NLP Emotion Classification Project Report

Parth Rastogi

June 7 , 2024

Contents

Chapter 1

Introduction

1.1 Overview

This report details the processes and methodologies employed in a Natural Language Processing (NLP) classification project using a dataset from Kaggle: `https://www.kaggle.com/datasets/nelgiriyewithana/emotions/data`. The goal of this project is to classify text data into six distinct emotion categories: sadness, joy, love, anger, fear, and surprise.

1.2 Dataset

The dataset consists of 416,808 entries with each entry containing text data and a corresponding emotion label. The class labels are as follows:

Label	**Emotion**
0	Sadness
1	Joy
2	Love
3	Anger
4	Fear
5	Surprise

Table 1.1: Class Labels and Corresponding Emotions

Chapter 2

Preprocessing of Data

2.1 Text Cleaning

Text cleaning involves removing unnecessary characters, symbols, or special characters from the text data. Additionally, all text is converted to lowercase to maintain consistency. This step ensures that the text is in a suitable format for further processing.

- Removal of punctuations and special characters.
- Conversion to lowercase.

2.2 Tokenization

Tokenization is the process of splitting the text into individual words or tokens. This allows for the analysis of individual words rather than entire text blocks, which is crucial for further processing.

- Sentence tokenization.
- Word tokenization.

2.3 Vectorization

Vectorization converts text data into numerical representations for use in machine learning algorithms. Technique used in this project include:

- Bag-of-Words (BoW)
- Term Frequency-Inverse Document Frequency (TF-IDF)

Chapter 3

Exploratory Data Analysis (EDA)

3.1 Class Frequency Analysis

To understand the most common and important classes in the dataset, a class frequency distribution was created.

- Frequency distribution analysis.

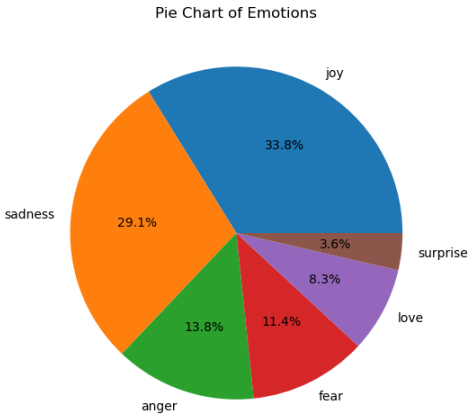


Figure 3.1: Class Frequency Pie Chart

- Visualizations using bar charts.

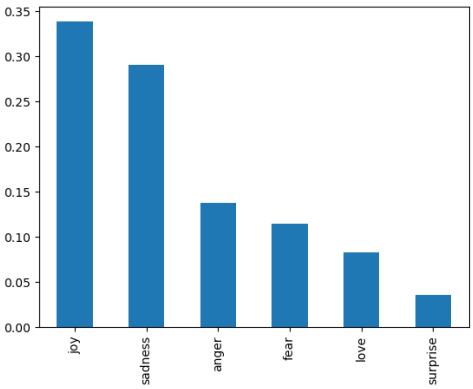


Figure 3.2: Class Frequency Bar Chart

3.2 Word Clouds

Word clouds visually represent the most frequent words in the text corpus. Two word clouds were created to compare the word frequencies before and after text cleaning.

