Sentiment Analysis on Customer Feedback

Final Report

Group 23

Group Members

21903488 Emrehan Hoşver

21901798 Gökberk Altıparmak

21802480 Tuğberk Dikmen

21901418 Efser Efe Kantik

1. Introduction

In the rising era of e-commerce, understanding customer sentiments has become paramount for enhancing customer experiences and driving business decisions. Sentiment analysis, a domain of natural language processing (NLP), offers a powerful tool to gauge public opinion, emotional nuances, and customer satisfaction by analyzing unstructured text data from customer feedback and reviews.

Our project, focused on sentiment analysis of customer feedback, aims to leverage machine learning techniques to decode the intricate layers of customer sentiment expressed in e-commerce platforms. This report outlines our progress in developing models capable of classifying customer feedback into distinct sentiment categories: positive, negative, and neutral.

2. Background Information

Our project began with an in-depth analysis of datasets from Brazilian e-commerce reviews and the Amazon Musical Instrument Reviews datasets. The first step in our project was data preprocessing, which included cleaning the text, removing irrelevant words, and categorizing the data to make it ready for machine learning algorithms.

We tested several machine learning models to find the best one for sentiment analysis. These are strong in classifying text and work well with large, complex datasets. This model is fast and straightforward, good for large datasets, and we improved it further by balancing the data to better handle different types of reviews. RNNs are great for understanding sequences,

like sentences in reviews, which helps in identifying sentiments over time. We used BERT for its advanced ability to understand the context in sentences, leading to highly accurate sentiment analysis.

By analyzing the performance of these models, we aimed to determine which was most effective at decoding the vast array of customer sentiments expressed in the reviews, with the ultimate goal of gaining a clearer understanding of consumer attitudes and opinions.

3. Dataset split, preprocessing, and detailed analysis.

For both datasets, a typical approach would be to split the data into training (70-80%) and testing (20-30%) sets. The training set is used to build and tune the sentiment analysis model, while the testing set is used to evaluate its performance.

3.1. Amazon

3.1.1. Preprocessing Steps:

3.1.1.1. Text Cleaning: Remove any HTML tags, URLs, and non-alphanumeric characters from reviewText.

3.1.1.2. Handling Missing Values: Check for and handle missing values, especially in reviewText.

3.1.1.3. Feature Engineering: Extract useful features like review length, word count, and sentiment scores from reviewText.

3.1.2. Analysis Approach:

3.1.2.1. Sentiment Analysis: Use the overall rating to classify reviews into positive, negative, and neutral. Perform sentiment analysis on reviewText and compare it with the overall rating.

3.2. Brazilian E-Commerce

3.2.1. Preprocessing Steps:

3.2.1.1. Text Cleaning: Clean the review_comment_message to remove non-alphanumeric characters and other unnecessary text.

3.2.1.2. Handling Missing Values: Handle missing values in review_comment_message.

3.2.2. Analysis Approach:

3.2.2.1. Review Score Analysis: Analyze the distribution of review_score and its relation to the sentiment expressed in review_comment_message.

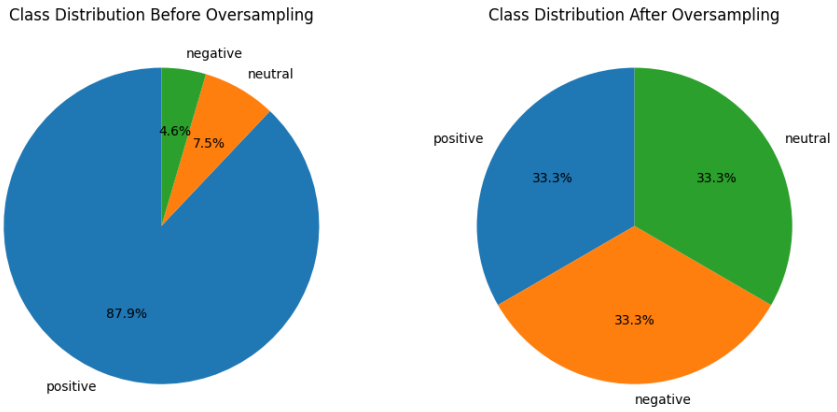


Figure 1: Class Distributions Before & After Oversampling of the Minority Classes

4. Coding environment summary

During the course of the project every group member used the coding environment that fit best to him. Some of us used Google Colab and some of us used the VScode and local computer to run and train the models. The development of our project progressed on Python because of the support for machine learning and natural language processing. The Python ecosystem, with its comprehensive set of libraries and tools, has been critical in the development of our sentiment analysis project. Below, we list the key Python libraries used in our project so far:

4.1. Pandas: A cornerstone in our data handling processes, Pandas provides high-level data structures and functions designed for easy manipulation and analysis of structured data. It has been used primarily in data cleaning, transformation, and the exploration of datasets.

4.2. Scikit-learn: This library forms the backbone of our machine learning operations. Scikit-learn offers a wide array of tools for data preprocessing, model building, and evaluation. We have specifically leveraged its capabilities for model training (using both SVM and Naive Bayes classifiers), data splitting, and feature extraction. The library's user-friendly interface and comprehensive documentation have greatly accelerated our development process.

4.3. NLTK (Natural Language Toolkit): Essential for NLP tasks, NLTK provides a suite of text processing libraries for classification, tokenization, stemming, tagging, and parsing. We have used NLTK for preprocessing the textual data, including removing stopwords, which are words that do not contribute significantly to sentiment analysis.

4.4. Imbalanced-learn: To address the challenge of imbalanced datasets, we have employed Imbalanced-learn, a library offering various resampling techniques. This includes SMOTE, which we have used to synthetically oversample the minority classes in our dataset, ensuring a more balanced and representative training process.

4.5. CountVectorizer from Scikit-learn: An integral part of our text preprocessing pipeline, CountVectorizer converts text data into a matrix of token counts, effectively transforming raw text into a format that can be easily processed by machine learning algorithms.

4.6. TfidfVectorizer from Scikit-learn: Similar to CountVectorizer, TfidfVectorizer was used in our SVM model to convert text documents into a matrix of TF-IDF features. This method emphasizes the importance of frequent terms in a document while accounting for their frequency across the entire dataset, providing a more nuanced feature set for model training.

4.7. Tensorflow: We used Tokenizer from Tensorflow to tokenize our input in RNN and Bert models. We used LSTM, GRU, Dense, Sequential and Dropout from Tensorflow. LSTM, GRU, Dense and Dropout are used as the layers of the RNN model and Sequential used during the initialization of the model in the Python code. We used to_categorical in the phase of defining the labels and lastly we used Adam optimizer as our Optimizer in the RNN.

5. Details of the trained models and configuration

5.1. MNB (Multinomial Naive Bayes)

5.1.1. MNB for Amazon Dataset

Data Preprocessing:

- Dataset Overview: The Amazon dataset consists of musical instrument reviews. Each entry includes a text review and a numerical rating.

- Data Loading: Utilized: Python's JSON module to load the data, ensuring proper handling of any encoding issues.
- Cleaning and Preprocessing: Implemented text preprocessing “stop word removal” technique to remove irrelevant characters and standardize the text data.
- Sentiment Classification: Ratings were categorized into three sentiment classes: 'negative' for ratings 1 and 2, 'neutral' for rating 3, and 'positive' for ratings 4 and 5.
- Feature Extraction: The `CountVectorizer` was employed to convert text reviews into a sparse matrix of token counts, excluding common English stop words.

Model Training:

- Data Splitting: Applied a 60-20-20 split for training, validation, and testing sets to ensure a balanced representation of classes.

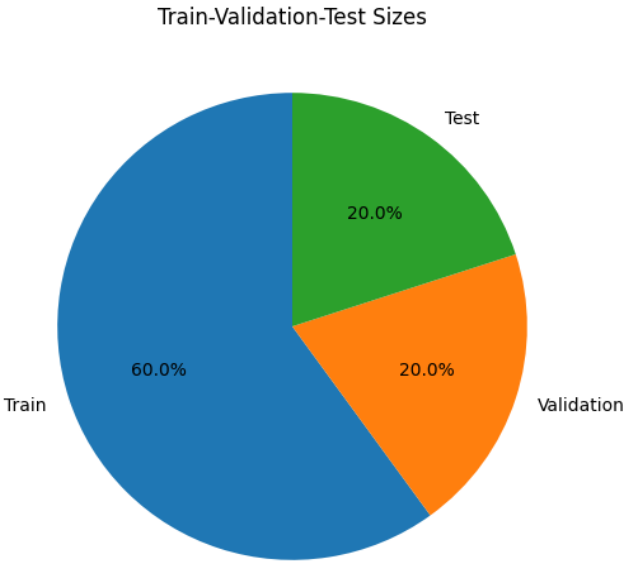


Figure 2: The train-validation-test dataset splitting

- Fitting the Model: Trained the model on the training set and performed hyperparameter tuning on the validation set to optimize performance.

Evaluation and Metrics:

- Accuracy Assessment: Evaluated the model's accuracy on the unseen test data to gauge its generalization capabilities.

- Detailed Metrics: Computed precision, recall, and F1-score for a more nuanced understanding of model performance across different classes.
- Confusion Matrix Visualization: Illustrated the model's performance in correctly predicting each class, highlighting any biases or tendencies towards specific sentiments.

5.1.2. MNB for Brazilian E-commerce Dataset

Data Preprocessing:

- Dataset Characteristics: The dataset comprises reviews from a diverse range of products in Brazilian e-commerce.
- Data Importing: Leveraged pandas to import and preprocess the CSV file, ensuring consistency in data formatting.
- Text Normalization: Undertook extensive text preprocessing, including language-specific considerations due to the dataset's Portuguese content.
- Sentiment Mapping: Similar sentiment classification was applied, utilizing the same logic as with the Amazon dataset.
- Vectorization Approach: Employed the same `CountVectorizer` technique, adapted to the linguistic characteristics of the Portuguese language.

Model Training and Evaluation:

- Training Process: The model training followed a similar approach to the Amazon dataset, with adjustments for the unique aspects of the Brazilian e-commerce dataset.
- Performance Metrics: Focused on accuracy, precision, recall, and F1-score, emphasizing the model's ability to handle diverse product reviews.

5.1.3. Comparative Analysis

- Dataset Differences: Discussed the variations in the datasets, including the linguistic and cultural differences impacting sentiment analysis.
- Model Performance Comparison: Detailed comparison of the model’s performance on both datasets, highlighting strengths and weaknesses in different e-commerce contexts.

