Project Overview & Guide

Sentiment Analysis

Introduction

The Sentiment Analysis project is designed to develop a robust model capable of automatically classifying text and social media posts into one of three categories: positive, negative, or neutral. This project leverages natural language processing techniques to gain valuable insights from textual data.

# Key Features

## Text Classification

* Create a text classification model that can accurately determine the sentiment behind text data.

## Multi-Class Classification

* Implement a multi-class classification system to classify text into one of three categories: positive, negative, or neutral.

## Natural Language Processing (NLP)

* Utilize NLP techniques, such as tokenization, word embeddings and feature engineering, to process and analyse textual data effectively.

# Check List

1. Define Project Objectives:

* Clearly define the project's objectives, including the type of text data to analyse and the sentiment categories to classify.

2. Data Collection:

* Gather a substantial dataset of text and social media posts with associated sentiment labels (positive, negative, neutral).
* Ensure diversity and representativeness in the data, covering relevant topics and domains.

3. Data Pre-processing:

* Clean and pre-process the dataset, addressing issues like spelling errors, special characters, and text normalization.
* Tokenize the text data for analysis.

4. Feature Engineering:

* Convert text data into numerical features using techniques like word embeddings (e.g., Word2Vec, GloVe) or TF-IDF.
* Select appropriate features and representations for sentiment classification.

5. Model Selection:

* Choose an appropriate machine learning or deep learning model for sentiment analysis.
* Consider models like Naive Bayes, Support Vector Machines and Recurrent Neural Networks (RNNs), or Transformer-based models.

6. Multi-Class Classification:

* Design the model to classify text into one of three sentiment categories: positive, negative, or neutral.
* Fine-tune the model to achieve high classification accuracy.

7. Training and Validation:

* Train the model using the pre-processed text data and engineered features.
* Perform cross-validation or holdout validation to assess the model's performance.

8. Model Evaluation Metrics:

* Assess the model's performance using metrics like accuracy, precision, recall, F1-score, and confusion matrices.
* Validate the model's ability to accurately classify text into the correct sentiment category.

9. Hyperparameter Tuning:

* Experiment with different hyperparameters to optimize model performance.
* Fine-tune hyperparameters based on validation results.

10. User Interface (Optional):

* Create a user-friendly interface for users to input text data and receive sentiment analysis results.