IST.664.U801.NATURAL LANGUAGE PROCESSING- FINAL PROJECT

Sai Akash Addala
202315111
Yaswanth LalpetVari
745216548

Abstract

This project analyses sentiment using the Rotten Tomatoes dataset from the Kaggle competition.
The primary objective is to classify movie review phrases into five sentiment categories: neutral, slightly positive, positive, negative, and relatively negative. Various feature engineering strategies,

Preprocessing techniques and machine learning models were used and evaluated. The best-performing model was logistic regression, which had an accuracy rate of 62%. This article presents the methodology, experiments, and findings, emphasising the importance of feature engineering and preprocessing in improving classification performance.

1. Introduction

Sentiment analysis, a subfield of natural language processing (NLP), is concerned with determining a document's tone.It may be applied to product evaluations, social media monitoring, and customer feedback analysis.
This study makes use of the Rotten Tomatoes dataset, which comprises sentiment-labeled movie review terms.The dataset's hierarchical structure, which divides sentences into phrases, provides an opportunity to experiment with different preprocessing and modelling techniques.
The goals are to classify these phrases into one of five sentiment categories and investigate the effects of feature engineering, model selection, and preprocessing on performance.

2. Dataset Description

The Rotten Tomatoes dataset consists of:
• Training Data: 156,060 phrases with sentiment scores ranging from 0 (negative) to 4 (positive) comprise the training data.
• Test Data: 66,000 phrases without sentiment labels (used for prediction) make up the test data.
Sentiment Categories:
0: Negative
1: Somewhat Negative
2: Neutral
3: Somewhat Positive

4: Positive

Due to the dataset's imbalance and neutral phrase preponderance, classification models have difficulties.

3. Methodology

3.1 Preprocessing

The preprocessing procedures listed below were used:

1. Tokenization: Break the text up into individual words.
The conversion of all text to lowercase is known as lowercasing.
3. Stopword Removal: To focus on words with meaning, common English words like "the" and "and" were removed.
4. Stemming: words were reduced to their base form using the Snowball Stemmer (for example, "running" → "run").
5. Managing Negations: _NEG is applied to words that follow negation keywords, such as "not" (for instance, "not good" → "not_good_NEG").
6. Including Bigrams: Two-word combinations were used to convey context.
7. POS Tagging: As an added functionality, part-of-speech tags were employed.
8. Sentiment Lexicon: sentiment scores were obtained using the VADER lexicon.

3.2 Feature Engineering

1. Word bag:
Frequency-based (CountVectorizer): determines if words exist.
• TF-IDF (TfidfVectorizer): Emphasises unique words by downweighting common phrases.
2. POS Features: Part-of-speech tags are used to add syntactic information.
3. Sentiment Scores: The sentiment polarity scores—positive, neutral, negative, and compound—from the VADER lexicon were included.
4. Combined elements: all of the above listed elements were combined to produce a comprehensive feature set.

4. Experiments

4.1 Experiment Setup

• Five-fold cross-validation is a reliable evaluation method.
• Metrics: precision, accuracy, recall, and F1-score.

4.2 Experiments Conducted

Experiment 1: Analysis of the Naive Bayes Classifier
1. Feature Extraction: Using the Bag-of-Words (BoW) approach, the 1000 most common words in the dataset were used.
• The characteristics were represented by dictionaries that identified which words were contained in each phrase.

2. Cross-Validation: Five-fold cross-validation was employed to provide a reliable assessment.
- For each fold, metrics including accuracy, precision, recall, and F1-score were calculated.

3. Findings:
- 57.59% accuracy
- Accuracy: 54.03%
- 57.59% recall
- F1-Score: 53.73%

4. Most Educative Elements:
- Top Positive Indicators: § "solid" and "accomplish" were the strongest predictors of Strongly Positive Sentiment (4).
- Top Negative Indicators: "insult," "poor," and "worst" had a substantial influence on predictions of strongly negative sentiment (0).