5.2. SVM

5.2.1. Data Loading and Preprocessing:

- Importing necessary libraries and modules.
- Loading a CSV file containing customer feedback.
- Extracting the relevant columns (`review_score` and `review_comment_message`).
- Handling missing values by dropping rows with NaNs.
- Encoding the `review_score` into labels ('negative', 'neutral', 'positive').

5.2.2. Data Splitting:

- Splitting the dataset into training, validation, and test sets using `train_test_split`.

5.2.3. Feature Extraction:

-Extracting features from the text data (`review_comment_message`) using `TfidfVectorizer`.

5.2.4. SVM Model Training and Evaluation:

- Constructing an SVM model pipeline with `TfidfVectorizer` and `SVC`.
- Training the model on the training set and evaluating it on the validation and test sets.
- Printing classification reports for performance evaluation.

5.2.5. Data Augmentation and Re-evaluation:

- Augmenting the dataset by duplicating 'neutral' samples.
- Re-splitting the augmented dataset into training, validation, and test sets.
- Re-training and evaluating the SVM model on the new dataset splits.

5.2.6. SVM Configuration

-Support Vector Machine (SVM): Used here is `SVC` (Support Vector Classifier), a type of SVM used for classification tasks.

- TfidfVectorizer: This is used for converting the text data into a matrix of TF-IDF features, which is a numerical representation suitable for machine learning models.

- C Parameter: In the first model training, the `C` parameter of SVC is set to 1. This parameter controls the trade-off between having a smooth decision boundary and classifying training points correctly. A smaller `C` value leads to a smoother decision boundary (more regularization), while a larger `C` allows more flexibility to the model (less regularization).

- Default SVC in Second Training: In the second training, the SVC is used with its default parameters. The `C` parameter, if not specified, typically defaults to 1.0.

5.3. RNN

RNN (Recurrent Neural Network) is a neural network type that uses the time series.

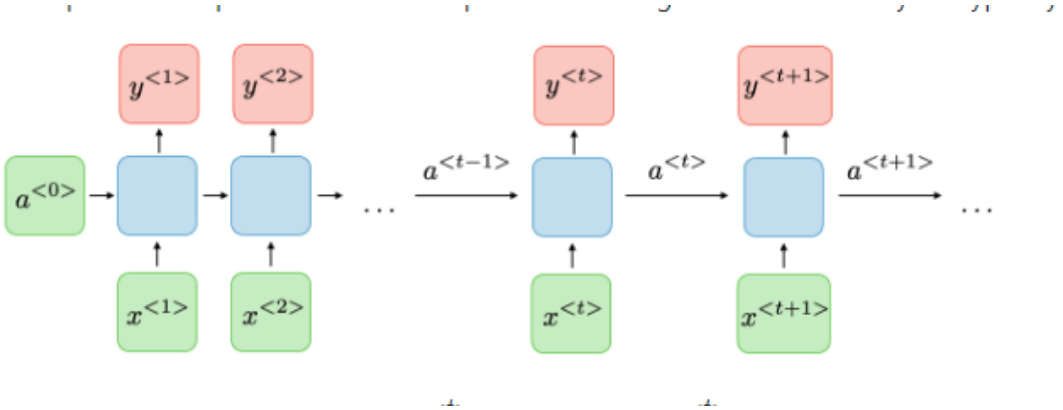


Figure 3:Architecture of Traditional RNN

Initially, we implemented the RNN with Simple RNN's with a Dense layer. Even though the model performed well in terms of computational complexity the accuracy was below our needs and goals. So designed an RNN which is experimental and includes LSTM and GRU as its inner layers. LSTM and GRU use tanh as their activation function internally.

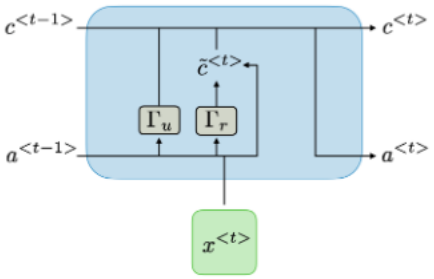
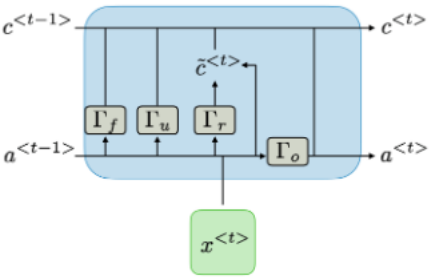
Characterization	Gated Recurrent Unit (GRU)	Long Short-Term Memory (LSTM)
$\tilde{c}^{<t>}$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$
$c^{<t>}$	$\Gamma_u \star \tilde{c}^{<t>} + (1 - \Gamma_u) \star c^{<t-1>}$	$\Gamma_u \star \tilde{c}^{<t>} + \Gamma_f \star c^{<t-1>}$
$a^{<t>}$	$c^{<t>}$	$\Gamma_o \star c^{<t>}$
Dependencies		

Figure 4. Architectures of GRU and LSTM

There are two dropout layers to eliminate possible overfit and there are two dense layers at the end of the RNN structure. Activation functions of the dense layers are Relu and Softmax. Relu is used at the inner dense layer and the softmax is used at the dense layer which is the final layer of the architecture.

5.4. BERT

The BERT model, standing for Bidirectional Encoder Representations from Transformers, is a state-of-the-art Natural Language Processing (NLP) algorithm. Unlike traditional methods that read input text sequentially, BERT applies bidirectional processing, considering both the left and right context in all layers. This enables the model to capture intricate relationships and dependencies among words, leading to a comprehensive understanding of context. This model excels in sentiment analysis by grasping nuances and context for more informed predictions.

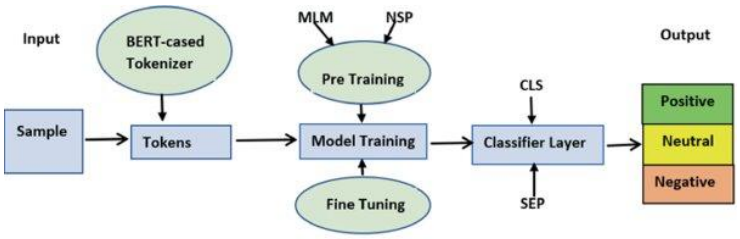


Figure 5: Architecture of the BERT algorithm of sentiment analysis with 3 sentiment categories

5.4.1 Dataset Preprocessing & Selection

The custom-built sentiment rating function is strategically applied to map the 'overall' scores to sentiment categories (negative, neutral, positive). Subsequently, the dataset is partitioned into training, validation, and test sets, adhering to predefined proportions.

Recognizing the persistent challenge of class imbalance encountered in earlier algorithms, our approach for the BERT model involves the direct utilization of the upsampled Amazon dataset. This deliberate choice is driven by the aim to augment model performance while concurrently mitigating computational costs. This approach not only reinforces the model's robustness but also accentuates a pragmatic balance between computational resources and performance optimization.

5.4.2 Model Configuration

The “TFBertForSequenceClassification” model provided by the Hugging Face Transformers library is employed. This model, pre-trained on the "bert-base-uncased" architecture, brings a robust foundation for sentiment classification. The sentiment

labels are the target variables, aligning seamlessly with the requirements of this sentiment analysis task.

Text data tokenization is executed using the "BertTokenizer," ensuring a consistent and efficient process. The maximum sequence length is set to 256 tokens, providing a balance between context preservation and computational efficiency. The tokenized data is then seamlessly converted into TensorFlow tensors, ready for integration into the BERT model.

For optimization, we adopt the AdamW optimizer, a proven choice previously employed in the RNN model. The learning rate is set at 2e-5 to facilitate stable and effective model training. The model compilation is fine-tuned with the Categorical Cross Entropy loss function, aligning with the multi-class nature of our sentiment classification task. Additionally, potential memory limitations are circumvented, by applying a strategic gradient accumulation approach. With an effective batch size of 32 and accumulation steps thoughtfully set, we optimize both training performance and memory utilization.

This BERT model undergoes training over three epochs, leveraging the upsampled Amazon training dataset. Each epoch allows the model to iteratively refine its understanding of the sentiment patterns within the data.

6. Using appropriate evaluation metrics

6.1. Overview of Applied Models

Multinomial Naive Bayes (MNB): Utilized for its simplicity and effectiveness in handling discrete data like word counts in text classification.

Bidirectional Encoder Representations from Transformers (BERT): A state-of-the-art model known for its deep understanding of language context and nuances.

Recurrent Neural Network (RNN): Employed for its ability to process sequences of text, capturing temporal dynamics in language.

Support Vector Machine (SVM): Chosen for its robustness in high-dimensional spaces, often used in text classification tasks.

6.2. Evaluation Metrics Overview

- Accuracy: Measures the overall correctness of the model but can be misleading in the presence of class imbalance.
- Precision, Recall, and F1-Score: These metrics provide a more nuanced understanding of the model's performance, especially in imbalanced datasets. Precision measures the accuracy of positive predictions, recall measures the model's ability to detect positive instances, and the F1-score provides a balance between precision and recall.
- Confusion Matrix: Offers a detailed breakdown of the model's performance across different classes.

7. Applying evaluations on the correct data splits

Each model, namely Multinomial Naive Bayes (MNB), Support Vector Machine (SVM), Recurrent Neural Network (RNN), and Bidirectional Encoder Representations from Transformers (BERT), underwent rigorous evaluation on their respective testing sets, which were different from the training and validation sets used during the model development phase.

7.1. Model-Specific Metric Analysis

7.1.1 MNB Evaluation

- Metrics Used: Accuracy, precision, recall, F1-score, and confusion matrix.
- Dataset-Specific Observations: On both datasets, MNB showed a tendency to perform better in the majority class but struggled with minority classes, especially before applying balancing techniques.

Its results provide insights into how well the MNB technique generalizes to unseen data and performs across different sentiment classes, ensuring an accurate representation of its capabilities.

7.1.2. SVM Evaluation

- Metrics Used: Focused on precision and F1-score, considering SVM's effectiveness in high-dimensional feature spaces.

- Dataset-Specific Observations: SVM models were effective in distinguishing between classes, showing robustness in handling diverse text data.

SVM models, known for their robustness in high-dimensional spaces, are evaluated on the designated test set. Evaluation of correct data splitting provides a reliable measure of SVM performance.

