy.

  - CNN improves over classical (0.856 acc), but LSTM underperforms (0.511 acc).

- **Error Analysis:**
  - Classical models struggle with negation ("not good") and sarcasm.
  - LSTM has a problem with vanishing gradients and poor convergence.
  - Short reviews (<20 words) are most error-prone across all models.

- **Practical trade-offs:**
  - For real-time scoring, Logistic Regression or CNN is better.

6 Conclusion

We implemented a single-script pipeline that runs classical and deep methods end-to-end on the IMDb dataset. While TF–IDF and Logistic Regression remains a strong, fast baseline (0.88 acc) and CNNs improve slightly (0.856). The LSTM model, despite theoretical potential, failed to train effectively.