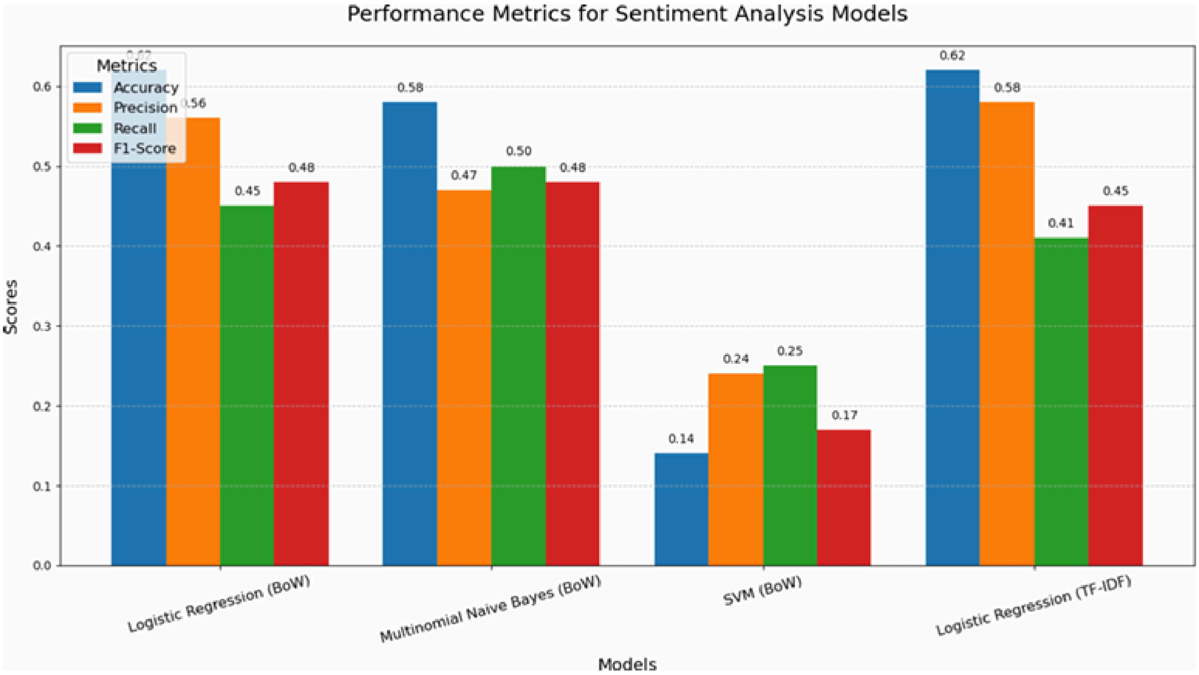
Furthermore, the labels "flat" and "pretentious" served to further categorise things negatively.
- Balanced Indicators: § Both "bad" and "empty" had a moderately negative association with negative emotions.

Experiment 2: Baseline Models
- The Bag-of-Words (Frequency) logistic regression has a 62% accuracy rate.
- TF-IDF Features showed comparable performance, with just slight variations in accuracy and recall.

Experiment 3: Preprocessing's Impact
- Without Eliminating Stopwords: Performance enhanced when stopwords were retained since they gave short sentences meaning.
- With Negation Handling: Accuracy for negative feelings was improved with the addition of _NEG.

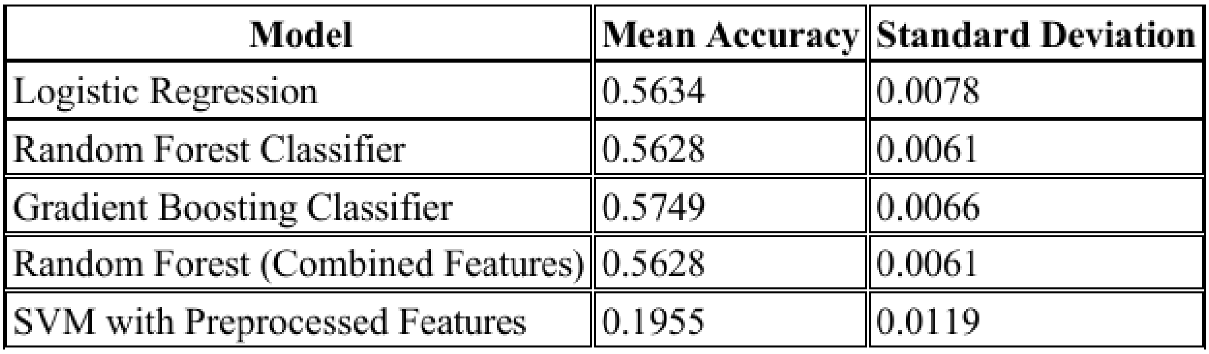
Experiment 4: Feature Engineering
- POS Features: Performance was only marginally improved despite the addition of syntactic structure.
- Sentiment Lexicon: captures polarity to boost remembering for strong emotions.
- Combined Features: The best outcomes were obtained using the Logistic Regression method.