7.1.3. RNN Evaluation

- Metrics Used: Emphasis on precision, recall, and F1-score, given RNN's temporal data processing capabilities.
- Dataset-Specific Observations: RNN models showed balanced performance across different classes, particularly effective in capturing sequential patterns in text.

RNN models, designed to capture temporal dynamics in language, are rigorously evaluated on the appropriate test set. The evaluation process ensures that the classification prowess of RNN models is accurately identified.

7.1.4. BERT Evaluation

- Metrics Used: The same set of metrics, with an added focus on recall and F1-score due to BERT's deeper contextual understanding.
- Dataset-Specific Observations: BERT's performance was notable for its high recall, indicating its effectiveness in capturing the context and nuances in text data.

BERT underwent a diligent evaluation on the specifically allocated test set, demonstrating its adeptness in sentiment analysis when presented with previously unseen instances

7.2. Comparative Analysis

- Performance Across Models: A comparative analysis highlighting each model's strengths and weaknesses in terms of the chosen evaluation metrics.

- Dataset Influences: Discussion on how the nature of each dataset (Amazon and Brazilian e-commerce) influenced the model performances.

This insightful comparison is conducted on the dedicated test sets of each model, ensuring that the observed variations accurately mirror their capabilities in sentiment classification. The analysis goes beyond accuracy, delving into precision, recall, and F1-score to provide a nuanced understanding of each model's performance intricacies. The evaluation phase is conducted carefully on the correct data splits, thereby guaranteeing that the reported metrics accurately represent the models' proficiency in sentiment analysis on previously unseen instances. This approach establishes a resilient and impartial evaluation of each model's capabilities, facilitating informed decisions and insights for future improvements.

8. Comparing and discussing the evaluation results of models

8.1. MNB (Multinomial Naive Bayes)

8.1.1. Amazon Dataset Results

Validation Set Before Oversampling:

- Validation Set Accuracy: 86.11%
    - Negative Class: Precision - 40%, Recall - 2%, F1-Score - 4%
    - Neutral Class: Precision - 8%, Recall - 1%, F1-Score - 1%
    - *Positive Class: Precision - 87%, Recall - 99%, F1-Score - 93%

Validation Set After Oversampling:

- Validation Set Accuracy: 87.90%
    - Negative Class: Precision - 91%, Recall - 92%, F1-Score - 92%
    - Neutral Class: Precision - 83%, Recall - 89%, F1-Score - 86%
    - Positive Class: Precision - 87%, Recall - 88%, F1-Score - 88%

Test Set Before Oversampling:

- Test Set Accuracy: 88.31%
    - Negative Class: Precision - 71%, Recall - 6%, F1-Score - 11%
    - Neutral Class: Precision - 12%, Recall - 1%, F1-Score - 2%

- Positive Class: Precision - 89%, Recall - 99%, F1-Score - 94%

Test Set After Oversampling:

- Test Set Accuracy: 87.90%
    - Negative Class: Precision - 91%, Recall - 92%, F1-Score - 92%
    - Neutral Class: Precision - 83%, Recall - 89%, F1-Score - 86%
    - Positive Class: Precision - 87%, Recall - 88%, F1-Score - 88%

8.1.2. Brazilian E-commerce Dataset Results

Validation Set Before Oversampling:

- Validation Set Accuracy: 83.00%
    - Negative Class: Precision - 73%, Recall - 86%, F1-Score - 79%
    - Neutral Class: Precision - 30%, Recall - 11%, F1-Score - 16%
    - Positive Class: Precision - 90%, Recall - 92%, F1-Score - 91%

Validation Set After Oversampling:

- Validation Set Accuracy: 75.41%
    - Negative Class: Precision - 78%, Recall - 80%, F1-Score - 79%
    - Neutral Class: Precision - 72%, Recall - 60%, F1-Score - 65%
    - Positive Class: Precision - 76%, Recall - 87%, F1-Score - 81%

Test Set Before Oversampling:

- Test Set Accuracy: 82.70%
    - Negative Class: Precision - 73%, Recall - 86%, F1-Score - 79%
    - Neutral Class: Precision - 30%, Recall - 11%, F1-Score - 16%
    - Positive Class: Precision - 90%, Recall - 92%, F1-Score - 91%

Test Set After Oversampling:

- Test Set Accuracy: 75.28%
    - Negative Class: Precision - 80%, Recall - 80%, F1-Score - 80%
    - Neutral Class: Precision - 72%, Recall - 60%, F1-Score - 66%
    - Positive Class: Precision - 74%, Recall - 86%, F1-Score - 80%

8.1.3. Key Comparisons

8.1.3.1. Initial Class Imbalance:

Amazon Dataset:

- Initially, the dataset was heavily skewed towards positive reviews, with a significantly lower representation of negative and neutral reviews. This imbalance could lead to a model bias, favoring the majority class (positive).

Brazilian E-commerce Dataset:

- Similar to the Amazon dataset, there was a pronounced class imbalance, with positive reviews dominating the dataset. Such disparities in class distribution often result in poor generalization performance, especially for minority classes.

8.1.3.2. Model Performance Pre-Adjustment:

Amazon Dataset:

- Prior to any adjustments, the model exhibited high precision and recall for the 'positive' class but performed poorly for 'negative' and 'neutral' classes. This was reflected in low F1-scores for these classes, indicating a model bias.

Brazilian E-commerce Dataset:

- The initial model also showed a similar tendency, with disproportionate precision and recall favoring the 'positive' class. The minority classes were not adequately captured, as evidenced by their lower performance metrics.

8.1.3.3. Impact of Balancing Techniques:

Amazon Dataset:

- Oversampling the minority classes ('negative' and 'neutral') resulted in a more balanced representation of all classes. This led to a notable improvement in the model's ability to correctly identify and classify these previously underrepresented classes.

Brazilian E-commerce Dataset:

- Applying similar balancing techniques helped in mitigating the initial bias. The model began to better recognize 'negative' and 'neutral' reviews, improving the equity of the predictive performance across classes.

8.1.3.4. Overall Accuracy:

Amazon Dataset:

- There was a slight decrease in overall accuracy after oversampling. However, this trade-off was beneficial as it led to a more balanced and fair classification system.

Brazilian E-commerce Dataset:

- Similar to the Amazon dataset, a slight dip in overall accuracy was observed post-oversampling. This decrease is often acceptable in the pursuit of a model that is more robust and less biased towards the majority class.

8.1.3.5. Precision, Recall, and F1-Score:

Amazon Dataset:

- After balancing, precision and recall for 'negative' and 'neutral' classes saw a significant increase, resulting in higher F1-scores. This indicates a more reliable and accurate model across all classes.

- The precision for 'positive' reviews slightly decreased, but this was offset by the gains in precision and recall for the other classes.

Brazilian E-commerce Dataset:

- The balancing approach also improved precision, recall, and F1-scores for the minority classes in this dataset. This improvement signifies that the model became more adept at correctly identifying all sentiment classes, not just the predominant one.