Figure 3.3: Word Cloud Before Text Cleaning

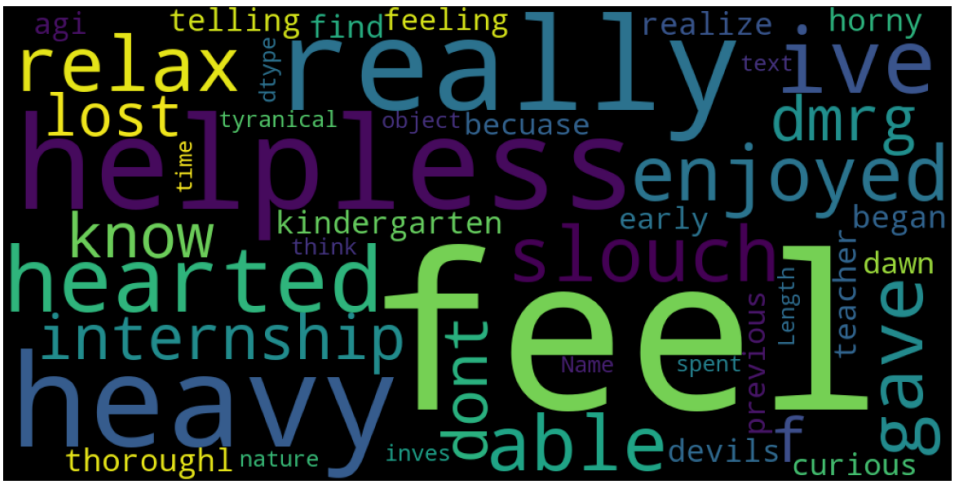


Figure 3.4: Word Cloud After Text Cleaning

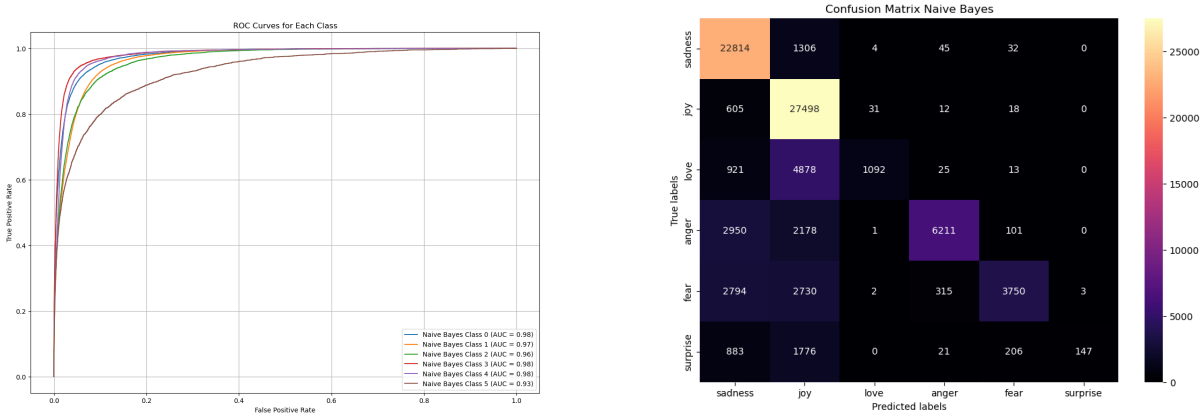
Chapter 4

Classification Models

4.1 Naive Bayes

Naive Bayes is a probabilistic classifier based on Bayes' theorem with the assumption of independence between predictors.

- Model accuracy :74%
- Model training and evaluation metrics.



(a) ROC curve

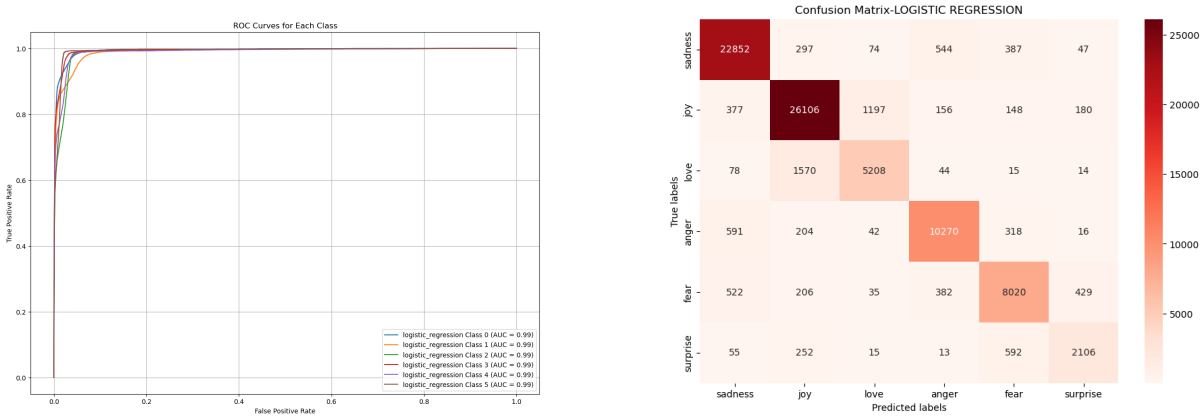
(b) Confusion Matrix

Figure 4.1: Evaluation Metric for XG boost

4.2 Logistic Regression

Logistic Regression is a statistical model that predicts the probability of a binary outcome.

- Model accuracy :89.44%
- Model training and evaluation metrics.



