Project Report: Multi-Class Sentiment Analysis of Tweets

Course: Business Economics

Project Topic: Tweets Sentiment Analysis

Group Members :

- Debanjan Maji (2023BCS017)
- Vansh Khardam (2023BCS070)

1. Anurag Singh (2023BCS010)

Problem Statement:

This project undertakes a multi-class sentiment analysis of tweets, aiming to classify text into one of four distinct categories:

`Positive`, `Negative`, `Neutral`, or `Irrelevant` . This objective presents a more complex and nuanced challenge than simpler binary classifications (e.g., hateful/not-hateful).

Dataset: It uses the "Twitter Entity Sentiment Analysis" dataset, which contains 74,682 training tweets.
Classification Task: The model classifies tweets into one of four categories: **Positive, Negative, Neutral, or Irrelevant**.

2. Methodology and Key Changes

The project's methodology employs a modern deep learning pipeline, representing a significant advancement over traditional machine learning techniques. This approach includes advanced data preprocessing, robust feature engineering, and a sophisticated neural network architecture.

- **Dataset:** Uses the "Twitter Entity Sentiment Analysis" dataset, which is much larger (74,682 training tweets) and more complex (4 sentiment classes).
- **Preprocessing:** Uses **Lemmatization** instead of stemming.
- **Feature Engineering:** Uses **TF-IDF** instead of a simple Bag-of-Words (TDM).
- **Modeling:** Uses a **Long Short-Term Memory (LSTM) Neural Network** instead of traditional ML models like CART

- **Data Cleaning & Preprocessing:**
  - Removes duplicate entries and rows with missing data.
  - Converts all text to lowercase.
  - Removes special characters, punctuation, and stopwords.
  - Applies **Lemmatization** (e.g., "running" becomes "run"), which is a more advanced technique than the stemming used in the PDF article.
- **Feature Engineering:**
  - The notebook uses **TF-IDF (`TfidfVectorizer`)** with a vocabulary size of 5,000 features. This method is a significant upgrade from the Bag-of-Words/TDM approach in the article, as it weighs words based on their importance and rarity.
- **Modeling:**
  - The primary model used is a **Deep Learning** model, specifically an **LSTM (Long Short-Term Memory)** network.
  - An LSTM is a type of Recurrent Neural Network (RNN) that is highly effective at understanding the *sequence and context* of words, which is crucial for NLP.
- **Results:**
  - The model achieves an accuracy of approximately **96.1%** on the validation data.

---

3. Results

The implemented LSTM model demonstrated strong performance, achieving an accuracy of approximately 96.1% on the validation dataset. This high level of accuracy validates the chosen methodology, confirming that the combination of deep learning and advanced preprocessing is highly effective for this complex four-class classification task.

4. Comparative Analysis

This project's methodology can be contrasted with a more traditional approach, such as one detailed in a hypothetical PDF article using simpler techniques. The project's modern pipeline provides clear and significant advantages.

4.1. Preprocessing: Lemmatization vs. Stemming

The traditional method (PDF Article) might use **Stemming**, a crude heuristic that chops off word endings (e.g., "service" becomes "servic"). This can lead to non-existent words and a loss of meaning. This project's use of **Lemmatization** is linguistically more sophisticated, ensuring that the root word is always a valid dictionary word, thus preserving semantic meaning for the model.

4.2. Classification Model: LSTM vs. CART

The most significant divergence lies in the choice of the classification model.

- **Traditional Method (CART):** The PDF article's posited model is a **CART (Classification and Regression Tree)**, or Decision Tree. A CART model learns by creating a series of simple "yes/no" rules based on word counts (e.g., "Does this tweet contain 'love' > 0 times?"). While fast and interpretable, this model's primary

disadvantage is that it **completely ignores the order and context of words**. It cannot differentiate between "not good" and "good, not bad," as it only sees the counts of "not" and "good."

- **Project Method (LSTM):** This project's **LSTM** network is fundamentally superior for this task. As a recurrent network, it is *specifically designed* to analyze sequences. The model reads the tweet word by word and remembers the preceding context. It learns that the word "not" placed before "good" reverses the sentiment, while "not" before "bad" makes it positive.

This ability to understand context, word order, and long-range dependencies within the text makes the LSTM model significantly more powerful and accurate for any nuanced NLP task, as demonstrated by the high validation accuracy achieved in this project.