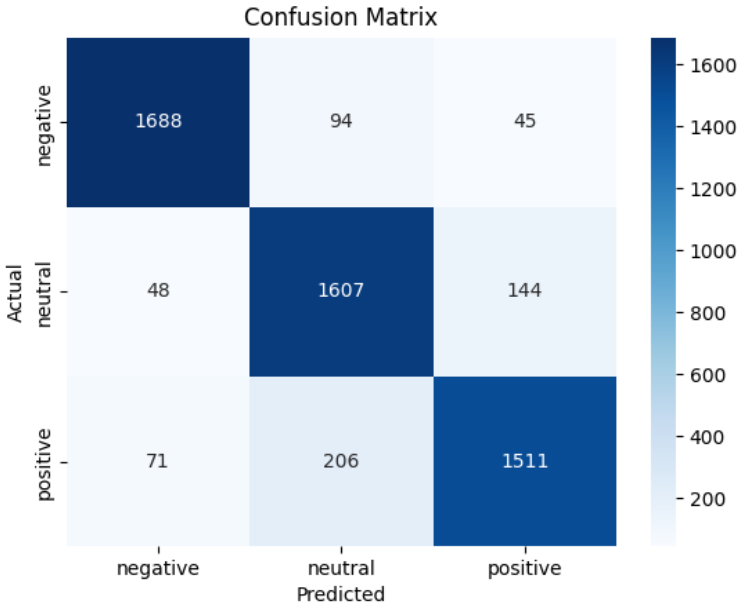


Figure 6: Confusion Matrix for MNB Amazon Dataset

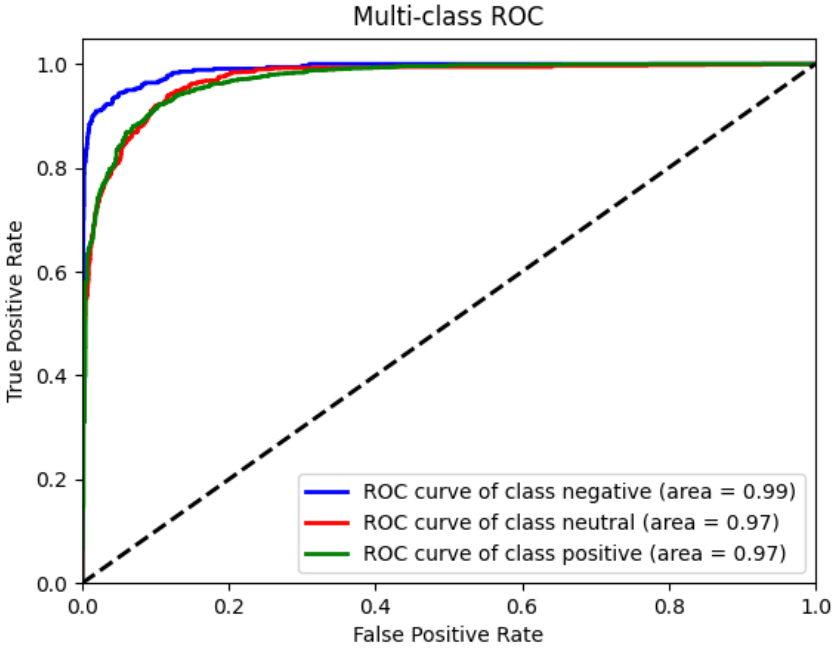


Figure 7: ROC Curve for MNB Amazon Dataset

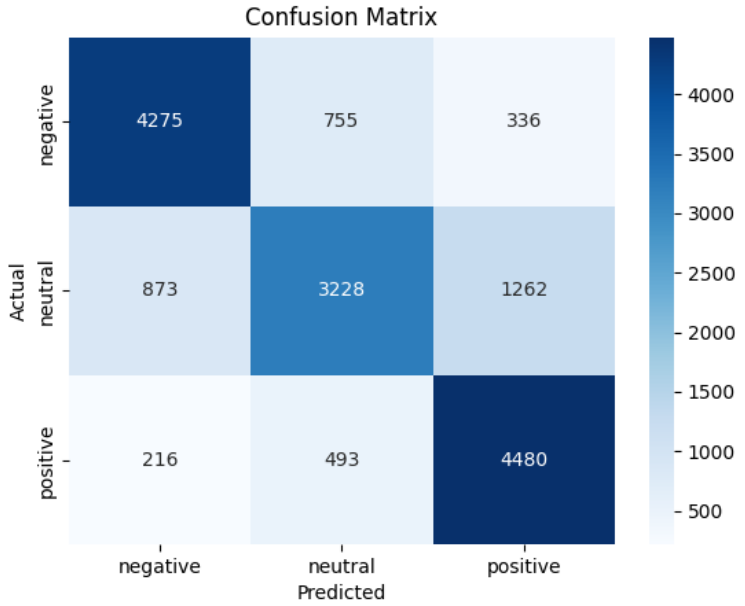


Figure 8: Confusion Matrix for MNB Brazilian E Commerce Dataset

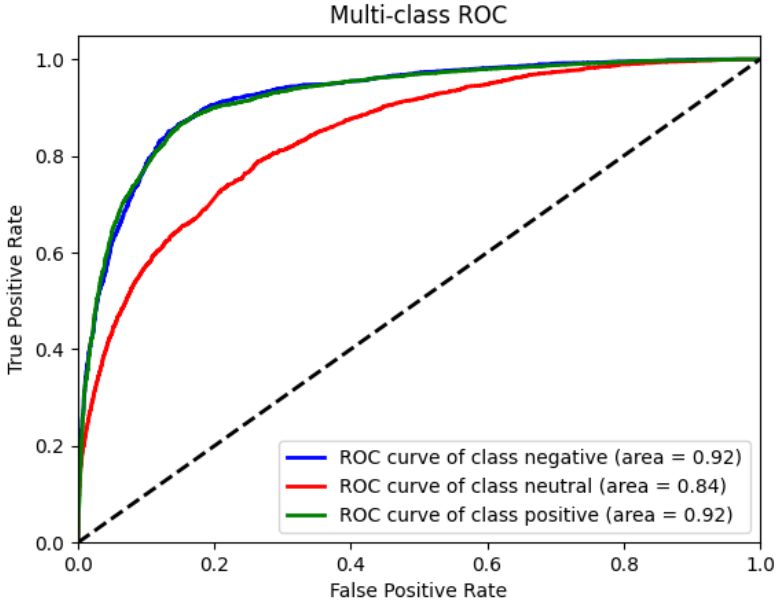


Figure 9: ROC Curve for MNB Brazilian E Commerce Dataset

8.2. SVM

8.2.1. Amazon Dataset

Before Upsampling

- Label Distribution: Positive (9022), Neutral (772), Negative (467)

- Validation Set:

- Accuracy: 82.8%
- Negative: Precision (0.35), Recall (0.29), F1-Score (0.32)
- Neutral: Precision (0.19), Recall (0.14), F1-Score (0.16)
- Positive: Precision (0.90), Recall (0.93), F1-Score (0.91)

- Test Set:
- Accuracy: 83.3%
- Negative: Precision (0.29), Recall (0.35), F1-Score (0.32)
- Neutral: Precision (0.18), Recall (0.17), F1-Score (0.17)
- Positive: Precision (0.92), Recall (0.91), F1-Score (0.91)

After Upsampling

- Label Distribution: Positive (9022), Neutral (9022), Negative (9022)
  - Validation Set:
    - Accuracy: 87.9%
    - Negative: Precision (0.91), Recall (0.92), F1-Score (0.92)
    - Neutral: Precision (0.83), Recall (0.89), F1-Score (0.86)
    - Positive: Precision (0.90), Recall (0.83), F1-Score (0.86)
  - Test Set:
    - Accuracy: 88.8%
    - Negative: Precision (0.93), Recall (0.92), F1-Score (0.93)
    - Neutral: Precision (0.84), Recall (0.89), F1-Score (0.87)
    - Positive: Precision (0.89), Recall (0.85), F1-Score (0.87)

8.2.2. Brazil Dataset

Before Duplication

- Label Distribution: Positive (26530), Neutral (3557), Negative (10890)
  - Validation Set:
    - Accuracy: 84%
    - Negative: Precision (0.74), Recall (0.89), F1-Score (0.81)
    - Neutral: Precision (0.32), Recall (0.01), F1-Score (0.03)
    - Positive: Precision (0.89), Recall (0.94), F1-Score (0.92)
  - Test Set:
    - Accuracy: 85%
    - Negative: Precision (0.77), Recall (0.90), F1-Score (0.83)

- Neutral: Precision (0.41), Recall (0.02), F1-Score (0.04)
- Positive: Precision (0.89), Recall (0.95), F1-Score (0.92)

After Duplication

- Label Distribution: Positive (26530), Neutral (10671), Negative (10890)
  - Validation Set:
    - Accuracy: 85%
    - Negative: Precision (0.78), Recall (0.85), F1-Score (0.82)
    - Neutral: Precision (0.81), Recall (0.68), F1-Score (0.74)
    - Positive: Precision (0.89), Recall (0.92), F1-Score (0.90)
  - Test Set:
    - Accuracy: 85%
    - Negative: Precision (0.81), Recall (0.83), F1-Score (0.82)
    - Neutral: Precision (0.79), Recall (0.68), F1-Score (0.73)
    - Positive: Precision (0.89), Recall (0.93), F1-Score (0.91)

8.2.3. Key Comparisons

8.2.3.1. Initial Class Imbalance:

- Amazon: More skewed with a very small 'negative' class.
- Brazil: Better initial balance but still skewed towards 'positive'.

8.2.3.2. Model Performance Pre-Adjustment:

- Amazon: Lower performance in 'negative' and 'neutral' categories.
- Brazil: Better performance in 'negative', poor in 'neutral'.

8.2.3.3. Impact of Balancing Techniques:

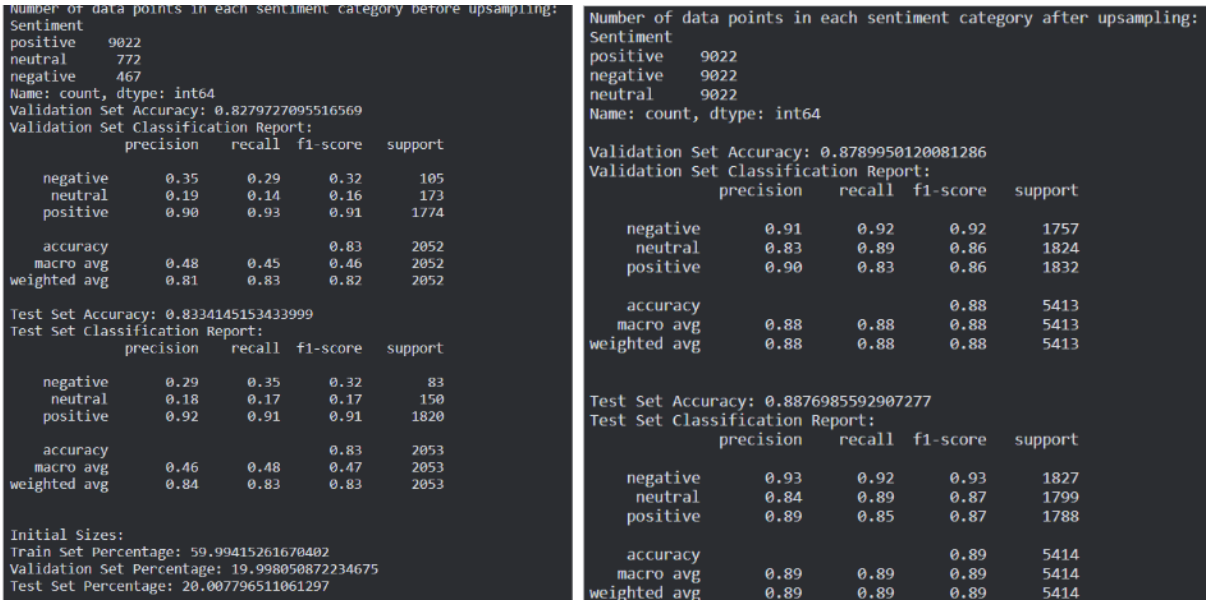
- Amazon: Significant improvement in 'negative' and 'neutral' after upsampling.
- Brazil: Improvement in 'neutral' after duplication, good preservation of 'negative' and 'positive' performance.

8.2.3.4. Overall Accuracy:

- Both datasets show improvement in overall accuracy after class balance adjustments, but the improvement is more pronounced in the Amazon dataset.

