

Title: Emotion Aware Detection

**Submitted by:**

Atta ul mohamin 2020-CS-43

Hafiz Haseeb Zahid 2020-CS-20

Hassan Nisar 2020-CS-49

**Submitted to:**

:

Dr. Usman Ghani

Department of Computer Science

University of Engineering and Technology Lahore

**1. Introduction**

**1.1 Project Description**

Emotions are expressed through words, gestures, and expression, and also with social media today, but emotion is now expressed through tweets and Instagram.

We build a web application for Text emotion detection/ Text emotion classification using Streamlit and Python. In this project, we will be taking the text as an input and classifying that text into emotions like- joy, sadness, fear, neutral, anger, shame, etc. using **Natural Language Processing.**

**1.2 Purpose**

The purpose of the Text Emotion Detection Chatbot with Streamlit project is to create an interactive tool capable of analyzing and detecting emotions in textual input. By leveraging machine learning models, the chatbot interprets user-provided text and predicts the associated emotion.

**1.3 Goals**

Develop a user-friendly chatbot that detects a range of emotions, providing a personalized and engaging experience.

Integrate the chatbot with the Streamlit framework to enable easy deployment and accessibility.

1.3 Motivation

The motivation behind this project is rooted in the increasing need for understanding sentiment and emotions expressed in text. By building a chatbot, we aim to simplify this process and provide a tool that can assist in tasks where emotional context is crucial.

**2. Project Overview**

**2.1 Technologies Used**

**Streamlit**

Streamlit is chosen for its simplicity in creating web applications with minimal code. Its interactive and real-time updates make it an ideal choice for the chatbot interface.

**Pandas, NumPy, Altair**

Pandas and NumPy are used for data manipulation, while Altair facilitates the creation of interactive visualizations for displaying emotion probabilities.

**Joblib**

Joblib is employed for saving and loading the machine learning model efficiently.

**2.2 Architecture Overview**

The project follows a modular architecture to ensure scalability and maintainability. The key components include the emotion detection model, Streamlit interface, and auxiliary modules for data processing.

**2.3 Emotion Detection Model**

The emotion detection model is a machine learning model trained on a dataset labeled with various emotions. It uses natural language processing techniques to understand and predict the emotion expressed in text.

**2.4 Emoji Mapping**

To enhance the user experience, a mapping between predicted emotions and emojis is established. This adds a visual component to the results, making them more intuitive and engaging.

**2.5 Streamlit Interface**

The Streamlit interface consists of an input area for users to type text and a dynamic display area for showing the original text, predicted emotion, associated emoji, and confidence level. The interface is designed to be intuitive and user-friendly.

**3. Emotion Detection Model**

**3.1 Machine Learning Model**

The machine learning model is a pipeline that combines a CountVectorizer for text representation with a Logistic Regression classifier. The model is trained on a labeled dataset of text samples and corresponding emotions.

**3.2 Data Preprocessing**

Data preprocessing involves cleaning the text data by removing user handles and stopwords. This ensures that the model focuses on the meaningful content of the text.

**3.3 Model Training**

The model is trained using a labeled dataset, and its performance is evaluated based on metrics such as accuracy and F1 score.

**3.4 Model Evaluation**

The trained model is evaluated on a separate test set to assess its generalization performance. Metrics such as precision, recall, and confusion matrix are considered.

**4. Emoji Mapping**

The project incorporates a dictionary that maps predicted emotions to corresponding emojis. This mapping enhances the interpretability of results and adds a visually appealing element to the chatbot's output.