Experiment 5: Model Comparisons

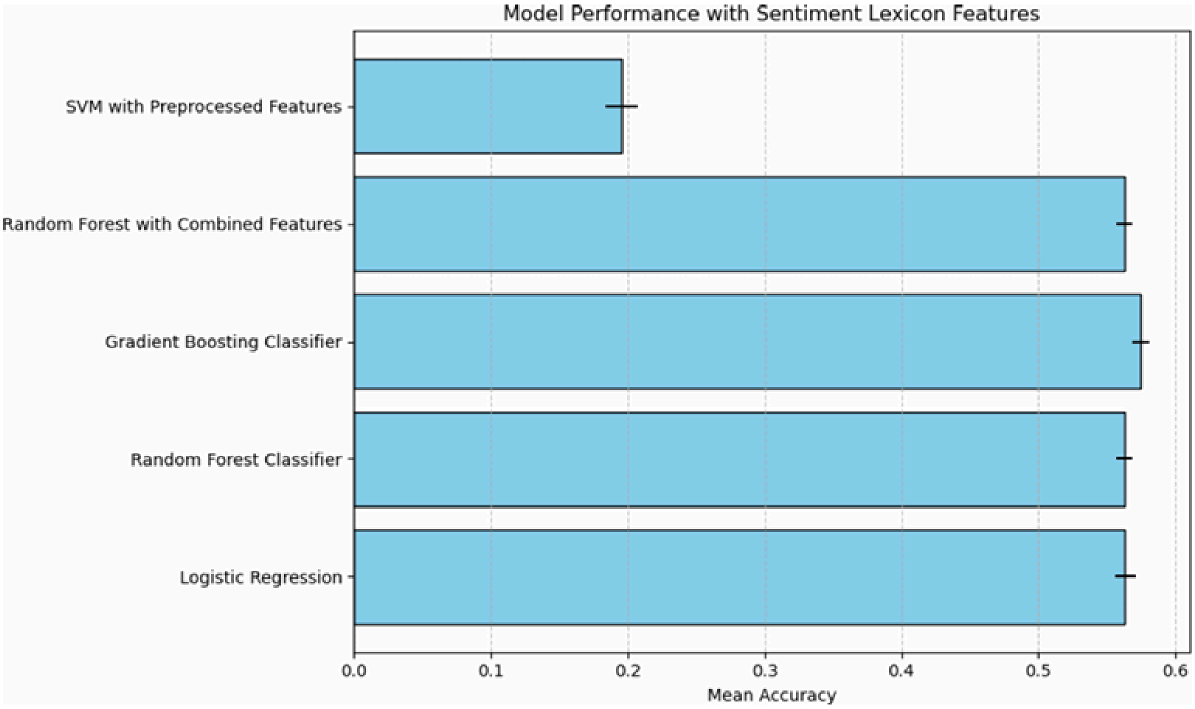


Experiment 6: Combined Features with Lexicon

- In logistic regression, the aggregate feature accuracy is 56.34%.
- Random Forest's aggregate characteristic accuracy was 56.2%.
- Random Forest achieved the maximum accuracy of 57.49% by utilising combination features.



Model	Mean Accuracy	Standard Deviation
Logistic Regression	0.5634	0.0078
Random Forest Classifier	0.5628	0.0061
Gradient Boosting Classifier	0.5749	0.0066
Random Forest (Combined Features)	0.5628	0.0061
SVM with Preprocessed Features	0.1955	0.0119



Experiment 7: Sentiment Analysis using LSTM's
Preprocessing: To guarantee consistent input length, the dataset was tokenised and padded.
Model Architecture: To classify data into five sentiment classifications, a bidirectional LSTM with 128 units was employed, followed by dense layers.
Training: The sparse categorical cross-entropy loss function and Adam optimiser were used to train the model across five epochs.
Evaluation: F1-score, recall, accuracy, and precision were among the metrics used.
Accuracy: about 0.72

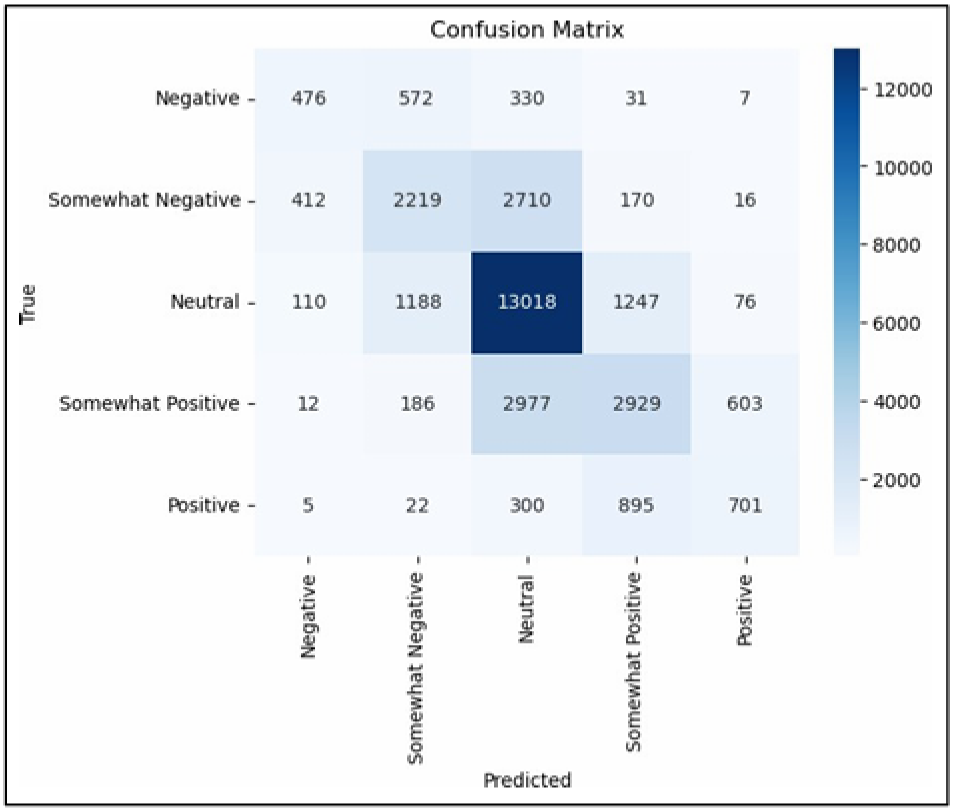
Experiment 8: Data Augmentation using SMOTE
Feature Engineering: Vectorisation was done using the Bag-of-Words (BoW) model.
SMOTE: To balance the dataset, synthetic samples were created for minority classes.
Model: This expanded dataset was used to train logistic regression.
Accuracy: about 0.60