8.2.3.5. Precision, Recall, and F1-Score:

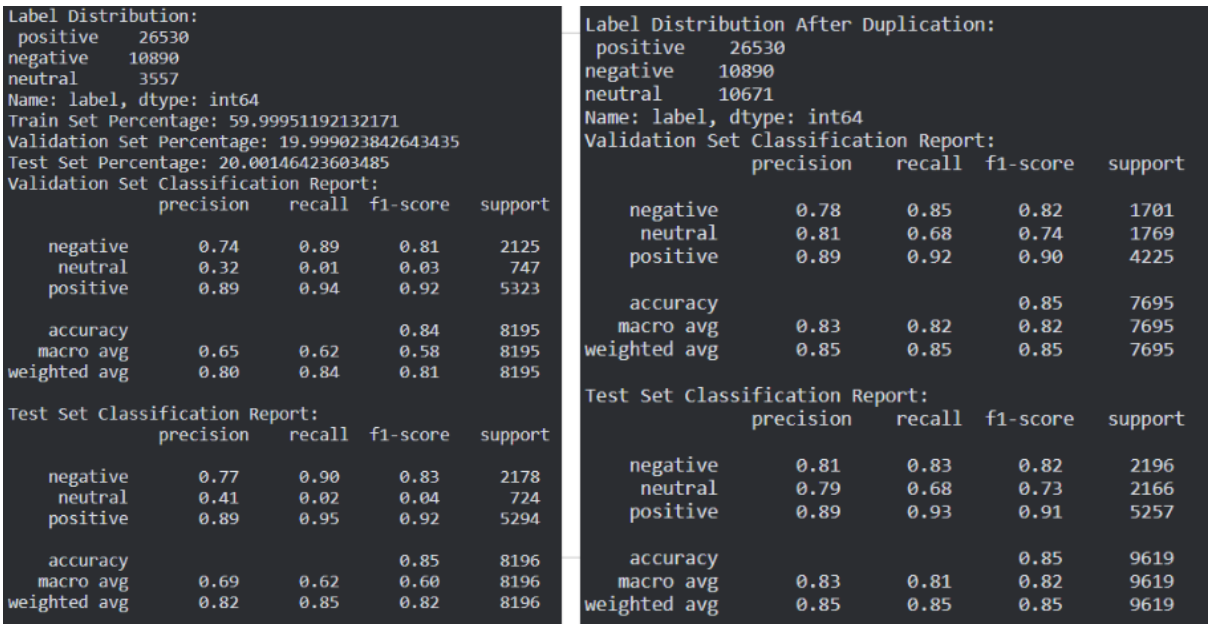
- Amazon: Large gains in precision, recall, and F1-score for minority classes post-adjustment.
- Brazil: Consistent performance across classes, with notable improvements in the 'neutral' class.



```
Number of data points in each sentiment category before upsampling:
Sentiment
positive    9022
neutral      772
negative     467
Name: count, dtype: int64
Validation Set Accuracy: 0.8279727095516569
Validation Set Classification Report:
              precision    recall  f1-score   support

    negative       0.35      0.29      0.32       105
     neutral       0.19      0.14      0.16       173
    positive       0.90      0.93      0.91      1774

    accuracy                           0.83      2052
   macro avg       0.48      0.45      0.46      2052
weighted avg       0.81      0.83      0.82      2052

Test Set Accuracy: 0.8334145153433999
Test Set Classification Report:
              precision    recall  f1-score   support

    negative       0.29      0.35      0.32        83
     neutral       0.18      0.17      0.17       150
    positive       0.92      0.91      0.91      1820

    accuracy                           0.83      2053
   macro avg       0.46      0.48      0.47      2053
weighted avg       0.84      0.83      0.83      2053


Initial Sizes:
Train Set Percentage: 59.99415261670402
Validation Set Percentage: 19.998050872234675
Test Set Percentage: 20.007796511061297
```

```
Number of data points in each sentiment category after upsampling:
Sentiment
positive    9022
negative    9022
neutral     9022
Name: count, dtype: int64

Validation Set Accuracy: 0.8789950120081286
Validation Set Classification Report:
              precision    recall  f1-score   support

    negative       0.91      0.92      0.92      1757
     neutral       0.83      0.89      0.86      1824
    positive       0.90      0.83      0.86      1832

    accuracy                           0.88      5413
   macro avg       0.88      0.88      0.88      5413
weighted avg       0.88      0.88      0.88      5413


Test Set Accuracy: 0.8876985592907277
Test Set Classification Report:
              precision    recall  f1-score   support

    negative       0.93      0.92      0.93      1827
     neutral       0.84      0.89      0.87      1799
    positive       0.89      0.85      0.87      1788

    accuracy                           0.89      5414
   macro avg       0.89      0.89      0.89      5414
weighted avg       0.89      0.89      0.89      5414
```

Figure 10: Test Results of Amazon Dataset



```
Label Distribution:
 positive    26530
negative     10890
neutral       3557
Name: label, dtype: int64
Train Set Percentage: 59.99951192132171
Validation Set Percentage: 19.999023842643435
Test Set Percentage: 20.00146423603485
Validation Set Classification Report:
              precision    recall  f1-score   support

    negative       0.74      0.89      0.81      2125
     neutral       0.32      0.01      0.03       747
    positive       0.89      0.94      0.92      5323

    accuracy                           0.84      8195
   macro avg       0.65      0.62      0.58      8195
weighted avg       0.80      0.84      0.81      8195

Test Set Classification Report:
              precision    recall  f1-score   support

    negative       0.77      0.90      0.83      2178
     neutral       0.41      0.02      0.04       724
    positive       0.89      0.95      0.92      5294

    accuracy                           0.85      8196
   macro avg       0.69      0.62      0.60      8196
weighted avg       0.82      0.85      0.82      8196
```

```
Label Distribution After Duplication:
 positive    26530
negative     10890
neutral      10671
Name: label, dtype: int64
Validation Set Classification Report:
              precision    recall  f1-score   support

    negative       0.78      0.85      0.82      1701
     neutral       0.81      0.68      0.74      1769
    positive       0.89      0.92      0.90      4225

    accuracy                           0.85      7695
   macro avg       0.83      0.82      0.82      7695
weighted avg       0.85      0.85      0.85      7695

Test Set Classification Report:
              precision    recall  f1-score   support

    negative       0.81      0.83      0.82      2196
     neutral       0.79      0.68      0.73      2166
    positive       0.89      0.93      0.91      5257

    accuracy                           0.85      9619
   macro avg       0.83      0.81      0.82      9619
weighted avg       0.85      0.85      0.85      9619
```

Figure 11: Test Results of Brazilian E Commerce Dataset

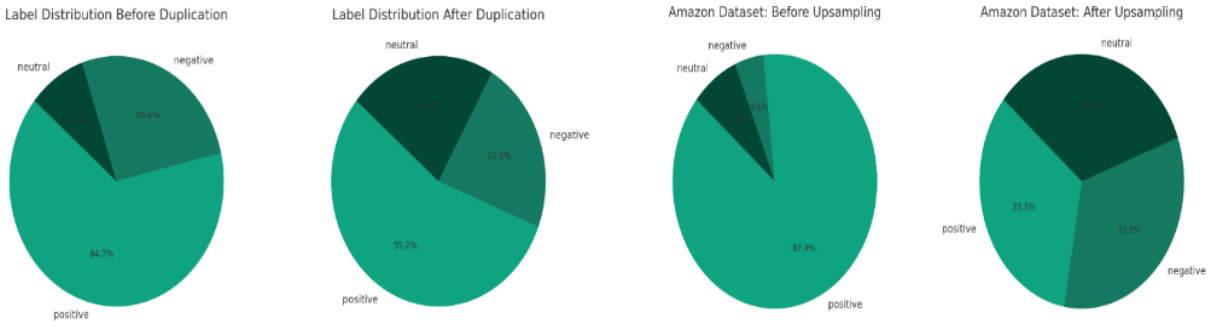


Figure 12: Label Distribution

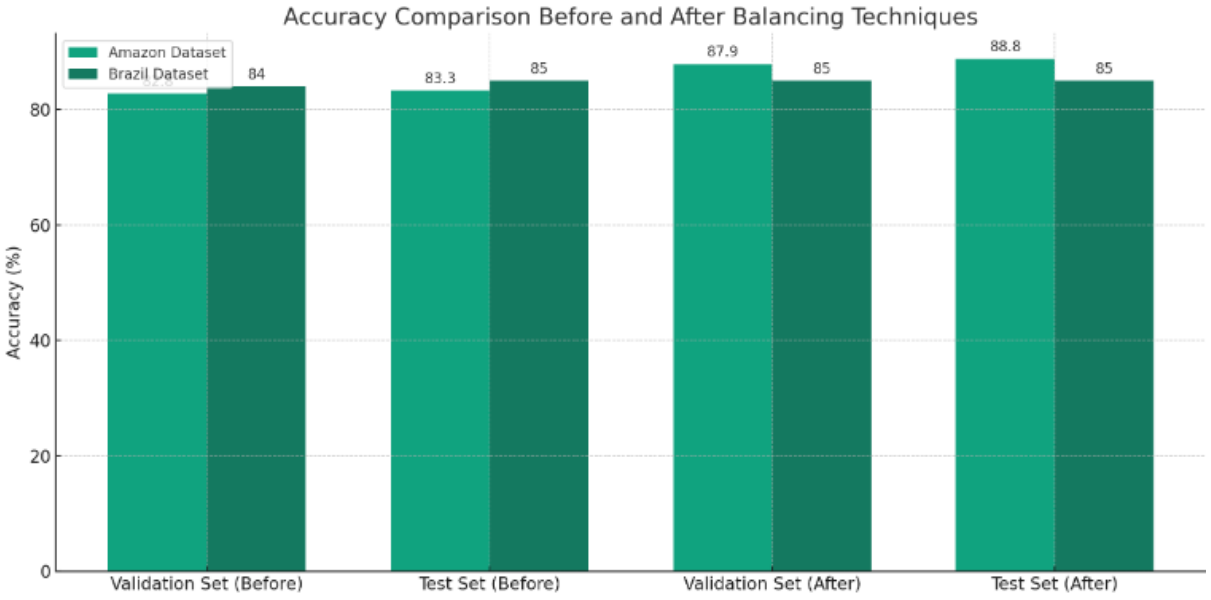


Figure 13: Accuracy Comparison Before and After Techniques

8.3. RNN

First four tests are done with the Brazilian Dataset. Our goal was to compare the hyper parameters one by one and then select the best ones to use in the last test. The last test was done with the Amazon Dataset and we compared all models with this dataset at the end. However, due to time required for the Bert Model we could run it only for two hours to make the comparison fair for the last test we ran the RNN with 2 epochs also.