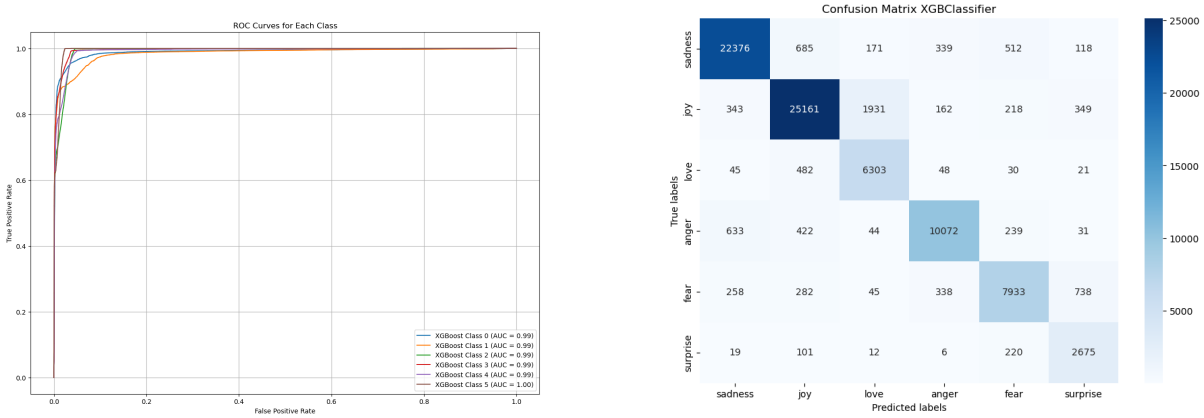
(a) ROC curve (b) Confusion Matrix

Figure 4.2: Evaluation Metric for logistic regression

4.3 XGBoost

XGBoost is an optimized gradient boosting algorithm designed for speed and performance.

- Model accuracy :89.39%
- Model training and evaluation metrics.



(a) ROC curve (b) Confusion Matrix

Figure 4.3: Evaluation Metric for XG boost

4.4 Custom Neural Network

A custom neural network was developed for this project. The architecture and specifics are detailed below:

4.4.1 Network Architecture

The custom neural network consists of several fully connected layers. The architecture includes:

- Input Layer: The input layer size corresponds to the number of features in the input vector.
- Hidden Layers: Each hidden layer utilizes the LeakyReLU activation function to allow a small gradient when the unit is not active, mitigating the vanishing gradient problem.
- Output Layer: The output layer uses a sigmoid activation function to ensure the output is a probability value, suitable for classification tasks.

4.4.2 Activation Functions

- **Leaky ReLU**: The Leaky ReLU function allows a small, non-zero gradient when the input is negative, given by:

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{otherwise} \end{cases}$$

  where α is a small constant (I used 0.01).

- **Sigmoid**: The sigmoid function, used in the output layer, is defined as:

$$f(x) = \frac{1}{1 + e^{-x}}$$

  It compresses input values into a range between 0 and 1, making it ideal for outputting probabilities.

4.4.3 Training Process

The model was trained for 100 epochs, which may not be sufficient for full convergence, suggesting the possibility for further optimization. The training process involved:

- **Loss Function**: Log loss (binary cross-entropy) was used, defined as:

$$\text{Log Loss} = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)]$$

  where y_i is the true label and p_i is the predicted probability.

- **Optimizer**: A gradient-based optimizer was employed to minimize the log loss, updating weights to reduce prediction error iteratively.

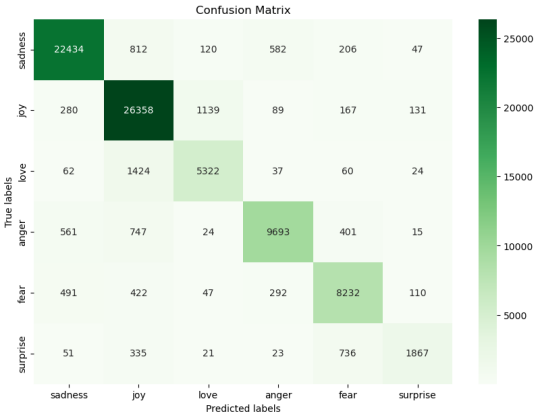
4.4.4 Potential Limitations

Training for only 100 epochs might not have allowed the model to fully converge, which can impact its performance. More epochs or a more complex learning rate schedule could potentially improve results.

4.4.5 Results and Performance

.

- Model accuracy :88.66%
- Model training and evaluation metrics.



(a) Confusion Matrix

Figure 4.4: Evaluation Metric for logistic regression

Chapter 5

Results and Application

5.1 Model Performance

Comparison of model performances based on accuracy, precision, recall, and F1-score.