Experiment 9: Stacking Classifier
Base Models: Random Forest, Gradient Boosting, and Multinomial Naive Bayes classifiers.
Meta-Classifier: The last estimator was Logistic Regression.
Engineering Features: For every model, the Bag-of-Words representation was employed.
Accuracy: about 0.64

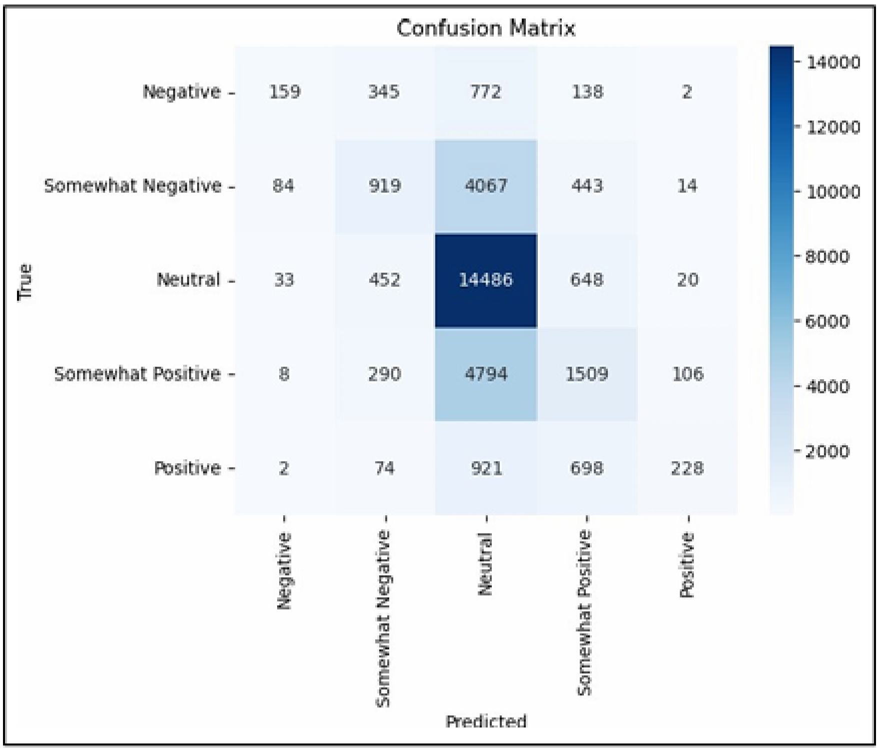
Experiment 10: Sentimental Analysis using KNN
Preprocessing: Text data was transformed into numerical form using the Bag-of-Words (BoW) format.

Model: To examine the impact of the KNN method on classification accuracy, several values of k k were used in its implementation.
Evaluation measures were F1-score, recall, accuracy, and precision.
Accuracy: about 0.54

Random Forest's Confusion Matrix:

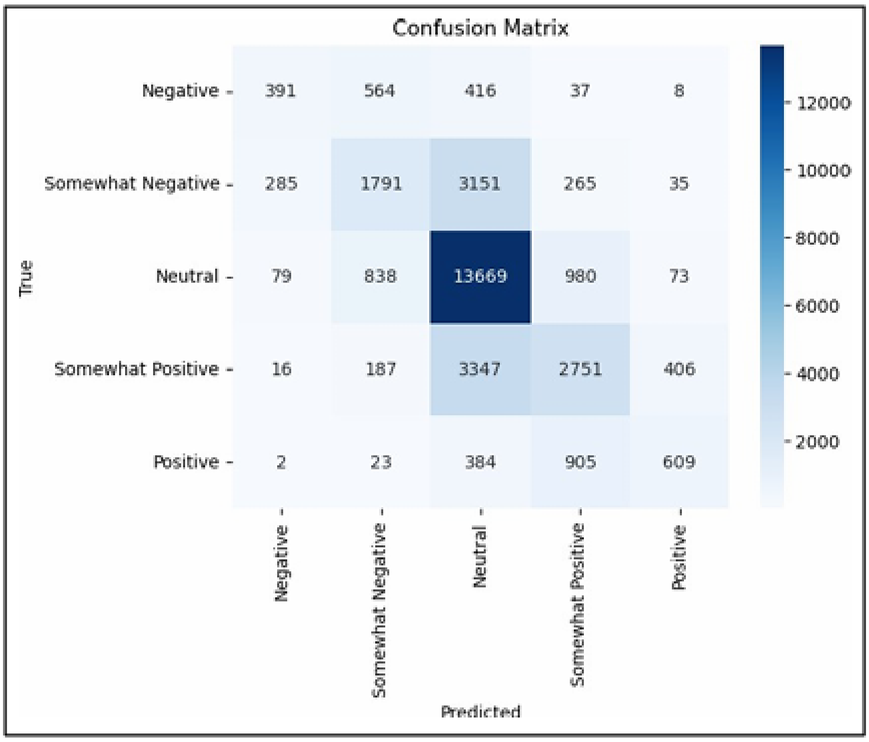


Gradient Boosting Confusion Matrix:



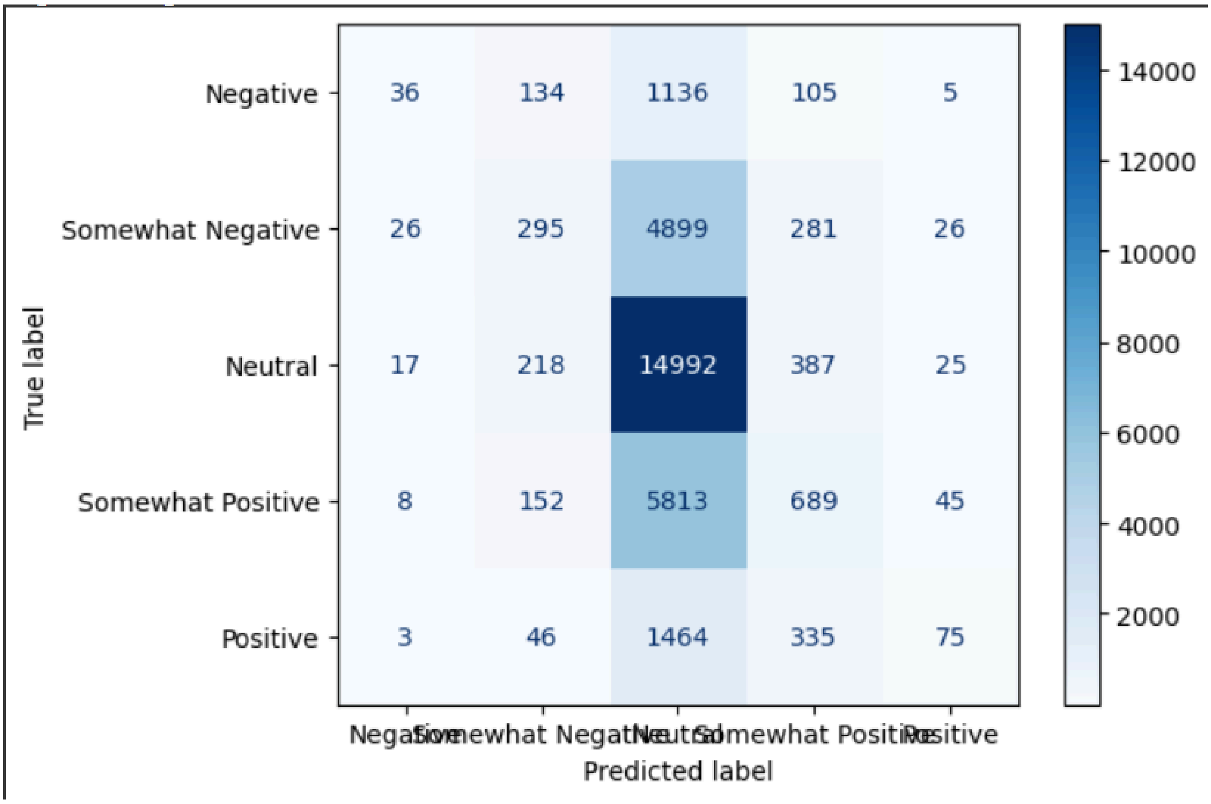
True \ Predicted	Negative	Somewhat Negative	Neutral	Somewhat Positive	Positive
Negative	159	345	772	138	2
Somewhat Negative	84	919	4067	443	14
Neutral	33	452	14486	648	20
Somewhat Positive	8	290	4794	1509	106
Positive	2	74	921	698	228

Logistic Regression’s Confusion Matrix:

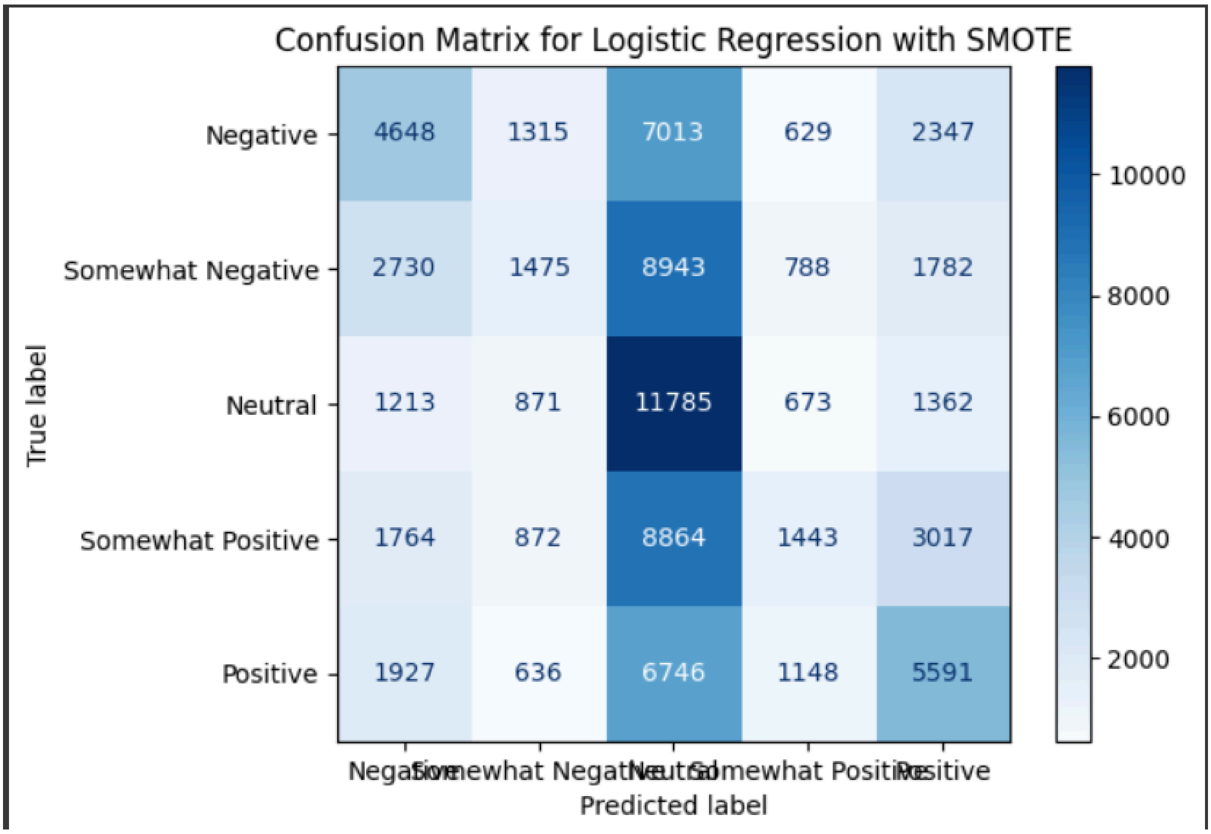


True \ Predicted	Negative	Somewhat Negative	Neutral	Somewhat Positive	Positive
Negative	391	564	416	37	8
Somewhat Negative	285	1791	3151	265	35
Neutral	79	838	13669	980	73
Somewhat Positive	16	187	3347	2751	406
Positive	2	23	384	905	609