8.3.1 RNN Results with Epoch = 2, Batch = 32, Lr = 1e-3

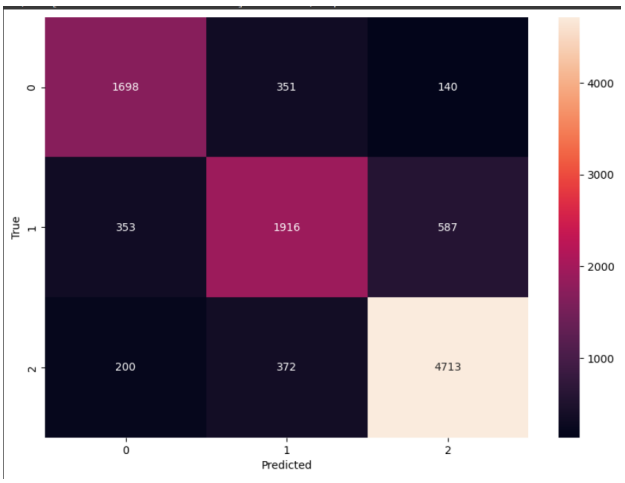


Figure 14:. Confusion Matrix of the RNN Test 1

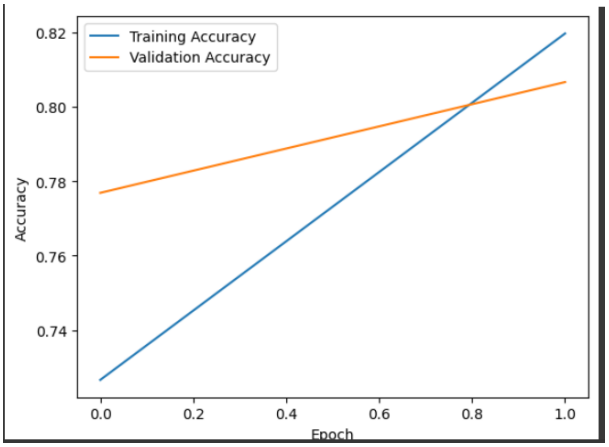


Figure 15: Training Accuracy vs Validation Accuracy Graph of RNN Test 1

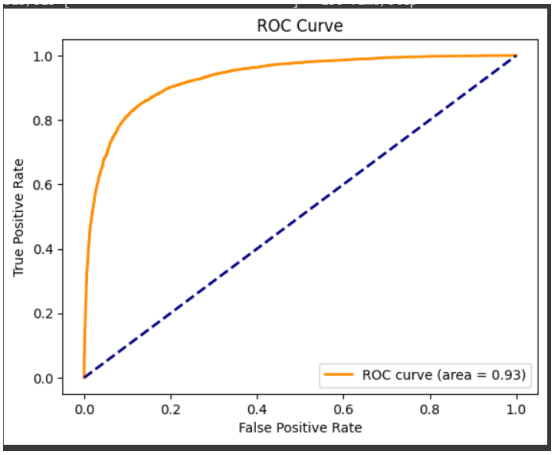


Figure 16:. Roc Curve of the RNN Test 1

Test Accuracy = 0.8

8.3.2 RNN Results with Epoch = 2, Batch = 64, Lr = 1e-2

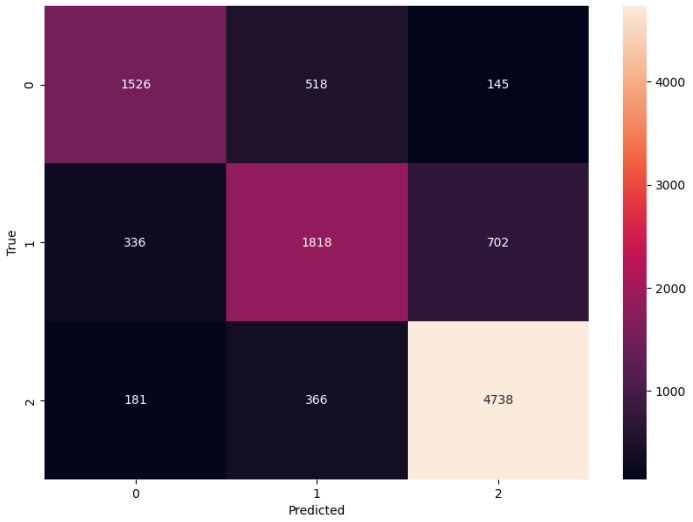


Figure 17: Confusion Matrix of RNN Test 2

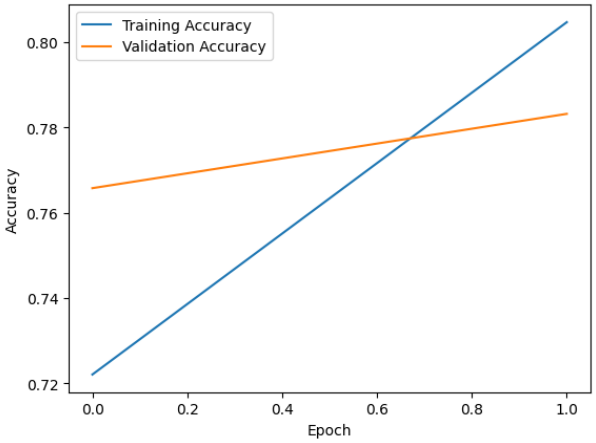


Figure 18: Training Accuracy vs Validation Accuracy Graph of RNN Test 2

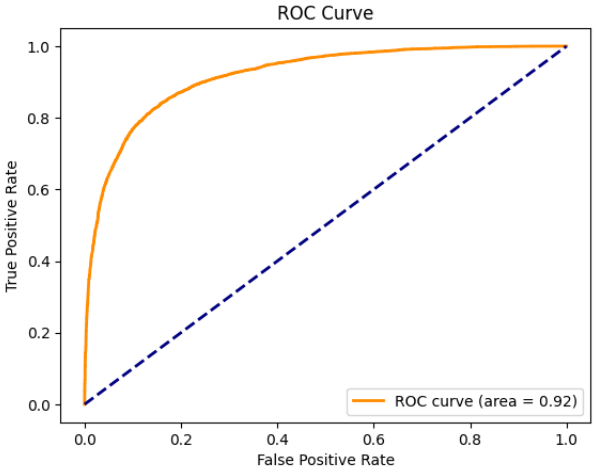


Figure 19: Roc Curve of RNN Test 2

Test Accuracy = 0.78

8.3.3 RNN Results with Epoch = 2, Batch = 64, Lr = 1e-3

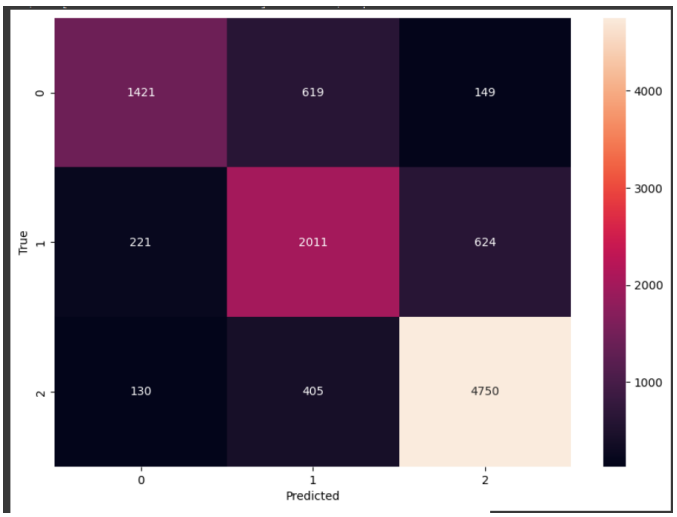


Figure 20: Confusion Matrix of RNN Test 3

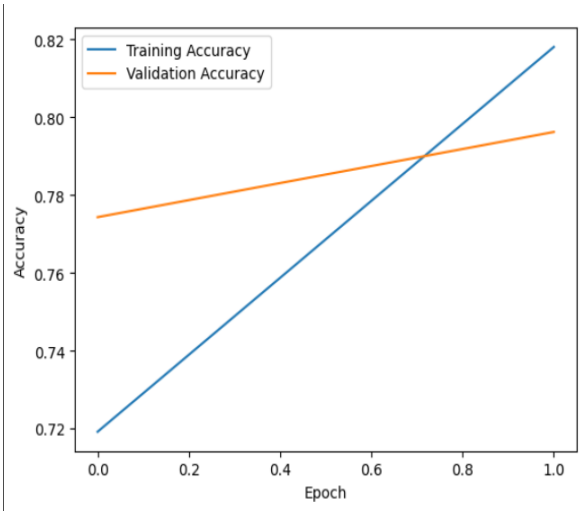


Figure 21: Test Accuracy vs Validation Accuracy of RNN Test 3

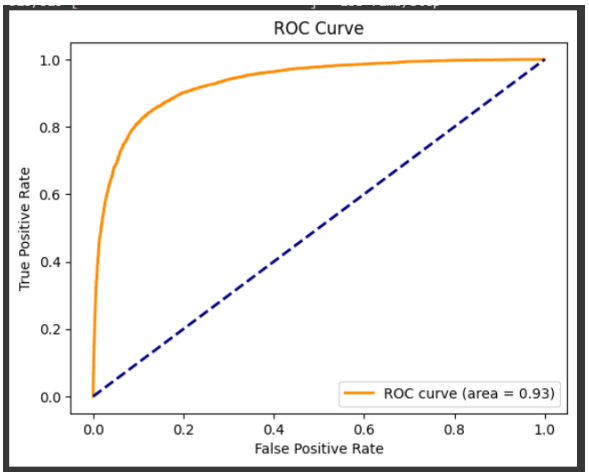


Figure 22: Roc Curve of RNN Test 3

Training Accuracy = 79

8.3.4 RNN Results with Epoch = 10, Batch = 64, Lr = 1e-3

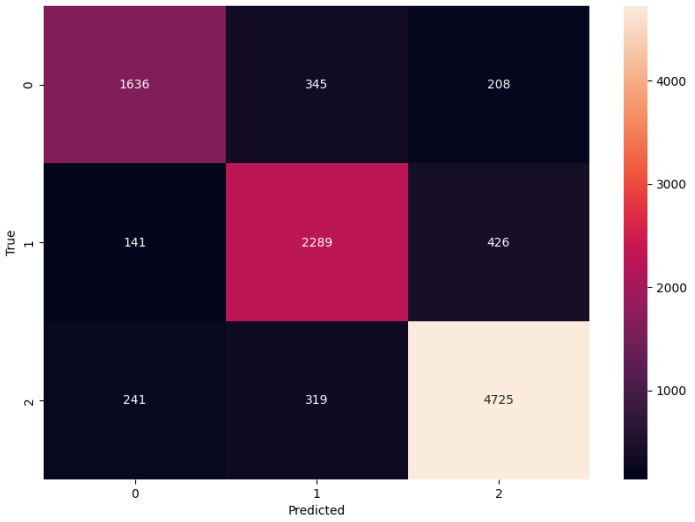


Figure 23: Confusion Matrix of RNN Test 4

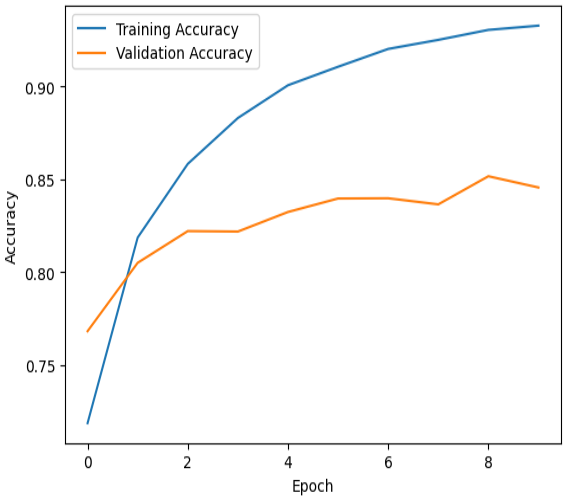


Figure 24: Training Accuracy vs Validation Accuracy Graph of RNN Test 4

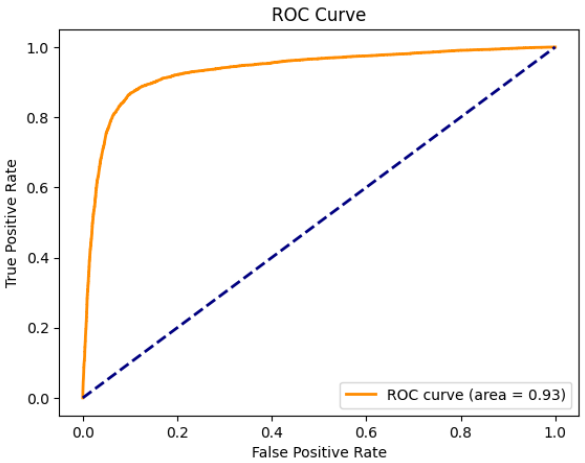


Figure 25: Roc Curve of RNN Test 4

Training Accuracy = 0.84

8.3.5 Hyperparameter Results

In the tests above, it can be observed that epoch = 10 is better than epoch = 2, learning rate = 1e-3 is better than learning rate = 1e-2, batch size = 32 is better than batch size = 64. Hence, we chose learning rate = 1e-3 and batch size = 32 however, as stated above to be fair, in the final comparison we stayed with epoch = 2.

8.3.5 RNN Results with Epoch = 2, Batch = 32, Lr = 1e-3 (AMAZON)

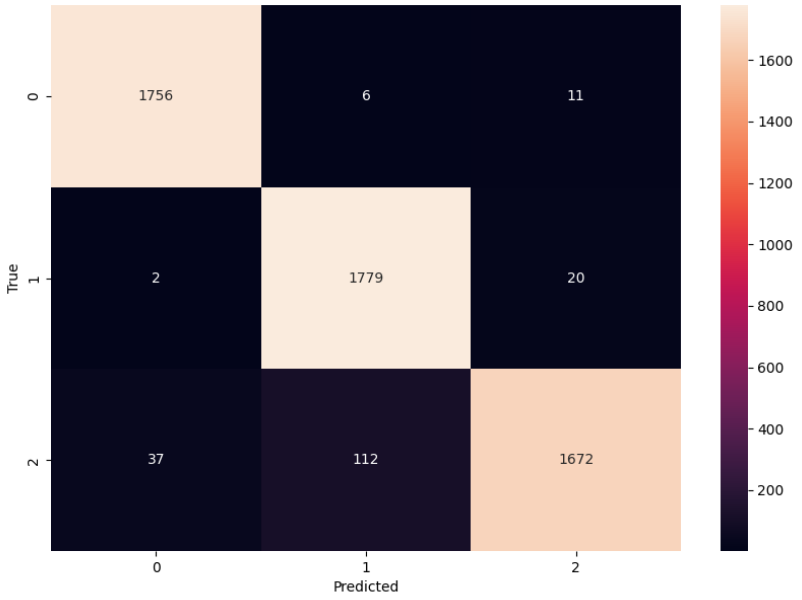


Figure 26: Confusion Matrix of RNN Test 5