**5. Streamlit Interface**

**5.1 Input Area**

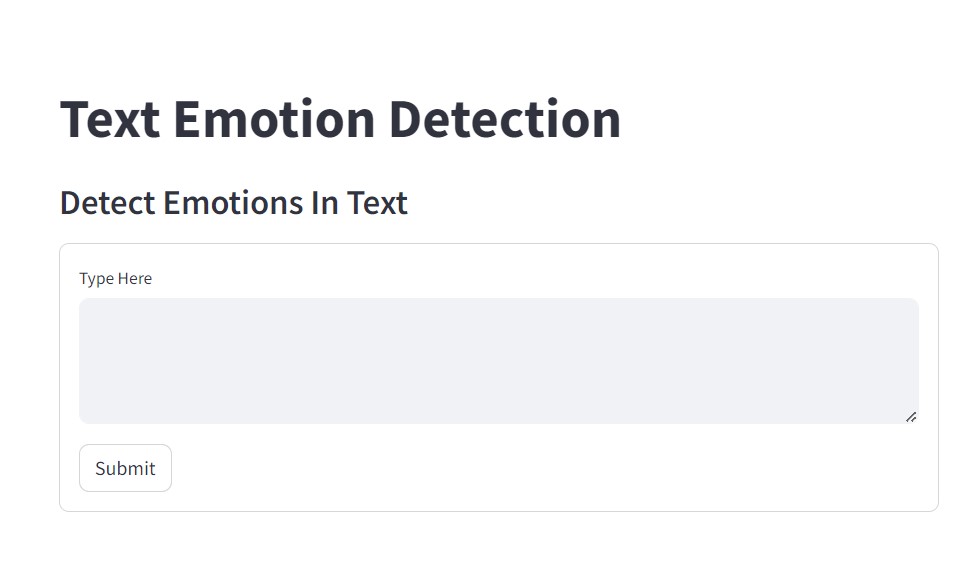
Users interact with the chatbot by typing text into the input area. The interface is designed to be user-friendly, providing a seamless experience for inputting textual data.

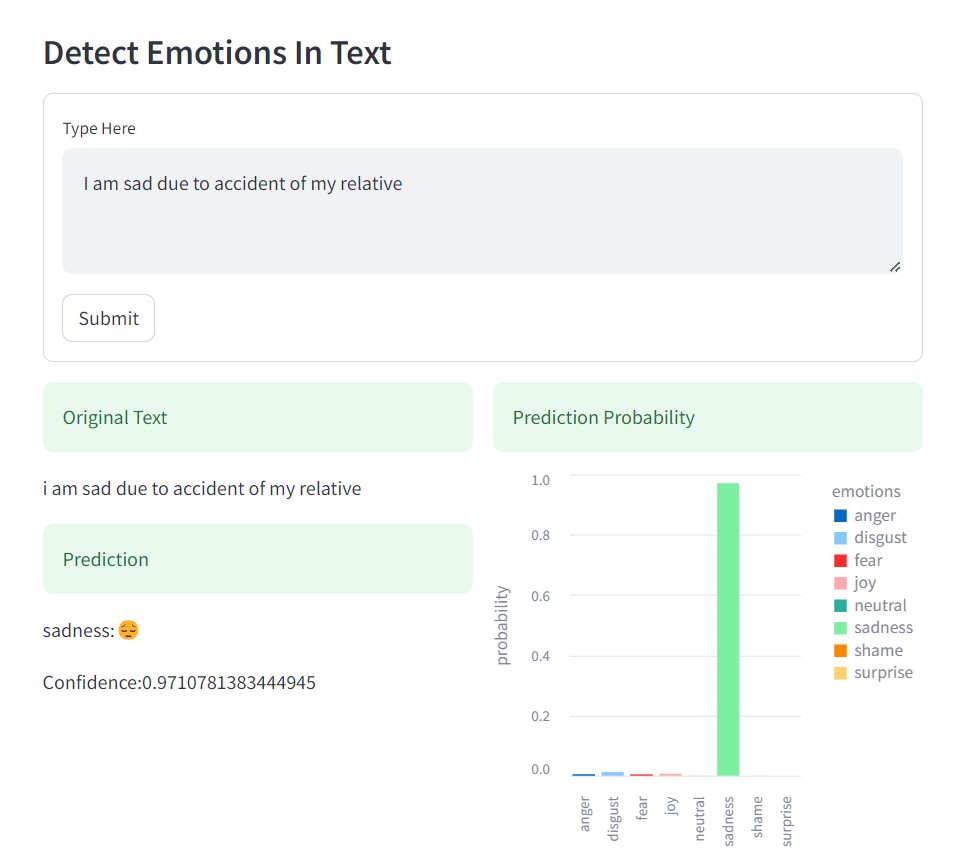
**5.2 Output Display**

Upon submitting text, the chatbot displays the original input, the predicted emotion, the associated emoji, and the confidence level. This dynamic display ensures real-time feedback to users.

**5.3 Live Demo**

A live demonstration of the chatbot is included to showcase its functionality. Users can experience the chatbot's capabilities by providing their own input and observing the predictions.





**6. Code Overview**

**6.1 Directory Structure**

The project follows a structured directory layout. Key directories include:

**model**: Contains the serialized machine learning model.

**data**: Houses the labeled dataset used for model training.

**scripts**: Holds Python scripts for data processing, training, and Streamlit app code.

Input and output components for user interaction.

Integration with the trained machine learning model.

Visualization using Altair for displaying emotion **probabilities.**