Model	Accuracy	Precision	Recall	F1-Score
Naive Bayes	0.74	0.79	0.74	0.69
Logistic Regression	0.8944	0.89	0.89	0.89
XGBoost	0.8939	0.90	0.89	0.90
Neural Network	0.8866	0.89	0.89	0.89

Table 5.1: Comparison of Model Performance Metrics

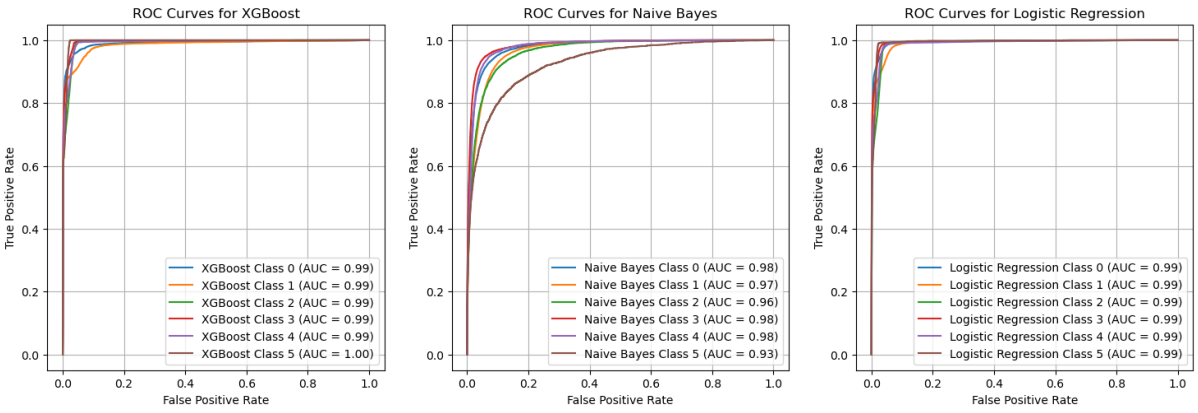


Figure 5.1: Comparison of ROC of all the models from Scikit-Learn

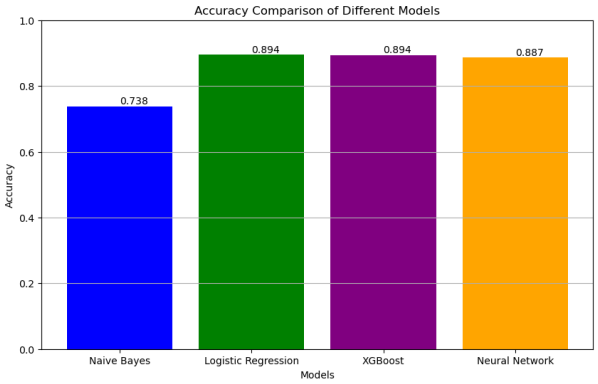


Figure 5.2: Comparison of accuracy of all models

5.2 Application:- the CLI Tool

The Emotion Prediction CLI (Command-Line Interface) tool "cliapp.py" is designed to classify textual input into one of six predefined emotion categories: sadness, joy, love, anger, fear, and surprise. It uses machine learning models that have been trained on a dataset of emotional text to predict the emotion conveyed in any given sentence.

5.2.1 Tool Description

The CLI tool interacts with three different machine learning models to predict emotions:

- **XGBoost Model**
- **Logistic Regression Model**
- **Naive Bayes Model**

When a sentence is inputted into the tool, it undergoes a series of preprocessing steps such as lowercasing, punctuation removal, and removal of URLs and special characters. The cleaned text is then transformed into a numerical representation using a pre-trained TF-IDF vectorizer. This vectorized text is fed into the models to predict the associated emotion.

5.2.2 Functionality

The CLI tool performs the following key steps:

- **Text Preprocessing**: The input text is cleaned by removing unnecessary characters and converting it to a consistent format.
- **Vectorization**: The cleaned text is transformed into a numerical vector using a TF-IDF vectorizer.
- **Prediction**: The vectorized text is fed into the models (XGBoost, Logistic Regression, and Naive Bayes) to obtain emotion predictions.
- **User Interaction**: The tool provides a command-line interface where users can input sentences and receive emotion predictions from each model.

5.2.3 Sample Inputs and Outputs

Below are some examples of sentences and the corresponding predicted emotions from each model:

- **Input**: "I am so happy and surprised you did this!"
  - **Predicted Emotion (XGBoost)**: Joy
  - **Predicted Emotion (Logistic Regression)**: Joy
  - **Predicted Emotion (Naive Bayes)**: Surprise
- **Input**: "I am already feeling frantic."
  - **Predicted Emotion (XGBoost)**: Fear
  - **Predicted Emotion (Logistic Regression)**: Fear
  - **Predicted Emotion (Naive Bayes)**: Fear

5.2.4 Usage Instructions

To use the CLI tool:

1. Launch the script from the command line.
2. Enter any sentence to receive an emotion prediction.
3. The tool will display the predicted emotion for each of the three models.
4. Type 'exit' to quit the tool.