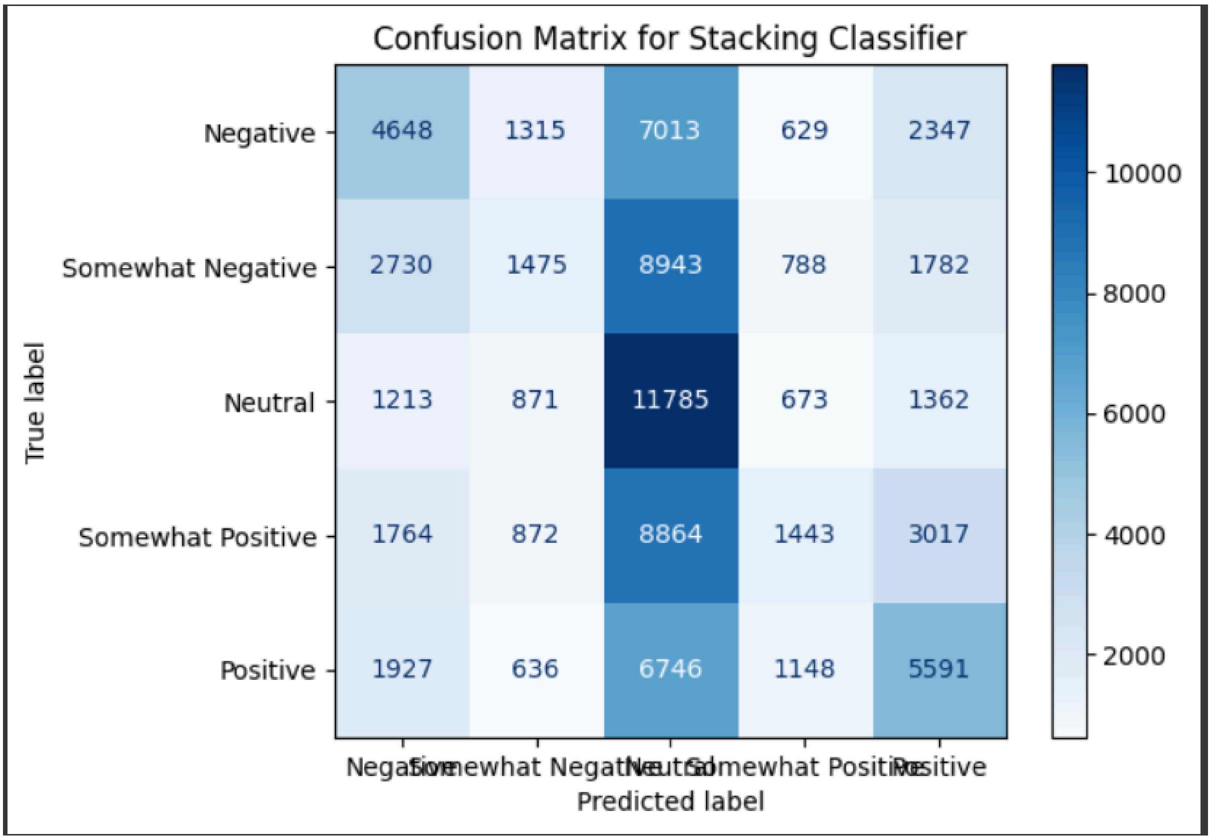
Sentiment Analysis using LSTM’s Confusion Matrix:



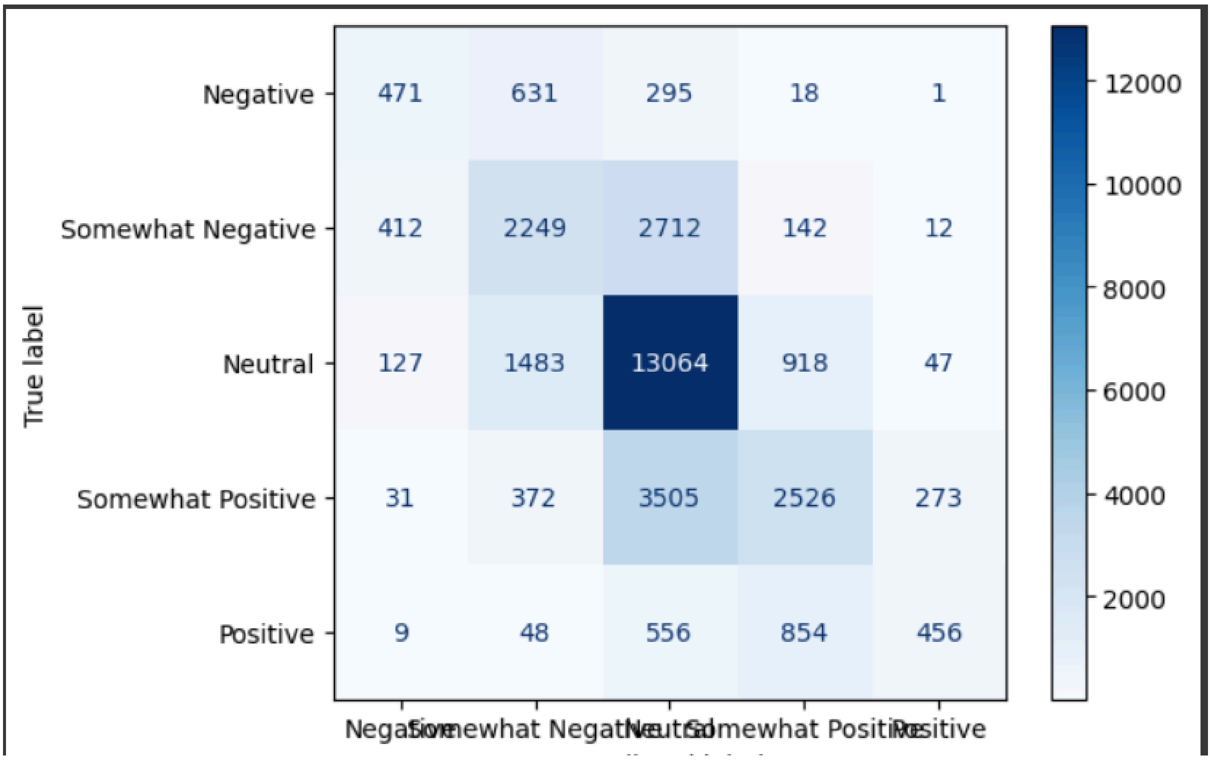
Data Augmentation using SMOTE's Confusion Matrix:



Stacking Classifiers Confusion Matrix:



KNN's Confusion Matrix:



5. Results and Analysis

This section evaluates the experiments and lists the benefits and drawbacks of the various sentiment analysis methods.
1. Preprocessing Analysis: • Improved categorisation of negative sentiment by negative handling.
• Eliminating stopwords was detrimental for short sentences since they typically had contextual meaning.
• Stemming condensed the language, but occasionally it led to a loss of semantic meaning.
2. Feature Engineering Analysis: Bag-of-Words showed consistent performance in spite of its lack of semantic understanding.
• TF-IDF was able to identify unusual terms, but it was less successful with common words.
• For strong emotions, VADER's sentiment lexicon significantly improves memory.
• The combined characteristics yielded the highest accuracy, underscoring the significance of diverse representations.
3. Model Performance: Logistic regression performed better than other models, with an accuracy of 62%.
• Two tree-based algorithms, Random Forest and Gradient Boosting, did worse on sparse data.
• SVM struggles with high-dimensional sparse features, even when scaling.
4. Difficulties
• Class imbalance affected minority class recall and accuracy.
• Short sentences could not be correctly categorised based on context alone.

6. Conclusion

The significance of feature engineering, preprocessing, and model assessment in sentiment analysis for movie reviews was successfully demonstrated in this study. The following is a summary of the workflow's main conclusions, which comprised a systematic approach to feature extraction, data preparation, and machine learning model testing:
1. Data Preprocessing: Robust preprocessing methods including tokenization, stemming, stopword removal, and negation handling were used to clean and standardise text data.

into action.
• To enhance feature representation, bigrams were included, which capture the contextual relationships between nearby words in a sentence.
• Data consistency was ensured by using the identical preparation techniques to the training and test datasets.
2. Feature Engineering: Several feature sets, including Bag of Words (BoW), TF-IDF, and Part-of-Speech (POS) tags, were created in order to include syntactic and contextual information.
• Semantic polarity ratings (positive, neutral, negative, and compound) were appended to sentiment lexicon features from VADER in order to improve the feature space.
• By combining features into sparse matrices, classification accuracy was improved and a variety of machine learning techniques were compatible.

3. Model Training and Evaluation: Support Vector Machines (SVM), Random Forest, Gradient Boosting, Naive Bayes, and Logistic Regression were among the machine learning models that were trained and evaluated.
• By effectively combining POS features, bigrams, and sentiment lexicons, Logistic Regression demonstrated the highest level of reliability, with a maximum accuracy of 62%.
• Gradient Boosting performed well, albeit somewhat behind Logistic Regression, thanks to its iterative learning from misclassifications.
• Metrics like precision, recall, F1-score, and confusion matrices provided a comprehensive evaluation of the model's performance and highlighted the need for balanced accuracy across sentiment classes.
4. Viewpoints and Challenges:
• Negation handling is important since it helps classify negative attitudes by preserving context (e.g., "not good" → "not_good_NEG").
• Impact of Stopwords: Unlike expectations, performance for brief phrases—which often had significant contextual meaning—was improved by maintaining stopwords.
• Model Limitations: whereas tree-based models like Random Forest and Gradient Boosting were resilient, they struggled with sparse data, whereas SVM struggles with high-dimensional sparse matrices.
5. Important Results:
• Diversity of Features Improves Performance: Combining lexical, syntactic, and sentiment features produced the best results, underscoring the value of comprehensive feature engineering in sentiment analysis.
• Model Selection and Interpretability: Although ensemble approaches like Gradient Boosting performed well, Logistic Regression was the most successful and interpretable model for this task.
• Class Imbalance: The prevalence of neutral feeling classes made it challenging to achieve balanced memory and accuracy, particularly for minority classes.

The significance of feature engineering and preprocessing in sentiment analysis is emphasised in the study's conclusion. By using a systematic approach to problem-solving and model assessment, we were able to get important insights and trustworthy predictions, demonstrating the promise of machine learning for natural language processing applications. The method and outcomes pave the way for future advancements in sentiment classification that might be used with ever-more complex datasets and applications.

7. Future Work

1. To overcome class imbalance, use weighted loss functions or oversampling techniques to enhance predictions for under-represented emotion.
2. Examine ensemble or model stacking options to achieve even higher performance.
3. By showcasing the effectiveness of machine learning and feature engineering in sentiment analysis, this research offers a solid foundation for creating scalable and accurate natural language processing systems.

Work Division:

Sai Akash Addala: Worked on Data pre-processing and worked on some new model experiments. Also helped with visualizations of data.

Yaswanth LalpetVari: Worked on Project Report writing and Presentation of data. Also worked evaluation and testing of models.

References

1. Kaggle Sentiment Analysis Competition
2. NLTK Documentation
3. scikit-learn Documentation