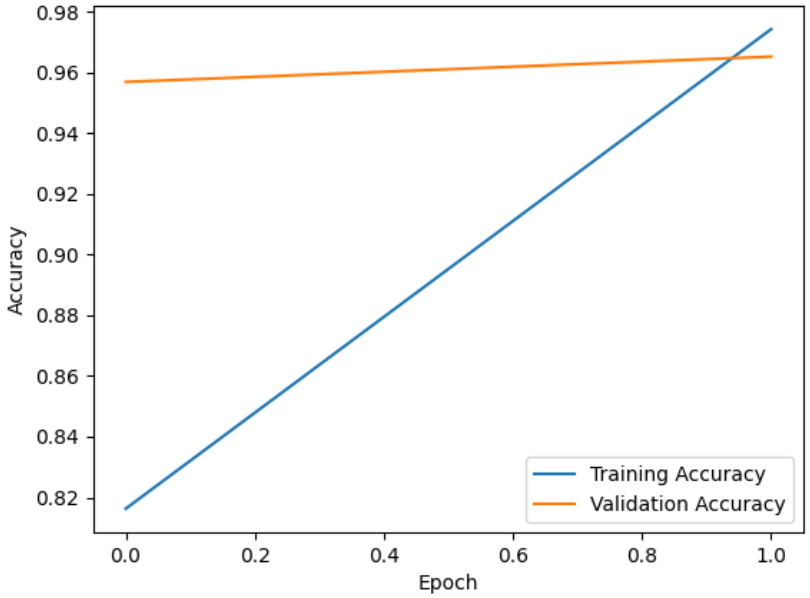


Figure 27: Training Accuracy vs Validation Accuracy Comparison Graph of RNN Test 5

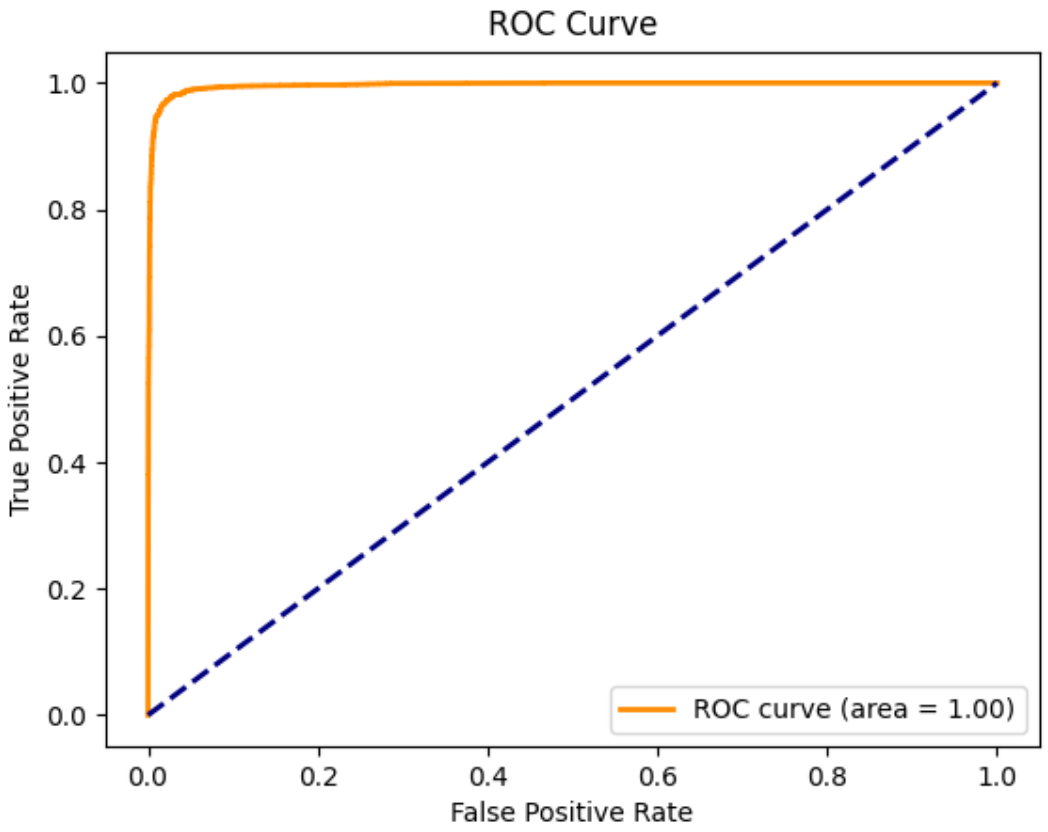


Figure 28: Roc Curve of RNN Test 5

Test Set Classification Report:

	precision	recall	f1-score	support
negative	0.98	0.99	0.98	1773
neutral	0.94	0.99	0.96	1801
positive	0.98	0.92	0.95	1821

Figure 29: Precision and Recall of the Classes

Test Accuracy = 0.965

8.4. BERT

8.4.1. Amazon Dataset Results

Training and Validation Set After Oversampling over Each Epoch:

- Epoch 1:

Training Set Accuracy: 36.92%

Training Loss: 1.0857

Validation Set Accuracy: 63.74%

Validation Loss: 0.9748

- Epoch 2:

Training Set Accuracy: 72.43%

Training Loss: 0.6697

Validation Set Accuracy: 84.06%

Validation Loss: 0.4543

- Epoch 3:

Training Set Accuracy: 87.39%

Training Loss: 0.3653

Validation Set Accuracy: 91.87%

Validation Loss: 0.2515

Test Set After Oversampling:

- Test Set Accuracy: 92.69%
  - Negative Class: Precision - 98%, Recall - 99%, F1-Score - 99%
  - Neutral Class: Precision - 86%, Recall - 94%, F1-Score - 90%
  - Positive Class: Precision - 95%, Recall - 85%, F1-Score - 90%

8.4.2. Key Comparisons

8.4.2.1. Training and Validation Performance over Epochs:

The model underwent three distinct training epochs, each marked by noticeable improvements in both training set accuracy and validation set performance.

Throughout the training process, this BERT model exhibited a noteworthy trend of ascending accuracy, indicating its capacity to learn and adapt to the complexity of sentiment patterns present in the Amazon Dataset. During the initial epoch, the model exhibited a training set accuracy of 36.92%, which then witnessed a subsequent leap to 72.43% in the second epoch and culminated in an expressive 87.39% in the third epoch. This progression illustrates a consistent and substantial improvement in the model's ability to capture sentiment patterns over successive training iterations.

