Warming up the Mic: Build your own speech to text engine

Week–1 Coding Task Assignment Report

Contents

Mentor: Aditiya Sanapala
Mentee: Jayent Dev
Roll Number: 24B1232

1 Introduction

Sentiment analysis is a core task in Natural Language Processing (NLP) that focuses on determining the polarity of textual data. In this work, a classical machine learning pipeline is implemented for binary sentiment classification on the IMDb movie reviews dataset using TF–IDF feature extraction and multiple supervised learning algorithms.

2 Dataset Description

The IMDb dataset consists of two CSV files:

- **Training set**: 25,000 labeled reviews
- **Test set**: 25,000 labeled reviews

Each review is associated with a binary sentiment label, where 0 denotes negative sentiment and 1 denotes positive sentiment.

3 Text Preprocessing

The following preprocessing steps were applied to both training and test datasets:

- Conversion to lowercase
- Removal of punctuation and special characters
- Removal of HTML tags
- Removal of URLs
- Removal of extra whitespaces
- Tokenization using `word_tokenize`
- Stopword removal
- Stemming using the Porter Stemmer

4 Feature Extraction using TF–IDF

TF–IDF was used to convert text into numerical feature vectors. The vectorizer was fitted only on the training dataset to prevent information leakage. Both unigrams and bigrams were included in the feature representation.

5 Machine Learning Models

The following supervised learning models were trained using the TF–IDF feature matrix:

1. Logistic Regression
2. Softmax Regression (Multinomial Logistic Regression)
3. Naive Bayes with Laplace Smoothing
4. Linear Support Vector Machine (SVM)
5. Ridge Classifier
6. Stochastic Gradient Descent (SGD) Classifier
7. Random Forest Classifier
8. Gradient Boosting Classifier

6 Evaluation Metrics

Model performance was evaluated on the test dataset using the following metrics: Accuracy, Precision, Recall, F1-score, ROC–AUC, Log Loss, Matthews Correlation Coefficient (MCC), and BLEU score. The F1-score was used as the primary metric for model selection.

7 Results and Comparison

Model	**Acc.**	**Prec.**	**Rec.**	**F1**	**ROC–AUC**	**Log Loss**	**MCC**	**BLEU**
Logistic Regression	0.8875	0.8862	0.8892	0.8877	0.9551	0.3270	0.7750	0.1578
Softmax Regression	**0.8897**	**0.8911**	0.8879	**0.8895**	**0.9568**	**0.2985**	**0.7794**	**0.1582**
Naive Bayes	0.8606	0.8781	0.8376	0.8574	0.9354	0.3790	0.7220	0.1530
Linear SVM	0.8830	0.8904	0.8734	0.8818	–	–	0.7661	0.1570
Ridge Classifier	0.8820	0.8875	0.8750	0.8812	–	–	0.7641	0.1568
SGD Classifier	0.8803	0.8780	0.8834	0.8807	0.9500	0.3813	0.7607	0.1565
Random Forest	0.8582	0.8682	0.8446	0.8562	0.9342	0.4531	0.7167	0.1526
Gradient Boosting	0.8086	0.7768	0.8660	0.8190	0.8950	0.4553	0.6213	0.1438

Table 1: Performance Comparison of TF–IDF Based Models

8 Best Model Selection

Based on the evaluation results, **Softmax Regression** achieved the highest F1-score of **0.8895**, along with strong performance across other metrics such as accuracy, ROC–AUC, MCC, and log loss. Therefore, Softmax Regression was selected as the final TF–IDF-based sentiment classification model.

9 Conclusion

This study demonstrates that classical linear models combined with TF–IDF feature extraction are highly effective for sentiment analysis on large-scale text datasets. Among all evaluated models, Softmax Regression provided the best balance between precision and recall, making it the most suitable choice for this task. The results highlight the importance of feature representation and metric-based evaluation in text classification problems.

10 GitHub Repo Link

The asignment solution is done inside the week-1 folder and the file is named as Coding_Task_Solutions and it is a Jupyter notebook. I am attaching the GitHub repo link below :
GitHub link = WiDS-2025-Speech_to_text_engine