Simultaneously, the validation set mirrored this upward trajectory, validating the model's enhanced capacity for generalization to previously unseen data. Commencing at a validation set accuracy of 63.74%, the second epoch saw a noteworthy elevation to 84.06%, and by the conclusion of the third epoch, the model

achieved an exceptional validation accuracy of 91.87%. These validation accuracies substantiate the model's proficiency in extending its learned representations, affirming a robust and adaptable comprehension of sentiment nuances beyond the boundaries of the training data.

8.4.2.2. Overall Accuracy:

The pivotal metric of overall accuracy encapsulates the model's ability to correctly classify instances across all categories. In the context of the Amazon Dataset, the BERT-based sentiment analysis model achieved an outstanding overall accuracy of 92.69% on the test set after oversampling. This signifies a robust generalization of the model's learned patterns to previously unseen data, substantiating its efficacy in the domain of sentiment classification.

8.4.2.3. Precision, Recall, and F1-Score:

A thorough evaluation of the model's discriminative capabilities necessitated an investigation of precision, recall, and F1-score metrics across distinct sentiment classes.

A high precision emphasized a low incidence of false positives, signifying the accuracy of positive predictions. Concurrently, acquired high recall indicates the model's effectiveness in identifying true positives. This major F1 score provides a comprehensive evaluation of the model's classification performance by harmonizing precision and recall.

The BERT-based sentiment analysis model, exhibited a trajectory of improving performance over epochs, culminating in outstanding accuracy and nuanced sentiment classification across diverse classes.

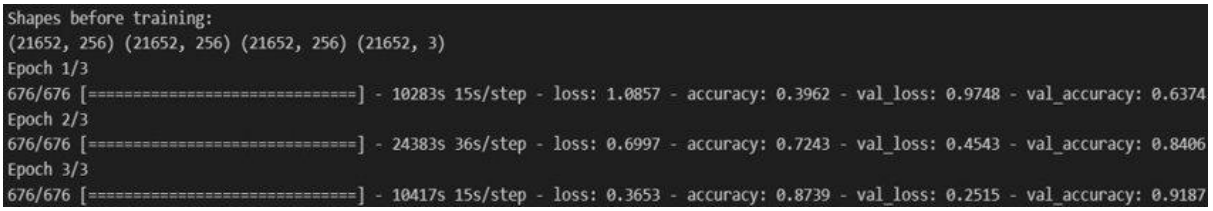


Figure 30: Training and Validation Results of BERT model

```
43/43 [==============================] - 398s 9s/step - loss: 0.2300 - accuracy: 0.9269

Test Set Performance:
Test Loss: 0.22998657822608948, Test Accuracy: 0.9268832802772522
```

Figure 31: Test Results of BERT model

```
Classification Report:
              precision    recall  f1-score   support

    negative       0.98      0.99      0.99       455
     neutral       0.86      0.94      0.90       440
    positive       0.95      0.85      0.90       459

    accuracy                           0.93      1354
   macro avg       0.93      0.93      0.93      1354
weighted avg       0.93      0.93      0.93      1354
```

Figure 32: Classification Report of BERT model

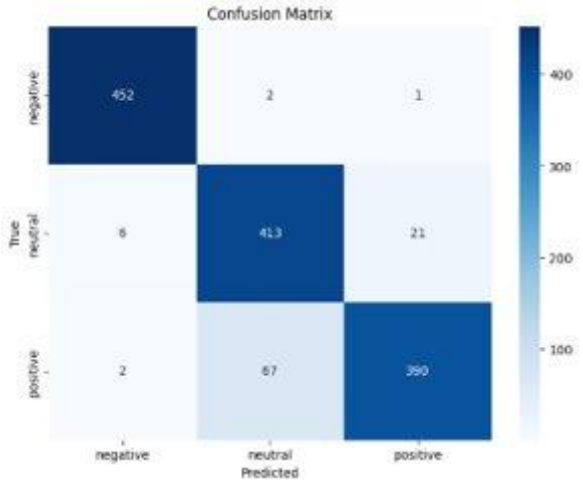


Figure 33: Confusion Matrix for BERT model with Amazon Dataset

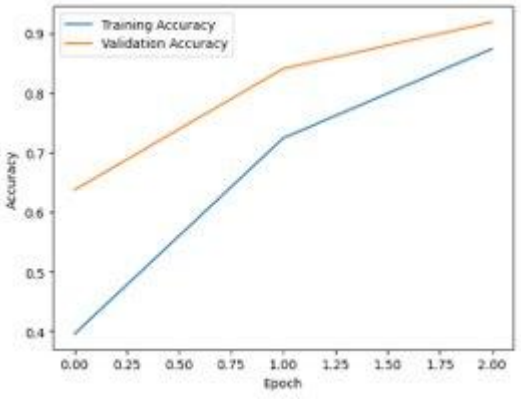


Figure 34: Training and Validation Accuracy Graph per Epoch of BERT model

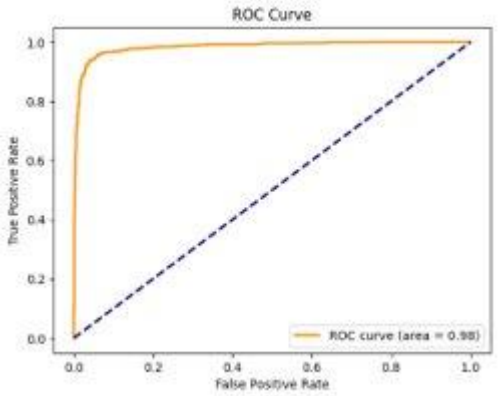


Figure 35: The ROC Curve for BERT model

9. Conclusion

In conclusion, our analysis across multiple models for sentiment analysis on the Amazon and Brazilian E Commerce datasets has yielded insightful findings. The evaluation encompassed Multinomial Naive Bayes (MNB), Support Vector Machine (SVM), Recurrent Neural Network (RNN), and Bidirectional Encoder Representations from Transformers (BERT), each assessed on test accuracy, precision, and recall.

	Multinomial Naïve Bayes			SVM			RNN			BERT		
Test Accuracy	0.88			0.87			0.87			0.93		
	Multinomial Naïve Bayes			**SVM**			**RNN**			**BERT**		
												
Precision	0.89	0.84	0.93	0.89	0.84	0.93	0.95	0.75	0.72	0.95	0.85	0.98
Recall	0.85	0.89	0.92	0.85	0.89	0.92	0.88	0.88	0.80	0.85	0.94	0.99

Figure 36: Comparison of Each Model

BERT emerged as the most accurate, boasting a test accuracy of 0.93, precision of 0.95, and recall of 0.94. These superior results can be attributed to BERT's deep learning capabilities, which enable a nuanced understanding of context within the text data. The RNN model, while slightly less accurate in test results with an accuracy of 0.87, showed a high precision of 0.93, closely rivaling BERT. Notably, its recall rate of 0.92 suggests that RNN is particularly adept at identifying relevant instances throughout the data sequence.

The MNB and SVM models showed commendable performance, with MNB achieving a balance between precision (0.89) and recall (0.85), and SVM displaying a strong precision of 0.89. However, both models fell short of the deep learning approaches in overall test accuracy.

While BERT leads in overall performance, it's crucial to consider the efficiency of the RNN model. Given its less computationally intensive nature, RNN offers a promising trade-off between performance and resource utilization. If provided with runtime comparable to BERT's, there's a potential for RNN to enhance its accuracy further and possibly rival BERT's results. This efficiency makes RNN an attractive option for environments where computational resources are a limiting factor.

Result of these observations, BERT stands out for its superior performance in precision and recall, affirming its strength in sentiment analysis tasks. However, RNN's efficiency and high precision present a compelling case for scenarios where runtime and resource optimization are paramount. Future studies could explore optimizing RNN's architecture and training process to close the gap with BERT's performance, potentially offering a more efficient yet equally effective solution for sentiment analysis.

10. Work Distribution

21903488 Emrehan Hoşver: 4. Coding environment summary, 5.3 RNN (Details of the trained models and configuration), 8.3 RNN (Comparing and discussing the evaluation results of models)

21901798 Gökberk Altıparmak: 3. Dataset split, preprocessing, and detailed analysis., 5.2 SVM (Details of the trained models and configuration), 8.2 SVM (Comparing and discussing the evaluation results of models)

21802480 Tuğberk Dikmen: 1. Introduction, 2. Background Information, 5.1. MNB (Details of the trained models and configuration), 6. Using appropriate evaluation metrics, 7. Applying evaluations on the correct data splits, 8.1. MNB (Comparing and discussing the evaluation results of models), 9. Conclusion for the final report. Implementation of MNB for Brazilian E Commerce Dataset, SVM and RNN for Amazon Dataset.

21901418 Efser Efe Kantik: 1. Introduction, 3. Dataset split, preprocessing and detailed analysis, 5.1 MNB (Building with Oversampling Method), 7. Applying evaluations on the correct data splits, 8.4 BERT (Comparing and discussing the evaluation results of models), 9. Conclusion for the final report,

Devising of Oversampling, Implementation of MNB for Amazon Dataset, SVM for Brazilian E Commerce Dataset, BERT for Amazon Dataset.

References

[1] Olist, "Brazilian E-Commerce Public Dataset by Olist, " Kaggle. [Online]. Available: https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce [Accessed: Oct 19, 2023].

[2]"1.9.NaiveBayes,"scikit-learn.org.[Online]. Available:https://scikit-learn.org/stable/modules/naive_bayes.html [Accessed: Nov 22, 2023].

[3]"1.4. Support Vector Machines," scikit-learn.org. [Online]. Available: https://scikit-learn.org/stable/modules/svm.html# [Accessed: Nov 23, 2023].

[4]"sklearn.svm.SVC,"scikit-learn.org.[Online]. Available:https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC [Accessed: Nov 23, 2023]

[5]E. Chand, "Amazon Musical Instruments Reviews " Kaggle. [Online]. Available:https://www.kaggle.com/datasets/eswarchandt/amazon-music-reviews [Accessed: Oct 25, 2023].

[6] A.Amidi and S.Amidi, "Recurrent Neural Networks Cheatsheet", *cs230deepleraning.* [Online]. Available: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks [Accessed: Dec. 18, 2023]